Brandenburgische Technische Universität
Cottbus – Senftenberg
Fakultät 3 - Institut für Elektronik und Informationstechnik
Lehrstuhl Kommunikationstechnik
Professor Dr.-Ing. habil Matthias Wolff

Brandenburgische
Technische Universität
Cottbus - Senftenberg

– Bachelorarbeit –

Rückweisung ungrammatischer Spracherkenner-Eingaben

Leonard Förster, 3041105

Abgabe: 05. März 2014
Verteidigung: 25. April 2014
Finale Version: 01. Mai 2014

Betreuer: Prof. Dr.-Ing. habil Matthias Wolff

Eidesstattliche Erklärung

Der Verfasser erklärt an Eides statt, dass er die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Ort, Datum Unterschrift des Verfassers

Inhaltsverzeichnis

1 Einführung **1**

2 Konfidenzmasse: Bewertung und Stand der Technik **3**

2.1 Konfidenzmaße 3

2.2 Stand der Technik 6

2.2.1 Kombination von Merkmalen 6

2.2.2 A-posteriori-Wahrscheinlichkeiten auf Wortgraphen 9

2.2.3 Konfidenzschätzung im UASR System 10

3 Testdaten **11**

3.1 Wizard-of-Oz-Experiment 11

3.1.1 Aufbau im SpeechLab 11

3.1.2 Aufbau Wizard-of-Oz-Experiment 12

3.2 Testdaten 15

4 Theoretische Grundlagen **17**

4.1 Hidden Markov Modelle 17

4.2 Konfidenzintervall von Detektionsergebnissen 19

4.3 Erkennerkonfidenz in UASR 20

4.4 Merkmale und Konfidenz 21

4.4.1 NAD 21

4.4.2 NHD 21

4.4.3 NWHD 21

4.4.4 NWLD 22

4.4.5 PDL 23

4.4.6 PDLW 24

4.4.7 Konfidenzberechnung 24

5 Evaluation **27**

5.1 Kalibrierung 27

5.2 Merkmalskombinationen 29

6 Zusammenfassung und Ausblick **33**

A Handout Wizard-of-Oz-Experiment **35**

B Liste der aufgenommenen Spracheingaben **39**
B.1 Proband 1 39
B.2 Proband 2 40
B.3 Proband 3 42
B.4 Proband 4 45
B.5 Proband 5 46
B.6 Proband 6 48
B.7 Proband 7 49
B.8 Proband 8 52
B.9 Proband 9 54
B.10 Proband 10 55
B.11 Proband 11 58

C Phonemdauerstatistiken **63**

D Standardisierte Histogramme der Merkmale **71**

E Codelisting Konfidenzschätzung **75**

F Inhalt der DVD **89**
F.1 Einrichtung dLabPro und UASR 89
F.2 Installation Testkorpus 90
F.3 Nutzung des eingerichteten Systems 90

G Abkürzungsverzeichnis **93**

Literaturverzeichnis **95**

1 Einführung

Die automatische Spracherkennung hat in vielen Bereichen des täglichen Lebens Einzug gehalten. So findet man sie heute in Handys, Navigationsgeräten, Diktiersystemen und vielen weiteren Anwendungen. Jeder der ein solches System schon einmal genutzt hat, wird festgestellt haben, dass es noch massiven Verbesserungsbedarf gibt. Unter Realbedingungen außerhalb des Labors werden die Erkennerergebnisse, sprecherabhängig und durch Umgebungsgeräusche, sowie viele weitere Faktoren, stark beeinträchtigt [1].

Es ist offensichtlich, dass es mit all diesen Fehlerquellen und der nötigen vereinfachenden Modellbildung nahezu unmöglich ist eine fehlerfreie Erkennung zu gewährleisten. Da Erkennungsfehler nicht ausgeschlossen werden können bzw. unvermeidbar sind, ist es anzustreben, dass diese Fehler wenigstens identifiziert und im Idealfall sogar korrigiert werden können (vgl. Berton, S. 39f [2]). Gängige Fragestellungen sind:

- Handelt es sich bei der Eingabe um Sprache? (Trennung von Sprache und Umgebungsgeräuschen)
- Handelt es sich beim Erkennungsergebnis um eine gültige Eingabe? (Phonem- oder Wortebene)
- Handelt es sich um eine gültige Eingabe nach der Grammatik? (Satz oder Kommandophrase bekannt)

Für die Beantwortung all dieser Fragestellungen werden verschiedene Arten von Konfidenzmaßen genutzt. Sie geben mit einem Wert zwischen 0 und 1 (in den meisten Fällen) an, wie zuverlässig eine Spracherkennerentscheidung ist. Somit bilden Konfidenzmaße einen der wichtigsten Bestandteile der Spracherkennung, da mit ihrer Hilfe das Erkennungsergebnis validiert wird und eventuelle Fehler erkannt werden können. Diese Fehleingaben sollen dann zurückgewiesen werden.

In dieser Arbeit soll ein Konfidenzmaß für die Rückweisung ungrammatischer Spracherkennereingaben entwickelt und am Experimentiersystem Unified Approach to Speech Synthesis and Recognition (UASR) implementiert und evaluiert werden. Bei Versuchen am Lehrstuhl

Kommunikationstechnik fiel auf, dass teilweise Eingaben mit sehr unwahrscheinlichen Phonemlängen angenommen wurden. Diese Eingaben sind durch einen Menschen schnell als Fehlinterpretation durch den Erkenner zu identifizieren. In dieser Arbeit wird versucht, die Erkenerkonfidenz zur Rückweisung solcher Fehlannahmen zu verbessern. UASR ist ein gemeinsames Langzeitforschungsprojekt des Lehrstuhls für Kommunikationstechnik der Brandenburgischen Technischen Universität (BTU), des Lehrstuhls für Systemtheorie und Sprachtechnologie der Technischen Universität Dresden (TUD) sowie des Lehrstuhls für Informatik an der Universität Augsburg und weiteren Partnern und nutzt als Basis die von den gleichen Partnern entwickelte Software dLabPro.

Aufbau dieser Arbeit

Ziel dieser Arbeit ist es das Erkennungsergebnis des Spracherkenners im Experimentiersystem UASR zu verbessern indem ungrammatische Spracheingaben zurückgewiesen werden und somit eine weitere Falscherkennung durch „Aufzwingen “der Erkennergrammatik zu verhindern.

Kapitel 2 gibt einen kurze Einführung zum Thema Konfidenzmaße mit einem Überblick über die wichtigsten Bewertungsmöglichkeiten solcher Konfidenzen und einer Zusammenfassung des allgemeinen Standes der Technik.

In Kapitel 3 wird die zur Gewinnung von Testdaten durchgeführte Wizard-of-Oz-Studie erläutert und eine Beschreibung des daraus entstandenen Testkorpus gegeben.

Die theoretischen Aspekte der Konfidenzberechnung im UASR Experimentiersystem werden in Kapitel 4 erklärt. In diesem Kapitel folgt auch die Vorstellung der neu hinzugekommenen Merkmale.

Kapitel 5 beschreibt die Simulationen der Merkmale auf dem Rechencluster des Lehrstuhls Kommunikationstechnik und wertet die gesammelten Daten aus.

Die Arbeit endet mit einer Zusammenfassung der Ergebnisse und bietet einen Ausblick auf zukünftige Untersuchungen und Ansätze in Kapitel 6.

2 Konfidenzmasse: Bewertung und Stand der Technik

In diesem Kapitel finden sich grundlegende Informationen zu Konfidenzmaßen. Der erste Abschnitt definiert kurz den Begriff der Konfidenzmaße in der Spracherkennung. Im zweiten Abschnitt wird ein kurzer Überblick über die gängigen Methoden zur Konfidenzschätzung gegeben.

2.1 Konfidenzmaße

Konfidenzmaße geben an, wie zuverlässig eine Entscheidung getroffen werden kann oder wurde. Im Anwendungsfall unterscheidet man im allgemeinen zwischen kontinuierlichen und binären Konfidenzmaßen. Ein idealer Konfidenzschätzer wäre binär, und würde für eine korrekte Eingabe 1 annehmen und für eine falsche Eingabe 0. Da allerdings fast nie so scharf getrennt werden kann, erzeugen die meisten Konfidenzschätzer einen Wert aus dem Intervall $[0, 1]$ als A-posteriori-Wahrscheinlichkeit.

Ein solcher Konfidenzschätzer mit kontinuierlichem Wertebereich lässt sich durch Einführen eines Schwellwertes in einen binären Konfidenzschätzer umwandeln. Für die Rückweisung werden ausschließlich solche binären Konfidenzmaße genutzt, da hierbei auch eine binäre Entscheidung, Rückweisung oder keine Rückweisung, getroffen werden muss.

In der automatischen Spracherkennung werden Konfidenzmaße für verschieden Aufgaben genutzt. Sei es zur Sprecherverifikation oder zur inhaltlichen Zuordnung einer Spracheingabe. Diese Arbeit beschäftigt sich mit Konfidenzmaßen zur Rückweisung von Spracheingaben, die einer gegebenen Kommandogrammatik nicht entsprechen.

Bewertung von Konfidenzmaßen

Die Ergebnisse eines binären Konfidenzschätzers können wie in Tabelle 2.1 in vier Gruppen aufgeteilt werden. Hierbei bezeichnen die Kategorien *true positive (TP)* und *true negative (TN)* die Fälle in denen der Schätzer korrekt gearbeitet hat und die Kategorien *false positive (FP)*

und *false negative (FN)* die Fehlerfälle. Für die Bewertung eines Konfidenzschätzers sind vor allem die Fehlerfälle von Bedeutung. Hierbei unterscheidet man zwischen Fehlern vom Typ I (FP) bei denen eine korrekte Eingabe zurückgewiesen wird, und Fehlern vom Typ II (FN) bei denen eine falsche Eingabe angenommen wird.

Die Unterscheidung der beiden Fehlerfälle bietet die Möglichkeit das System auf spezielle Schwächen zu überprüfen oder auf eine bestimmte Anwendung zu optimieren. Hierfür können die verschiedenen Fehlerarten in der Bewertung und Konfiguration eines Schätzers verschieden stark gewichtet werden.

Im Folgenden werden verschiedene Maßnahmen zur Bewertung von Konfidenzschätzern vorgestellt. Zu beachten ist, dass es sich bei vielen der hier dargestellten Werte um *Quoten* und nicht wie in den Namen geschrieben *Raten* handelt. Die falsche Benennung rührt von der Übersetzung aus dem Englischen her und wird auch im deutschen genutzt, auch wenn die Bezeichnung *Rate* an dieser Stelle nicht mathematisch korrekt ist.

Konfidenzfehlerrate (KFR)

Die KFR beschreibt als Summe der beiden Fehlerkategorien die Fehlerrate des Gesamtsystems und bietet als solche auch eine gute Vergleichsmöglichkeit zwischen verschiedenen Schätzern (gleiche oder vergleichbare Stichproben vorausgesetzt). Als Maß für die Verbesserung eines Systems durch ein Merkmal wird oft die relative Verringerung der KFR genutzt.

Für bestimmte Anwendungen kann es sinnvoll sein, die verschiedenen Fehlerarten unterschiedlich zu gewichten. Hierfür wird der Gewichtungsfaktor $0 \leq \lambda \leq 1$ eingeführt. Mit $N = TP + TN + FP + FN$ als Gesamtanzahl der geschätzten Turns wird KFR berechnet wie Formel 2.1 (nach Berton, S. 78 [2]).

$$KFR = \frac{\lambda \cdot FP + (1 - \lambda) \cdot FN}{N} \tag{2.1}$$

		Schätzung	
		akzeptiert	verworfen
Evaluation	korrekt	true positive	false positive
	falsch	false negative	true negative

Tabelle 2.1: Verwechslungsmatrix für binäre Entscheider nach Berton [2]

Die Konfiguration des Konfidenzschätzers ist so vorzunehmen, dass die KFR minimal wird.

False-Acceptance-Rate (FAR) und False-Rejection-Rate (FRR)

Analog zur KFR ergeben sich FAR und FRR als Fehlerrate aller falsch angenommenen Eingaben im Verhältnis zu allen als falsch anzusehenden Eingaben bzw. aller falsch zurückgewiesenen Eingaben im Verhältnis zu allen als korrekt anzusehenden Eingabehn.

$$FAR = \frac{FP}{FP + TN} \tag{2.2}$$

$$FRR = \frac{FN}{TP + FN} \tag{2.3}$$

Equal-Error-Rate (EER)

Die EER beschreibt die Schätzerkonfiguration, bei der der Anteil von falschen Rückweisungen an allen Rückweisungen dem Anteil falscher Akzeptanzen an allen Akzeptanzen entspricht. Hierbei werden die beiden Fehlerarten getrennt betrachtet und angeglichen. Hierfür muss das System so eingestellt werden, dass Gleichung 2.4 erfüllt ist.

$$EER = \frac{FP}{FP + TP} = \frac{FN}{FN + TN} \tag{2.4}$$

In der Praxis findet die EER normalerweise eher wenig Anwendung, da häufiger der Fall mit den minimalen Fehlern (KFR) als der Fall mit ausgeglichenen Fehlern betrachtet wird [2].

Operationscharakteristiken

Mithilfe von Operationscharakteristiken lassen sich die Abhängigkeiten zwischen den Häufigkeiten der verschiedenen Fehlertypen und korrekten Zuordnungen gut darstellen. Somit eignen sie sich gut um Erkennersysteme unter bestimmten Vorgaben oder für bestimmte Anwendungen zu optimieren. So kann man mit ihnen zum Beispiel den Arbeitspunkt finden, an dem die korrekten Rückweisungen maximal sind unter der Voraussetzung, dass maximal 5 % der korrekten Eingaben zurückgewiesen werden.

Ein Beispiel ist der Detection-Error-Tradeoff (DET), wobei Typ-II-Fehler in Abhängigkeit von Typ-I-Fehlern aufgetragen werden. DET veranschaulicht, wie man einen Fehlertyp auf Kosten des anderen verringern kann. In Abbildung 2.1 ist beispielhaft der DET des

Auswertungsmerkmals NAD dargestellt. Mit mehr Messpunkten sollte sich ein deutlich geglätteter Verlauf einstellen. Erwünscht wäre ein DET, der sich hyperbelförmig dicht an den Achsen anschmiegt.

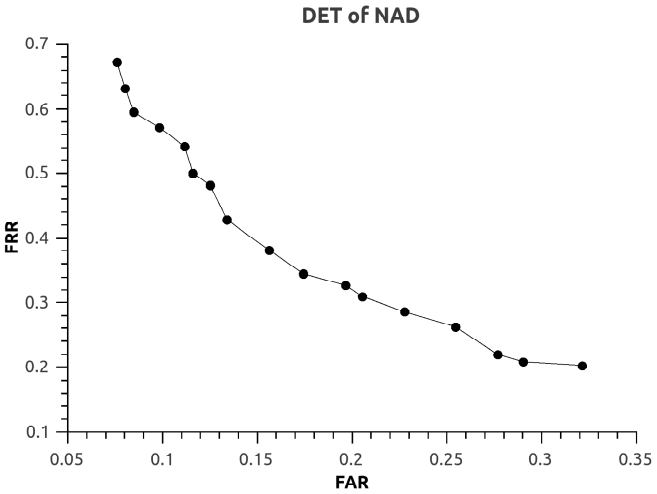


Abbildung 2.1: DET des Konfidenzmerkmals NAD

2.2 Stand der Technik

Zur Schätzung von Konfidenzmaßen haben sich in der Vergangenheit zwei grundlegende Konzepte etabliert [1] :

- Kombination von Bewertungsmerkmalen als Konfidenzmaß
- A-posteriori-Wahrscheinlichkeiten auf Wortgraphen als Konfidenzmaß

In dem kommenden drei Abschnitten sollen diese Konzepte kurz vorgestellt und mit einigen Beispielen eingeführt werden.

2.2.1 Kombination von Merkmalen

Als Grundlage für dieses Konzept dient in den meisten Fällen die Signalanalyse oder die Ergebnisse der Dekodierung, also N-best-Listen oder Wortgraphen. Diese Wortgraphen sind normalerweise gerichtet und zyklenfrei. Auf diesen Strukturen werden nacheinander

verschiedene Merkmale zur Konfidenzschätzung berechnet. Die Merkmale können dann entweder gleichberechtigt oder verschieden gewichtet kombiniert werden.

Hierbei ist darauf zu achten, dass die Korrelation der gewählten Merkmale minimal ist, da ansonsten Rechenaufwand ohne Zugewinn aufgebracht werden muss. Die übliche Vorgehensweise ist dabei, für jedes Merkmale einzeln die Verringerung der KFR zu ermittelt und dann die beste Kombination aus den bestgeeigneten Merkmalen zu suchen.

In der Vergangenheit wurde eine Vielzahl an solchen Merkmalen entwickelt. Im Folgenden wird eine kleine Auswahl vorgestellt.

SNR - Signal-Rausch-Abstand

Der Signal-Rausch-Abstand kann als Gütemaß für das Sprachsignal eine generelle Einschätzung über die Kanalqualität geben. Damit eignet sich der SNR um eine generelle Schätzung der Konfidenz einer Spracheingabe abzugeben. Ein großer Abstand zwischen Signal und Rauschen spricht für ein sehr sauberes Signal. Ein sauberes Signal eignet sich offensichtlich deutlich besser für die Spracherkennung als ein gestörtes Signal.

Von besonderem Interesse sollte dieses Merkmal sein, um die Signalqualität außerhalb des Labors bei Anwendungen in lauten Umgebungen (im Auto, in Maschinenhallen, etc.) zu schätzen. Berton [2] schlägt vor, den Geräuschpegel in den Sprachpausen zu ermitteln, um eine genauere SNR-Schätzung zu erhalten.

Hypothesendichte

Eine sichere Worterkennung sorgt dafür, dass eine Hypothese besonders dominant ist, da sie besonders sicher ist. Hypothesen mit geringer Wahrscheinlichkeit werden verworfen und somit ergibt sich für das sicher erkannte Zeitsegment nur eine oder eine geringe Anzahl an Worthypothesen. Gibt es dagegen eine große Unsicherheit über das benannte Zeitsegment, so wird eine Vielzahl an konkurrierenden Hypothesen entstehen.

In der Umkehrung kann daraus geschlossen werden, dass die Konfidenz für einen Zeitabschnitt bei steigender Dichte der konkurrierenden Hypothesen in diesem Zeitsegment sinkt. Es gilt also: je mehr Hypothesen zu einem Zeitpunkt, desto niedriger die Konfidenz.

Kemp und Schaaf [3] schlagen verschiedene Ansätze für die Wahl von Zeitpunkten vor. So berechnen sie die Hypothesendichte, zum Wortbeginn und Wortende, sowie den Durchschnitt

über die Wortdauer. Die Betrachtung der Hypothesendichte der Nachbarschaft, explizit am Ende des Vorgängerwortes und zu Beginn des Nachfolgewortes, brachte in ihren Versuchen weitere Verbesserungen.

Wortkorrektrate

Aus dem Spracherkennertraining können für alle Worte darüber Daten erhoben werden, wie oft diese korrekt geschätzt wurden. Bestimmte Worte werden häufig verwechselt. Ein Beispiel wären *fünfzig* und *vierzig* im UASR System. Auf einer Trainingsstichprobe können Wortkorrektraten für jedes Wort erhoben werden, sofern das Wort häufig genug in der Stichprobe vorkommt. Wenn ein Wort nicht oft genug vorkommt, wird die Korrektrate auf eine mittlere Wortkorrektrate approximiert [2]. Weiterhin kann eine Verwechslungsmatrix für die Worte aufgestellt werden und somit ein weiteres Merkmal erzeugt werden. Diese Daten sind allerdings stark anwendungsabhängig und erfordern ein statisches Vokabular.

Phondauerbasierte Merkmale

Mit einer ähnlichen Motivation wie für diese Arbeit wurden von Silke Goronzy verschiedenen phonemdauerbasierte Merkmale Entwickelt [4]. Die Basis für ihren Ansatz bildete die Sprachgeschwindigkeit α_c. Diese wird berechnet durch Formel 2.5 und normiert die Phonemdauerstatistiken für weitere Berechnungen.

$$\alpha_c = \frac{1}{N} \sum_{i=1}^{N} \frac{dur_i}{\overline{x}_p} \tag{2.5}$$

N steht hierbei für die Anzahl der Phoneme in der beobachteten Phrase, dur_i für die Dauer des i-ten Phonems in der Folge und $\overline{x}_p$ für die durchschnittliche Dauer des korrespondierenden Phonems. Die Phonemdauerstatistiken der einzelnen Phoneme sind aus dem Training des Erkenners bekannt.

Aufbauend auf der Sprachgeschwindigkeit und durch diese zum Teil normiert wurden verschiedenen Merkmale eingeführt, darunter verschiedene Merkmale über die Anzahl von Phonemen in einer Äußerung die nicht der Norm der Phonemdauerstatistik entsprechen. Ein Beispiel ist die Anzahl der Phoneme in einem Wort, die um mehr als das 5 % Perzentil im Vergleich zu den Trainingsdaten vom Durchschnitt abweichen. Es wurden mehrere Merkmale mit verschiedenen Perzentilen eingeführt. Weitere Merkmale beziehen sich auf die durchschnittliche Sprachgeschwindigkeit, bzw. die absolute Abweichung von dieser.

2.2.2 A-posteriori-Wahrscheinlichkeiten auf Wortgraphen

Dieses Verfahren wurde maßgeblich durch die Forschungsgruppe um Prof. Hermann Ney entwickelt und erforscht [5, 6]. Der Grundgedanke ist, dass die A-posteriori Wahrscheinlichkeit eines Wortes in einer Satzhypothese ein gutes Konfidenzmaß für die Spracherkennung ergeben würde.

Grundlage für das von Ney et al. entwickelte Verfahren sind gewichtete gerichtete azyklische Wortgraphen. Jeder Knoten im Wortgraphen steht für einen diskreten Zeitpunkt. Jede Kante beginnt in einem der Knoten und endet in einem anderen Knoten und repräsentiert eine Worthypothese und ist mit der akustischen Wahrscheinlichkeit dieser Hypothese gewichtet. Jeder Pfad durch den Graphen beginnend im ersten Knoten und endend im letzten Knoten steht für eine Satzhypothese. Der Graph beschreibt den unendlichen Suchraum in begrenztem Maße.

Gesucht wird nun die Wortfolge, die den optimalen Pfad, also den Pfad mit der höchsten A-posteriori Wahrscheinlichkeit, beschreibt. Hierfür müssen zunächst die A-posteriori Wahrscheinlichkeiten der einzelnen Worthypothesen berechnet werden. Dies geschieht in einer Art Forward-Backward Algorithmus auf dem Wortgraphen.

Um die Wahrscheinlichkeit eines Wortes zu berechnen werden seine *Geschichte*, also die direkten Vorgänger, und seine *Zukunft*, die Nachfolger, betrachtet. Um die Vorgänger zu berechnen werden beginnend am Startknoten des Graphen alle Pfade die im betrachteten Wort enden rekursiv über die Vorgänger der einzelnen Worte berechnet. Summiert man nun alle Wahrscheinlichkeiten dieser rekursiv berechneten Pfade auf, so erhält man die Forward-Wahrscheinlichkeit des betrachteten Wortes. Analog dazu werden rekursiv über die Zukunft der Worte alle Pfade beginnend im beobachteten Wort bis zum zeitlichen Ende des Wortgraphen berechnet. Summiert man diese Pfadwahrscheinlichkeiten auf so erhält man die Backward-Wahrscheinlichkeit des betrachteten Wortes.

Nun bildet man für jede mögliche Kombination aus Vorgängerfolge und Nachfolgerfolge das Produkt der Wahrscheinlichkeiten, also die Wahrscheinlichkeiten jedes möglichen Weges durch das betrachtete Wort. Schlussendlich werden alle diese Kombinationen aufsummiert und ergeben zusammen die Wahrscheinlichkeit der Worthypothese.

2.2.3 Konfidenzschätzung im UASR System

Die Konfidenzschätzung im UASR System beruht im wesentlichen auf dem Vergleich zweier Phonemerkennungen. Die eine liefert ein Erkennerergebnis unter Berücksichtigung der Erkennergrammatik, während die zweite eine freie Phonemerkennung ohne Grammatik als Referenz liefert. Da auf diesem System aufgebaut werden soll, erfolgt eine genauere Beschreibung der bisherigen Merkmale und ihrer Leistungsfähigkeit in Kapitel 4.

3 Testdaten

Das folgende Kapitel beschreibt wie mithilfe eines Wizard-of-Oz-Experimentes Sprachaufnahmen für einen Testkorpus aufgenommen wurden. Diese Aufnahmen dienten weiterhin zur Verfeinerung und Erweiterung der Kommandogrammatik für einen Aufbau im SpeechLab des Lehrstuhls Kommunikationstechnik.

3.1 Wizard-of-Oz-Experiment

Als Wizard-of-Oz-Experiment bezeichnet man ein Experiment bei dem ein Proband annimmt mit einer autonomen Maschine zu interagieren. In Wirklichkeit werden die Reaktionen des Systems allerdings von einem verborgenen menschlichen Versuchsleiter erzeugt. Der Begriff Wizard-of-Oz-Experiment wurde eingeführt und geprägt durch J. F. Kelly [7].

Vor allem im Feld der Mensch-Maschine-Kommunikation werden solche Experimente genutzt um Erkenntnisse über die Art und Form der Interaktion potentieller Benutzer mit einem System zu gewinnen. Hierbei kann zum Beispiel mit einer repräsentativen Anzahl von Probanden die Menge der benötigten Sprachphrasen für ein sprachgesteuertes System bestimmt werden.

3.1.1 Aufbau im SpeechLab

Der im SpeechLab des Lehrstuhls Kommunikationstechnik vorhanden Aufbau ist in Abbildung 3.1 dargestellt und besteht aus zwei Mikrofonfeldern. Jedes der Mikrofonfelder besteht aus 32 Mikrofonen. Ihre Anordnung ist in den Abbildungen 3.2(a) und 3.2(b) dargestellt. Neben jedem Mikrofon befindet sich eine LED, die in verschiedenen Farben den Status des korrespondierenden Mikrofons angibt. Mikrofonfeld 1 umrahmt einen Touchscreen, der zur weiteren Bedienung und Visualisierung des Aufbaus dient. Mikrofonfeld 2 befindet sich unter der Decke und ist auf einem fahrbaren Schlitten angebracht und lässt sich orthogonal zum Bildschirm vor- und zurückbewegen. Weiterhin befindet sich hinter den Mikrofonfeldern

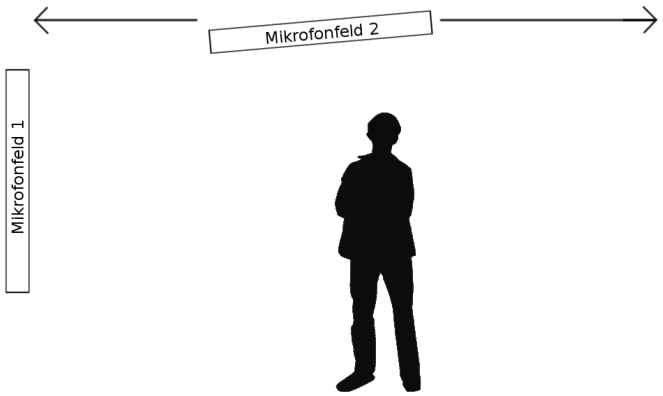


Abbildung 3.1: Mikrofonfelder im SpeechLab

eine Hintergrundbeleuchtung, mit deren Hilfe visuelles Feedback in Form von verschiedenen Farben und Blinkmustern an die Nutzer gegeben werden kann.

Mithilfe von Sprachkommandos und dem Touchscreen lässt sich der Versuchsaufbau bedienen. Hierbei können Mikrofone ein und ausgeschaltet werden und das obere Mikrofonfeld vor und zurückgefahren werden. Der Versuchsaufbau dient weiterhin der Forschung für Aufgaben wie zum Beispiel die Lokalisierung von Geräuschquellen im Raum und der Weiterentwicklung des UASR Systems.

3.1.2 Aufbau Wizard-of-Oz-Experiment

Für das Wizard-of-Oz-Experiment wurden die an den Mikrofonfeldern installierten Indikator-LEDs so eingestellt, dass sie blau leuchten wenn das entsprechende Mikrofon eingeschaltet ist und erlöschen, sobald das korrespondierende Mikrofon aus ist. Die Ausgabe von Bedienelementen auf dem Touchscreen wurde deaktiviert. Stattdessen wurde den Probanden auf dem Bildschirm die Mikrofonanordnung mit Nummerierung und Vorschläge für Mikrofongruppierungen angezeigt. Zusätzlich wurde neben jedem Mikrofon im Versuchsaufbau die Mikrofon-ID angebracht.

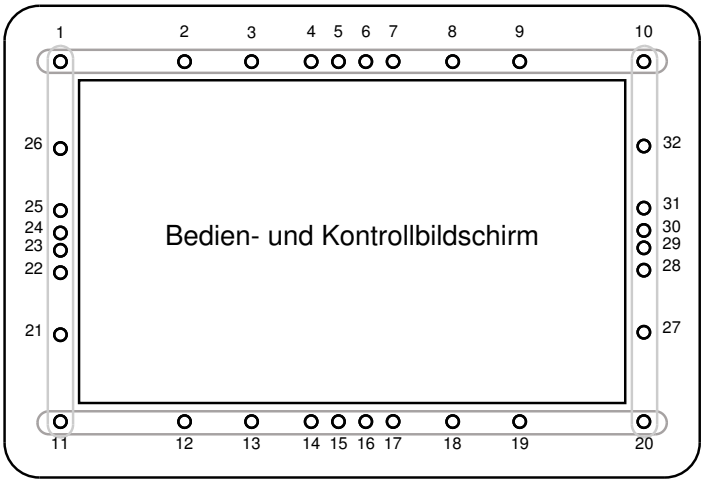
Um sowohl die Anonymität der Probanden zu gewährleisten, als auch ihre Einverständniserklärung zu erhalten musste jeder Proband zu Beginn des Experimentes folgenden Text einlesen: *Ich bin Proband *Nr.*, *Geschlecht*, *Alter* Jahre alt und damit einverstanden, dass meine Spracheingaben für lehrstuhlinterne Verwendungen aufgezeichnet werden.* Durch

Überprüfung der Stimmübereinstimmung kann so das Einverständnis für jede Aufnahme im Testkorpus belegt werden. Gleichzeitig ist die Anonymität aller Probanden gewährleistet.

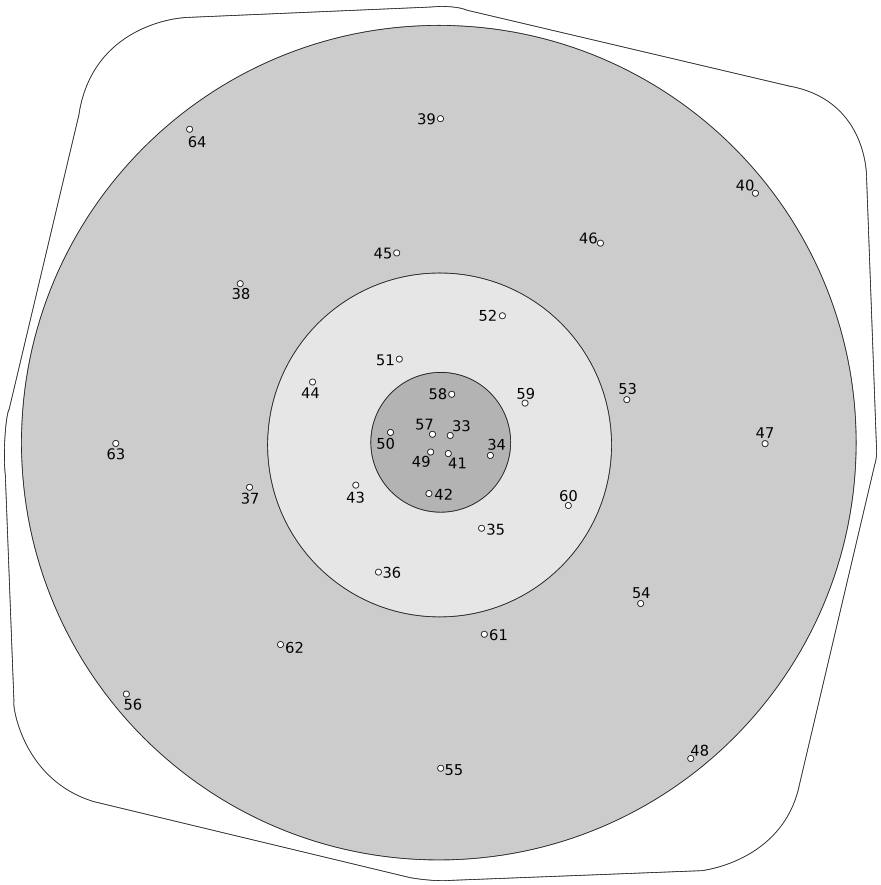
Um die Sprachkommandos der Probanden aufzunehmen wurde ein Headset genutzt. Dies garantierte gleichbleibende Aufnahmequalität egal wohin sich der Proband im Raum bewegt und gab weiterhin die Möglichkeit ein Talkback zu realisieren und durch die Kopfhörer Anweisungen an den Probanden zu geben. Alle Aufnahmen wurden entsprechend dem Datenformat des UASR Systems als mono wav-Dateien mit 16 kHz Abtastfrequenz und einer Bit-Tiefe von 16 Bit aufgezeichnet.

Das Backoffice des Versuchsleiters wurde in der im SpeechLab vorhandenen schallisolierten Sprecherkabine eingerichtet. Mit einem Kopfhörer und einem Mikrofon wurden Abhöhrfunktion und Talkback zum Probanden realisiert. Ein Kamerasignal mit Überblick über das SpeechLab und den Probanden wurden auf einen Bildschirm übertragen. Die Bedienung des Versuchsaufbaus erfolgte über ein von Prof. Wolff entwickeltes und leicht angepasstes Steuerinterface auf einem Touchscreen.

Um einen erfolgreichen Ablauf der Studie zu garantieren, wurde den Probanden vor Versuchstermin ein Handout mit einer Beschreibung ihrer Aufgabe und des Versuchsaufbaus bereitgestellt. Dieses Handout ist im Anhang A zu finden. Zusätzlich wurde mit jedem Probenden ein kurzes Vorgespräch zur Klärung von offenen Fragen geführt. Des weiteren sind in Anhang B alle von den Probanden an das System gerichteten Spracheingaben aufgelistet.



(a) Mikrofonanordnung in Mikrofonfeld 1



(b) Mikrofonanordnung in Mikrofonfeld 2

Abbildung 3.2: Mikrofonanordnung im Aufbau des SpeechLab

3.2 Testdaten

Mit dem in Abschnitt 3.1 beschriebenen Experiment konnten mithilfe von drei weiblichen und acht männlichen Probanden insgesamt 730 Sprachturns aufgenommen werden. Zusätzlich wurden 55 nichtsprachliche Turns mit Geräuschen aus der Laborumgebung, wie zum Beispiel das Öffnen der Tür oder das Summen des Motors von Mikrofonfeld 2, aufgenommen. Zusammen bilden diese Aufnahmen den dieser Arbeit zugrunde liegenden Testkorpus.

Mithilfe der Sprachaufnahmen wurde weiterhin die vorhandene Steuergrammatik für den in Abschnitt 3.1.1 beschriebenen Aufbau erweitert. Es bleibt zu erwähnen, dass nicht alle von den Probanden *vorgeschlagenen* Änderungen aufgenommen werden konnten. Mit der Erweiterung gibt es im Testkorpus 335 Turns, die der Grammatik entsprechen und 395 Turns, die der Grammatik nicht entsprechen. Der Gesamtkorpus wurde in eine 392 Turns umfassende Teststichprobe und eine 393 Turns umfassende Entwicklungsstichprobe aufgeteilt. Die Aufteilung erfolgte zufällig, wobei gewährleistet wurde, dass beide Stichproben etwa gleiche Anzahlen von korrekten, falschen und nichtsprachlichen Turns enthalten.

4 Theoretische Grundlagen

Dieses Kapitel behandelt die theoretischen Grundlagen der Konfidenzschätzung im UASR System. Die ersten beiden Abschnitte geben einen kurzen Überblick über das mathematische Konzept Hidden Markov Modell (HMM) und die Berechnung von Konfidenzintervallen der Evaluationsergebnisse. Im dritten Abschnitt wird ein Überblick über die Beschaffenheit der generellen Konfidenzschätzung in UASR gegeben und das Baselinesystem als Ausgangspunkt dieser Arbeit beschrieben. Der letzte Abschnitt beschäftigt sich mit den neu hinzugefügten Merkmalen zur Konfidenzschätzung.

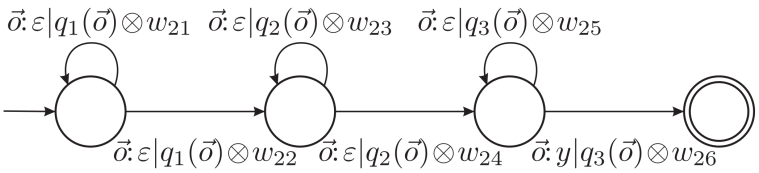
4.1 Hidden Markov Modelle

In der automatischen Spracherkennung haben sich Hidden Markov Modell als Standardstruktur zur Modellierung eines Suchraumes für die Erkennung etabliert. Ein HMM stellt ein System aus zwei gekoppelten Zufallsprozessen dar, von denen einer versteckt (engl. hidden) ist. Der versteckte Prozess steuert den zweiten Zufallsprozess, welcher zu jedem Zeitpunkt aus einer zustandsabhängigen Wahrscheinlichkeitsverteilung eine Emission erzeugt [8].

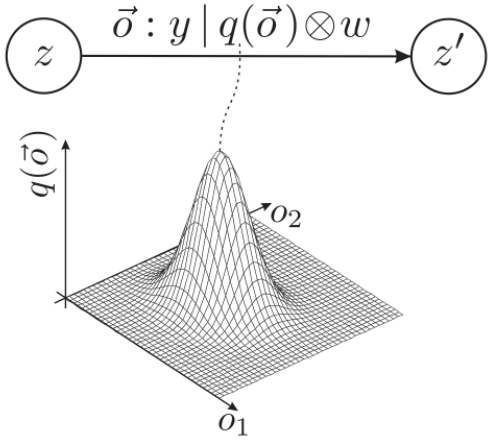
Für die automatische Spracherkennung wird ein Eingabesignal in Zeitabschnitte von 10 ms Länge gerastert. Zu jedem diesem Intervall wird ein Merkmalsvektor erzeugt. Diese Merkmalsvektoren werden dann als die Emissionen eines HMM interpretiert. Verschiedene Algorithmen (zum Beispiel Forward-Backward-Algorithmus) können dann genutzt werden um Phonem- oder Wortfolge der Eingabe zu berechnen.

Ein HMM $\mathcal{H}$ kann als Automat aufgefasst werden und ist dann eindeutig durch seinen

Automatengraphen definert [9, 10]:



Mit jedem Übergang wird ein Merkmalsvektor $\vec{o}$ akzeptiert (gelesen) und ein Ausgabezeichen y generiert (geschrieben). Hierbei kann das Ausgabezeichen auch leer, gekennzeichnet durch ε sein. Zusätzlich wird jedem Übergang eine Verteilungsdichtefunktion q und eine Übergangswahrscheinlichkeit w zugeordnet.



Ein HMM übersetzt also *beobachtete* Merkmalsvektorfolgen $\vec{\mathbf{o}} = (\vec{o}^1, \vec{o}^2, \ldots, \vec{o}^K)$ der Länge K in Wort- oder Phonemfolgen $\mathbf{y} = (y^1, y^2, \ldots)$ und ordnet der Übersetzung eine Wahrscheinlichkeit zu:

$$\mathcal{H}(\vec{\mathbf{o}}, \mathbf{y}) = \bigoplus_{U \in \mathcal{P}(I, \mathbf{y}, F)} \left[\bigotimes_{e^k \in U} \left[q_{e^k}(\vec{o}^k) \otimes w_{e^k} \right] \right] .$$

4.2 Konfidenzintervall von Detektionsergebnissen

Dieser Abschnitt beschäftigt sich mit der Berechnung des Vertrauensbereichs oder Konfidenzintervalls der in der Evaluation erzeugten Fehlerraten. Die Fehlerraten des Spracherkenners sind Bernoulli-verteilte Zufallsgrößen X. Das bedeutet X kann entweder den Wert 0 (kein Fehlerfall) oder 1 (Fehlerfall) annehmen. Für den Erwartungswert von X gilt dann:

$$m = E(X) = \frac{C}{N} \,, \tag{4.1}$$

wobei N die Stichprobengröße und $0 \leq C \leq N$ die Anzahl der falsch erkannten Werte ist. Die Standardabweichung der Stichprobe ist:

$$s = \sqrt{Var(X)} = \sqrt{\frac{N}{N-1}(E(X^2) - E(X)^2)} \,. \tag{4.2}$$

Da X nur die Werte 0 und 1 annehmen kann, gilt weiterhin:

$$E(X^2) = E(X) = m \,, \tag{4.3}$$

und damit:

$$s = \sqrt{\frac{N}{N-1}(m - m^2)} \,. \tag{4.4}$$

Das Konfidenzintervall für Bernoulli-verteilte Zufallsgrößen wird exakt berechnet mithilfe des Clopper-Pearson-Intervalls. Üblicherweise wird das Konfidenzintervall unter der falschen Annahme der Normalverteilung approximiert. Hierbei gilt mit der Umkehrung der Standardnormalverteilungsfunktion $\Phi^{-1}(x)$ laut Rinne [11]:

$$c = \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \,, \tag{4.5}$$

mit

$$c_\gamma \approx \pm c\frac{s}{\sqrt{N}} \,. \tag{4.6}$$

Mit $\alpha = 1 - \gamma$ ergibt sich für ein 95-Konfidenzintervall also:

$$c_{95} \approx \Phi^{-1}\left(1 - \frac{0,05}{2}\right)\frac{s}{\sqrt{N}} = \pm 2\sqrt{\frac{m - m^2}{N-1}} \,. \tag{4.7}$$

Zu beachten ist, dass diese Annäherung nur für Stichproben mit $N > 50$ verwendet werden sollte.

4.3 Erkennerkonfidenz in UASR

Im Erkennersystem von UASR werden zwei Erkenner parallel bearbeitet. Der erste ist ein Erkenner, der ein Ergebnis entsprechend der Erkennergrammatik generiert. Der zweite ist eine freie Phonemerkennung, die lediglich eine Folge von Phonemen ohne Grammatik erkennt. Mit der Automatendarstellung der HMM lassen sich diese Erkenner wie folgt beschreiben:

$$\mathcal{R}_{rec} = \left(\bigoplus_m \mathcal{H}_m \right)^* \circ \left(\bigoplus_l \mathcal{L}_l \right)^* \circ \mathcal{G} \,, \tag{4.8}$$

$$\mathcal{R}_{ref} = \left(\bigoplus_m \mathcal{H}_m \right)^* \,. \tag{4.9}$$

Wobei $\mathcal{H}$ ein HMM beschreibt, welches aus Merkmalvektoren der Eingabe eine Phonemfolge erzeugt. $\mathcal{L}$ ist das Wörterbuchmodell, welches aus einer Phonemfolge eine Phonemfolge die ein Wort repräsentiert erzeugt. $\mathcal{G}$ ist schließlich das Modell der Erkennergrammatik und erzeugt aus einer eine Wortfolge repräsentierenden Phonemfolge eine Phonemfolge, die eine Äusserung in der Grammatik repräsentiert. Die allgemeine Summation $\oplus$ beschreibt, dass aus allen vorhandenen Modellen gewählt wird und der Kleene-Stern $*$ steht für die beliebige Wiederholung eines Modelltypes, sodass also Folgen von Phonemen oder Worten gebildet werden können. Abbildung 4.1 zeigt beispielhafte Ergebnisse der beiden Erkenner pro Frame.

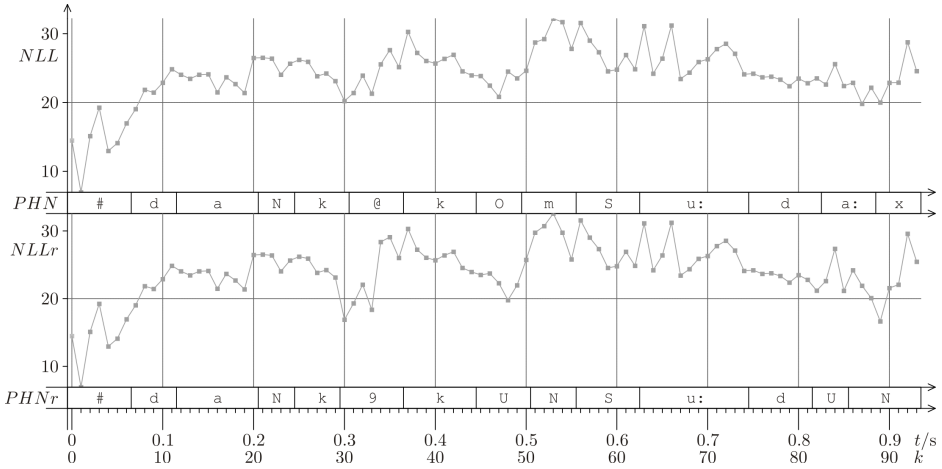


Abbildung 4.1: Erkennungsergebnisse pro Merkmalsvektor für die Eingabe *Danke, Computer.* [12] Die negativ logarithmischen Likelihoods (NLL und NLLr) geben dabei das Ergebnis der akustischen Auswertung an. Je niedriger der Wert ist, desto besser passt der Merkmalvektor zum gewählten Lautmodell (HMM).

4.4 Merkmale und Konfidenz

In diesem Abschnitt werden die 6 Merkmale beschrieben, mit deren Hilfe die Konfidenzmaße im UASR System geschätzt werden. Bisher waren nur die ersten beiden Merkmale vorhanden und implementiert und bilden somit das Baselinesystem für diese Arbeit. Vor der Berechnung der Merkmale werden alle Frames die von einem der beiden Erkennersysteme als Pause /./ oder Müll /#/ gekennzeichnet sind entfernt.

4.4.1 NAD

Das Merkmal Normalized Acoustic Distance (NAD) berechnet die akustische Distanz. Hierfür wird pro Merkmalsvektor die Differenz der akustischen Auswertungen (NLL und NLLr) gebildet und über die Gesamtlänge K in Merkmalsvektoren aufsummiert und normiert:

$$NAD = \frac{\sum_{k=1}^{K} \left[NLL(k) - NLLr(k)\right]}{\sum_{k=1}^{K} NLL(k)} \ . \tag{4.10}$$

4.4.2 NHD

Das Merkmal Normalized Hamming Distance (NHD) bildet über die Gesamtlänge K einer Eingabe die frameweise Hammingdistanz zwischen den Ergebnissen. Durch Aufsummieren der frameweisen Distanzen und Normieren mit der Gesamtlänge entsteht der durchschnittliche Hamming-Abstand der Ergebnisse:

$$NHD = \frac{1}{K} \sum_{k=1}^{K} d_H \left[PHN(k), PHNr(k)\right] \ , \tag{4.11}$$

mit der Hamming-Distanz zwischen zwei Phonemen:

$$d_H(x, y) = \begin{cases} 0, \text{falls } x = y \\ 1, \text{falls } x \neq y \end{cases} \ . \tag{4.12}$$

4.4.3 NWHD

Das gewichtete Merkmal Normalized weighted Hamming Divergence (NWHD) nutzt die aus dem Erkennertraining bekannten absoluten Phonemverwechslungshäufigkeiten $H(x_{rec}, x_{lab})$. x_{rec} bezeichnet das erkannte Phonem und x_{lab} die korrekte Annotation der Testdaten. Mit

dem leeren Zeichen ε werden die Häufigkeiten von Auslassungen $H(\varepsilon, x_{lab})$ des Phonems x_{lab} und Einfügungen $H(x_{rec}, \varepsilon)$ des Phonems x_{rec} dargestellt. Mithilfe dieser Daten können Phonemverwechslungswahrscheinlichkeiten geschätzt werden:

$$P(x_{rec}|x_{lab}) \approx \frac{H(x_{rec}, x_{lab}) + 1}{\sum_x H(x, x_{lab}) + N} , \tag{4.13}$$

wobei N die Anzahl aller Phoneme im Phonemalphabet ist. In der Schätzgleichung ist eine *add-one-Glättung* enthalten um Nullwerte zu vermeiden [13].

Unter der Annahme der unabhängigen Verwechslung aufeinanderfolgender Phonempaare kann die Verwechslungswahrscheinlichkeit der Folgen $\mathbf{x}_{rec} = (x_{rec}^{(1)} \dots x_{rec}^{(K)})$ und $\mathbf{x}_{ref} = (x_{ref}^{(1)} \dots x_{ref}^{(K)})$, mit den Formeln 4.15 und 4.16 als Produkt der Verwechslungswahrscheinlichkeiten angegeben werden:

$$P(\mathbf{x}_{ref}|\mathbf{x}_{rec}) = \prod_{k=1}^{K} P(x_{ref}^{(k)}|x_{rec}^{(k)}) . \tag{4.14}$$

Hierbei übernimmt die mit Grammatik erkannte Folge x_{rec} die Rolle der Annotation, die ohne Grammatik erkannte Folge x_{ref} die Rolle des Erkennungsergebnisses.

$$\mathbf{x}_{rec} = \arg \operatorname*{ext}_{\mathbf{x} \in X^*} [\![\mathcal{R}_{rec}]\!](\vec{\mathbf{o}}, \mathbf{x}) , \tag{4.15}$$

$$\mathbf{x}_{ref} = \arg \operatorname*{ext}_{\mathbf{x} \in X^*} [\![\mathcal{R}_{ref}]\!](\vec{\mathbf{o}}, \mathbf{x}) . \tag{4.16}$$

Mit der Bildung des geometrischen Mittels und Logarithmierung berechnet sich NWHD dann mit Formel 4.13 wie folgt:

$$NWHD = -\frac{1}{K} \sum_{k=1}^{K} \ln P(PHNr(k)|PHN(k)) . \tag{4.17}$$

4.4.4 NWLD

Für die Berechnung des Merkmals Normalized weighted Levenshtein Divergence (NWLD) werden die beiden erkannten Phonemfolgen zunächst mithilfe des Levenshtein-Distanz-Algorithmus so ausgerichtet, dass jedes Phonem aus der Erkennung mit Grammatik genau ein korrespondierendes Phonem aus der freien Phonemerkennung hat. Die neu ausgerichteten Phonemfolgen werden in PHN' und $PHNr'$ geschrieben und Auslassungen und Einfügungen

werden durch Zuordnung des leeren Symbols ε dargestellt. Die Länge der neuen Phonemfolgen ist K' in Phonemen. Ein Beispiel für die Ausrichtung der Phonemfolgen ist in den Abbildungen 4.2 und 4.3 dargestellt.

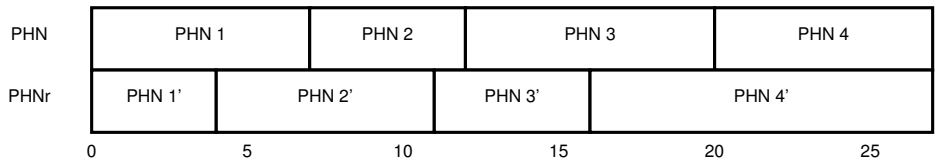


Abbildung 4.2: Erkannte Phonemfolgen vor der Ausrichtung

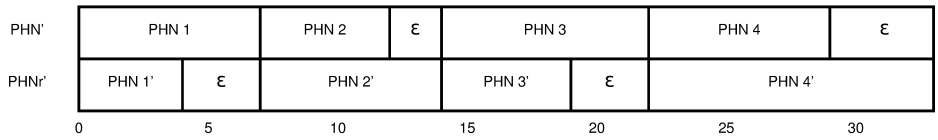


Abbildung 4.3: Neue Phonemfolgen nach der Ausrichtung.

Nun wird mithilfe der Phonemverwechselungswahrscheinlichkeiten aus Formel 4.13 NWLD berechnet:

$$NWLD = -\frac{1}{K'} \sum_{k=1}^{K'} \ln P(PHNr'(k)|PHN'(k)) \tag{4.18}$$

4.4.5 PDL

Als Basis für das Merkmal Phone Duration Likelihood (PDL) dienen die Phonemdauerhistogramme aus Anhang C. Sie wurden anhand der erweiterten deutschen Trainingsstichprobe der Verbmobil Datenbasis (VMX) ermittelt. Mithilfe dieser Histogramme $H_x(l)$ können die Wahrscheinlichkeitsverteilungen der phonemabhängigen Dauern geschätzt werden:

$$P(l|x) \approx \frac{H_x(l)}{\sum_l H_x(l)} \tag{4.19}$$

wobei $H_x(l)$ die absolute Häufigkeit darstellt, mit der die Lernstichprobe das Phonem x mit einer Dauer von l Merkmalsvektoren enthält. Nach entfernung der Pausen /./ und Müll /#/ Segmente kann PDL nun wie folgt aus der neuen Phonemfolge PHN' der Länge K' und

Formel 4.19 berechnet werden:

$$PDL = -\frac{1}{K'} \sum_{k=1}^{K'} \ln P(l(k)|PHN'(k)) \tag{4.20}$$

wobei $l(k)$ die Länge des k-ten Phonems in der Folge bezeichnet.

4.4.6 PDLW

Das Merkmal Weighted Phone Duration Likelihood (PDLW) berechnet sich wie auch PDL anhand der Phonemdauerstatistiken. Zusätzlich werden die einzelnen Phonemwahrscheinlichkeiten noch mit der Länge des Phonems in der Eingabefolge gewichtet.

$$PDLW = -\frac{\sum_{k=1}^{K'} l(k) \cdot \ln P(l(k)|PHN'(k))}{\sum_{k=1}^{K'} l(k)} \tag{4.21}$$

4.4.7 Konfidenzberechnung

Die einzelnen Merkmale werden vor ihrer Weiterverwendung standardisiert um die Kalibrierung des Systems anhand der Histogramme zu vereinfachen. Die Standardisierung eines Merkmals $scor_i$ erfolgt nach Formel 4.22, wobei m_i der empirische Mittelwert und s_i die Standardabweichung sind. op_i ist eine merkmalsabhängige Funktion wie Negation oder Absolutwertbildung. Die standardisierten Histogramme der Merkmale sind in Anhang D abgebildet.

$$scor_{i,S} = \frac{op_i(scor_i - m_i)}{s_i} \tag{4.22}$$

Das Konfidenzmaß im UASR System wird schlussendlich als Kombination der einzelnen Merkmale gebildet:

$$conf = \sum_i \lambda_i \cdot \min\left[\max(scor_{i,S} - t_i, -1), 1\right] \tag{4.23}$$

Hierbei beschreibt t_i den Schwellwert und λ_i das Gewicht des Merkmals i. Die Summe aller Gewichte λ_i muss dabei 1 betragen. Der berechnete Konfidenzwert liegt in einem Intervall $[-1, 1]$ wobei $conf = -1$ für eine sichere Rückweisung und $conf = 1$ für eine sichere Akzeptanz steht.

Die bisherige Konfidenzschätzung im UASR System beschränkte sich auf die Merkmale NAD und NHD, welche zu gleichen Teilen, also mit $\lambda_{NAD} = \lambda_{NHD} = 0,5$ in die Schätzung eingingen. Dieses Konfidenzmaß wurde im Zuge der Arbeit evaluiert und optimiert (Kapitel

5). Die vier weiteren Merkmale wurden für diese Arbeit in Zusammenarbeit mit Prof. Wolff ausgewählt und von ihm im Experimentiersystem UASR implementiert.

Zur Auswertung unterscheidet das Experimentiersystem die als TP erkannten Eingaben in zwei Gruppen. Zum einen die Eingaben, die angenommen wurden, angenommen werden sollten und entsprechend dem Referenzlabel erkannt wurden und zum anderen, die Eingaben, die angenommen wurden, angenommen werden sollten aber deren Erkennungsergebnis nicht mit dem Referenzlabel übereinstimmt. Für den zweiten Fall muss entschieden werden, ob diese Gruppe als korrekt erkannt, oder falsch erkannt angesehen werden soll. Hierfür stellt das System zwei Modi bereit. Zum einen den Modus *OOT* in dem diese falsch erkannten aber anzunehmenden als korrekt erkannt angesehen werden und den Modus *ERR*, bei dem diese als falsch erkannt gewertet werden.

5 Evaluation

Dieses Kapitel beschäftigt sich mit der Evaluation der in Kapitel 4 beschrieben sechs Konfidenzmerkmale. Der erste Abschnitt behandelt die Kalibrierung des Erkennersystems. Im zweiten Abschnitt werden die erhobenen Daten analysiert und Schlüsse über die Qualität der einzelnen Merkmale gezogen.

Auf der dieser Arbeit beigelegten DVD befindet sich der verwendete Testkorpus, sowie die verwendete Software. Eine Beschreibung zur Reproduktion der Evaluationsergebnisse und weiterverwendung der Software findet sich in Anhang F.

5.1 Kalibrierung

Zu Beginn der Evaluationsphase wurden die Standardisierungsparameter der einzelnen Merkmale so bestimmt, dass alle Merkmale in ihrer standardisierten Form einen Mittelwert von etwa 0 und eine Standardabweichung von 1 haben. Durch die Standardisierung der Merkmale ist eine einfache Lesbarkeit der Histogramme gewährleistet, sodass die Suche nach geeigneten Schwellwerten einfacher begonnen werden konnte. Ausserdem erhalten alle Merkmale einen nahezu einheitliches Intervall an möglichen Ausgabewerten.

Tabelle 5.1 zeigt die Kalibrierungsdaten Mittelwert m_i und Standardabweichung s_i der einzelnen Merkmale wie in Formel 4.22.

Merkmal	m_i	s_i
NAD	0,0612	0,0260
NHD	0,4850	0,1550
NWHD	2,5900	0,7500
NWLD	1,2275	0,2100
PDL	2,8800	0,3650
PDLW	2,9000	-0,7500

Tabelle 5.1: Kalibrierungswerte der Merkmalsstandardisierung

Mit dem so eingestellten System wurde dann die Leistungsfähigkeit der Einzelmerkmale geprüft. Hierfür wurden mit der Entwicklungsstichprobe eine Grid-Search nach dem optimalen Schwellwert der einzelnen Merkmale gesucht. Es wurden jeweils Schwellwerte mit der geringsten KFR also der geringsten Gesamtfehlerquote gesucht. Es wäre auch möglich nach zum Beispiel der geringsten Quote der falsch akzeptierten Eingaben (FAR) zu suchen. Da der DET keines der Merkmale eine gute Charakteristik aufwies, wurde sich auf die Optimierung nach Gesamtfehlerquote beschränkt.

In Tabelle 5.2 sind die optimalen Schwellwerte der Einzelmerkmale und die daraus resultierenden Fehlerraten mit entsprechendem Konfidenzniveau angegeben. Die KFR der Merkmale ist in Abbildung 5.1 dargestellt. Offensichtlich ist, dass NAD das stärkste Merkmal ist und dass PDL sehr schlechte Werte liefert. Aufgrund der schlechten Performance wurde PDL in den weiteren Versuchen nicht weiter beachtet.

Merkmal	Schwellwert	KFR	FAR	FRR
NAD	0,36	$0{,}2423 \pm 0{,}0433$	$0{,}1786 \pm 0{,}0513$	$0{,}3274 \pm 0{,}0726$
NHD	0,28	$0{,}2908 \pm 0{,}0459$	$0{,}2188 \pm 0{,}0554$	$0{,}3869 \pm 0{,}0754$
NWHD	0,24	$0{,}2934 \pm 0{,}0461$	$0{,}2589 \pm 0{,}0587$	$0{,}3393 \pm 0{,}0733$
NWLD	0,19	$0{,}3597 \pm 0{,}0485$	$0{,}3125 \pm 0{,}0621$	$0{,}4226 \pm 0{,}0765$
PDL	0,80	$0{,}4541 \pm 0{,}0504$	$0{,}1473 \pm 0{,}0475$	$0{,}8631 \pm 0{,}0532$
PDLW	-0,40	$0{,}3827 \pm 0{,}0492$	$0{,}1875 \pm 0{,}0523$	$0{,}6429 \pm 0{,}0742$

Tabelle 5.2: Fehlerraten mit Abweichung des Vertrauensbereichs bei Konfidenzniveau von 95%. Die Konfidenzfehlerrate wurde hierbei mit $\lambda = 0,5$ *nicht* zugunsten eines Fehlertyps gewichtet.

Baselinesystem

Wie zuvor in Abschnitt 4.4.7 beschrieben wurde die Konfidenz für Rückweisung im UASR System bisher nur mit den Merkmalen NAD und NHD mit gleichen Gewichten durchgeführt. Mit der Teststichprobe konnte hierfür eine Konfidenzfehlerrate von $27,37 \pm 4,52\%$ erreicht werden. Durch eine Grid-Search über verschiedene Gewichtungen der beiden Merkmale und verschiedene aus den Einzeltests als vielversprechend hervorgegangene Schwellwerte stellte sich heraus, dass sich die KFR auf $24,81 \pm 4,37\%$ verringern lässt. Für diese Verbesserung wird nur das Merkmal NAD mit einem Gewicht von $\lambda_{NAD} = 1$ genutzt. Es kann leider nicht von einer signifikanten Verbesserung durch diese Optimierung ausgegangen werden, da die Werte im Konfidenzintervall des jeweils anderen Wertes liegen.

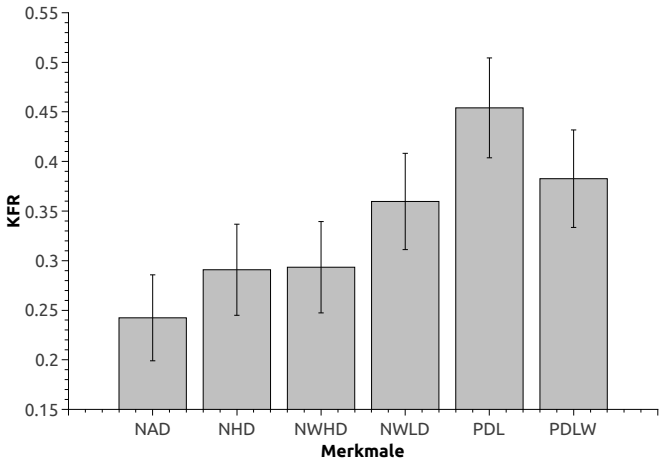


Abbildung 5.1: Konfidenzfehlerraten der einzelnen Merkmale (siehe Gleichungen 4.10, 4.11, 4.17, 4.18, 4.20, 4.21). Die beiden Fehlertypen sind gleich gewichtet ($\lambda = 0,5$).

5.2 Merkmalskombinationen

Im weiteren Verlauf wurden verschiedene Merkmalspaare getestet. Jedes Merkmal wurde mit der Entwicklungsstichprobe und einer Grid-Search über verschiedene Gewichtungen und Schwellwertbereiche mit dem stärksten Merkmal NAD als Paar geprüft. Die besten Ergebnisse wurden mit der Teststichprobe geprüft. Getestet wurden die Baselinekonfiguration, eine optimierte Baselinekonfiguration die nur mit NAD berechnet wird und die Paare NAD+NWLD und NAD+PDLW. Die weiteren Paare NAD+NWHD und NAD+PDL wurden aufgrund ihrer schlechten Performance nicht weiter evaluiert.

Die Konfigurationen haben folgende Parameter:

Baseline

Merkmale	λ_{NAD}	λ_{NHD}	t_{NAD}	t_{NHD}
NAD + NHD	0,5	0,5	0,36	0,28

Baseline optimiert

Merkmale	λ_{NAD}	t_{NAD}
NAD	1,0	0,36

NWLD + NAD

Merkmale	λ_{NAD}	λ_{NWLD}	t_{NAD}	t_{NWLD}
NAD + NWLD	0,75	0,25	0,37	0,19

PDLW + NAD

Merkmale	λ_{NAD}	λ_{PDLW}	t_{NAD}	t_{PDLW}
NAD + PDLW	0,85	0,15	0,35	−0,5

Alle vier Konfigurationen wurden mit dem Testset in beiden Bewertungsmodi *OOT* und *ERR* getestet. Die Ergebnisse sind in den Tabellen 5.3 und 5.4 sowie den Abbildungen 5.2 und 5.3 dargestellt.

Monfiguration	KFR	FAR	FRR
Baseline	$0{,}2737 \pm 0{,}0452$	$0{,}1920 \pm 0{,}0527$	$0{,}3832 \pm 0{,}0755$
Baseline opt	$0{,}2481 \pm 0{,}0437$	$0{,}1563 \pm 0{,}0486$	$0{,}3713 \pm 0{,}0750$
NAD + NWLD	$0{,}2404 \pm 0{,}0433$	$0{,}1607 \pm 0{,}0492$	$0{,}3473 \pm 0{,}0739$
NAD + PDLW	$0{,}2532 \pm 0{,}0440$	$0{,}1429 \pm 0{,}0469$	$0{,}4012 \pm 0{,}0761$

Tabelle 5.3: Fehlerraten mit Abweichung des Vertrauensbereichs bei Konfidenzniveau von 95% für die vier ausgewählten Konfigurationen im Modus *OOT*

Monfiguration	KFR	FAR	FRR
Baseline	$0{,}2916 \pm 0{,}0460$	$0{,}2960 \pm 0{,}0510$	$0{,}2714 \pm 0{,}1071$
Baseline opt	$0{,}2609 \pm 0{,}0445$	$0{,}2679 \pm 0{,}0495$	$0{,}2286 \pm 0{,}1011$
NAD + NWLD	$0{,}2430 \pm 0{,}0434$	$0{,}2648 \pm 0{,}0493$	$0{,}1429 \pm 0{,}0843$
NAD + PDLW	$0{,}2506 \pm 0{,}0439$	$0{,}2492 \pm 0{,}0484$	$0{,}2571 \pm 0{,}1052$

Tabelle 5.4: Fehlerraten mit Abweichung des Vertrauensbereichs bei Konfidenzniveau von 95 % für die vier ausgewählten Konfigurationen im Modus *ERR*

Leider konnte sich keine der ausgewählten Konfigurationen klar vom bisherigen Baselinesystem trennen. Für fast alle Konfigurationen liegt der Messwert im Konfidenzintervall des Baselinesystems. Lediglich im Modus *ERR* lässt die Konfiguration NAD+PDLW eine Tendenz zur Verbesserung erkennen. Hier liegt der Messwert nur knapp innerhalb des Vertrauensbereiches der Baselineschätzung.

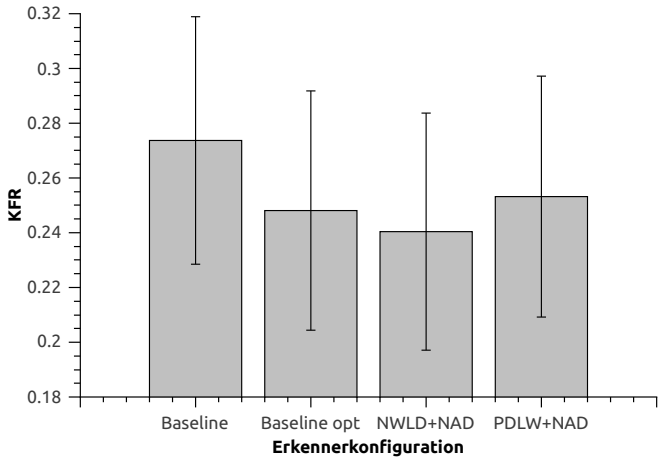


Abbildung 5.2: Konfidenzfehlerraten der vier Konfigurationen im Modus *OOT*

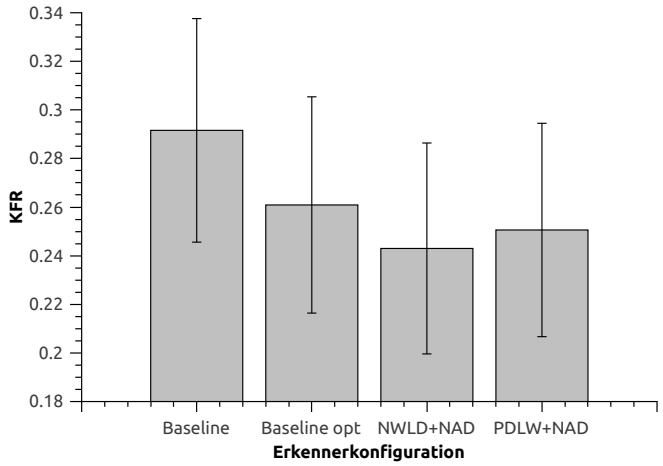


Abbildung 5.3: Konfidenzfehlerraten der vier Konfigurationen im Modus *ERR*

6 Zusammenfassung und Ausblick

Im Zuge dieser Arbeit wurde erklärt was Konfidenzmaße für die automatische Spracherkennung bedeuten. Es wurde zusammengefasst, wie Konfidenzmaße zu bewerten sind und wie der aktuelle Forschungsstand ist. In einer Wizard-of-Oz-Studie wurde ein Testkorpus für das UASR System erhoben und die Kommandogrammatik für den Mikrofonfeldaufbau im SpeechLab erweitert. Für die Rückweisung wurden neue Konfidenzmerkmale zum Experimentiersystem UASR hinzugefügt und getestet. Leider konnte sich keine der getesteten Konfigurationen klar vom Baselinesystem abheben und für eine eindeutige Verbesserung sorgen. In einigen Vorversuchen mit der alten Grammatik und einer deutlich kleineren Teststichprobe zeigten sich einige Merkmale als sehr vielversprechend. Dass diese Aussichten nicht bestätigt werden konnten kann mit der erweiterten Komplexität der neuen Grammatik erklärt werden. in der Ausarbeitung der Grammatik liegt noch viel Verbesserungspotential um diese eindeutiger zu machen. Die Ergebnisse der Vorversuche sind weiterhin durch die deutlich zu kleine Stichprobe verzerrt.

Ausblick

Um die Rückweisungskonfidenz im Experimentiersystem UASR zu verbessern sollten entweder weitere Tests mit verschiedenen Merkmalskonfigurationen und -kombinationen evaluiert werden und neue Konfidenzmaße auf anderer Basis entwickelt werden. Da als eine der Motivationen für die Entwicklung neuer Konfidenzmaße die Rückweisung von für den Menschen klar ersichtlichen falschen Eingaben, wie sehr lange Phoneme gefolgt von einigen viel zu kurzen Phonemen, war, ist die Entwicklung eines *Müllmodells* für solche Worte nachzudenken. So könnten eventuell Worte mit bestimmter offensichtlich falscher Struktur eindeutiger zurückgewiesen werden.

A Handout Wizard-of-Oz-Experiment

Brandenburgische Technische Universität Cottbus
Fakultät 3
Lehrstuhl Kommunikationstechnik

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Validierung einer Erkennergrammatik

Im Zuge des Forschungsprojekts UASR entstand das SpeechLab des Lehrstuhls für Kommunikationstechnik der BTU. Teil des SpeechLabs ist ein durch Sprachkommandos steuerbarer Versuchsaufbau. Hierfür wurde eine Kommandogrammatik entwickelt, welche in diesem Experiment mithilfe einer repräsentativen Anzahl von Probanden untersucht und validiert werden soll.

Versuchsaufbau

Im SpeechLab steht ein Aufbau mit 64 Mikrofonen bereit. 32 Mikrofone sind statisch um einen Touchscreen angeordnet. Die restlichen 32 Mikrofone befinden sich in einem weiteren Aufbau in einem bewegbaren Mikrofonfeld an der Decke. Neben jedem Mikrofon befindet sich eine Status-LED. Ist das korrespondierende Mikrofon abgeschaltet, so erlischt auch die LED. Ist das Mikrofon eingeschaltet leuchtet die LED blau.

Die Mikrofone sind nummeriert und können einzeln oder in Gruppen aktiviert und deaktiviert werden. Für die Gruppierung können entweder mehrerer Mikrofone aufgezählt werden, oder die vorgefertigten eingezeichneten Gruppen genutzt werden. Gegenüber des Bildschirms befindet sich ein Kontrollpult. Das obere Mikrofonfeld kann orthogonal zum Bildschirm zwischen diesem und dem Arbeitsplatz gegenüber vor und zurück bewegt werden. Diese

Funktionalitäten deckt die entwickelte Kommandosprache ab, sodass der Aufbau mittels Sprachkommandos bedient werden kann.

Der Spracherkenner muss vor einer Spracheingabe mit dem Schlüsselwort *Computer* geweckt werden. Erst danach können Eingaben verarbeitet werden. Die einzelnen Mikrofone sind zur Identifikation nummeriert. Die Nummerierung der Mikrofone und ihre vorgegebenen Gruppierungen, sind in den Abbildungen A.1 und A.2 dargestellt und ist während des Experimentes am Versuchsaufbau markiert.

Der aktuelle Erkennerstatus wird durch die Hintergrundbeleuchtung des Bildschirms visualisiert. Folgende Feedbacks sind möglich:

Dunkelblau Schlafmodus - Vor einer Spracheingabe muss der Computer erst geweckt werden.

Türkis Wachmodus - Erkenner ist bereit eine Eingabe anzunehmen.

Grünes Blinken Spracheingabe war gültig und wurde angenommen.

Rotes Aufleuchten Spracheingabe wurde als ungültig abgewiesen.

Ablauf des Experiments

Sie erhalten vor Beginn des Experiments eine kurze Einführung zum Versuchsaufbau, und können eventuelle Fragen stellen. Danach bekommen Sie einige Zeit das Versuchssystem zu benutzen und zu testen. Wenn eine Ausreichende Anzahl an Testdurchläufen erfolgt ist, wird der Versuchsleiter den Versuch beenden.

Hinweis: Im Zuge dieses Experiments werden Ihre Spracheingaben anonym aufgezeichnet. Diese Tonaufnahmen werden lediglich lehrstuhlintern für weitere Versuche genutzt. Eine Weitergabe an Dritte ist ausgeschlossen.

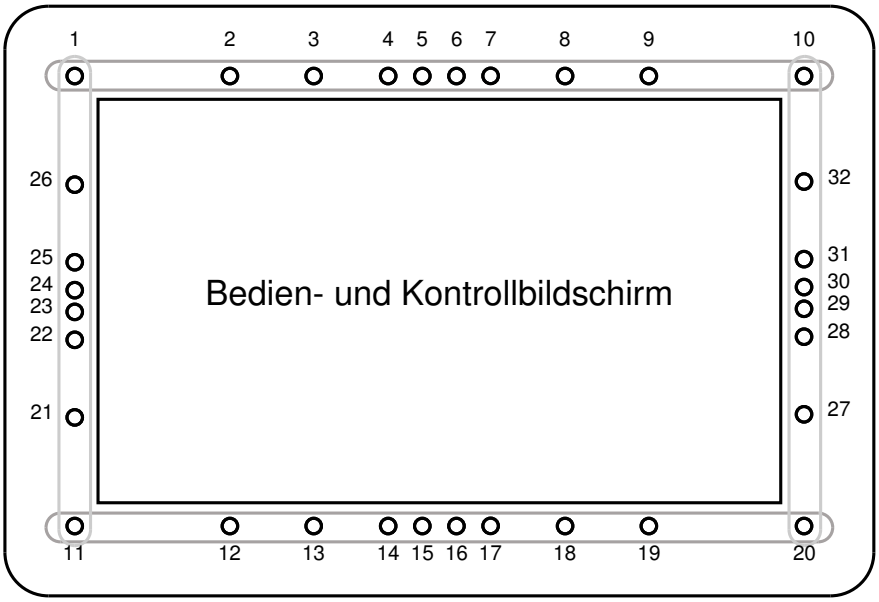


Abbildung A.1: Mikrofonanordnung mit Nummerierung am Kontrollbildschirm; Gruppentemplates grau eingerahmt

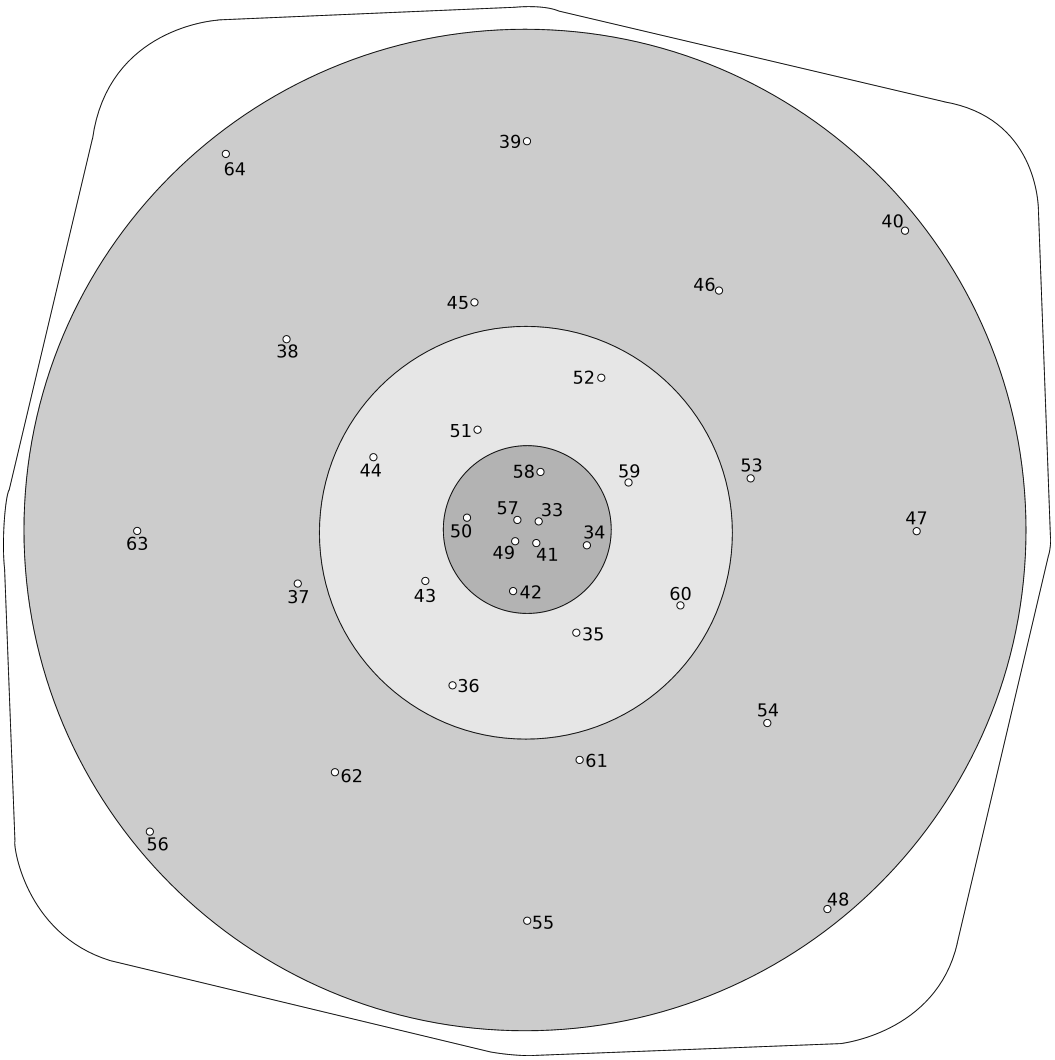


Abbildung A.2: Mikrofonanordnung im Mikrofonfeld an der Decke; Gruppentemplates mit grauen Flächen hinterlegt

B Liste der aufgenommenen Spracheingaben

B.1 Proband 1

Computer, gebe mir bitte Mikrofon 2, 3, 8, 9 und 10.
Computer, gebe mir bitte Mikrofon 2, 3, 8, 9 und 10.
Computer, schalte bitte Mikrofon 2, 3, 8, 9 und 10 aus.
Computer, fahr bitte Mikrofonfeld nach vorne.
Computer, stopp Mikrofonfeld.
Computer, schalte bitte Mikrofon 61, 58, 63, 37 aus.
Computer, schalte bitte Mikrofon 26, 13, 14, 15, 16, 17, 18, 20 aus und fahre Mikrofonfeld nach hinten.
Computer, Stopp.
Computer, bitte schalte Mikrofon 2 und 3 an.
Computer, schalte Mikrofon 27 bis 32 aus.
Computer, schalte alle Mikrofone aus.
Computer, 1, 10, 31, 18, 17, 16 an.
Computer, zwee, dree, acht neun zehn an.
Computer, 2, 3.
Computer, 55, 56, 62 an.
Computer, 55, 56, 62 anschalten.
Computer, Mikrofonfeld an.
Computer, 42, 50, 44, 58 Mikrofon an.
Computer, Mikrofonfeld nach vorn.
Computer, Mikrofonfeld stopp.
Computer, alle Mikrofone an.
Computer, alle Mikrofonfelder am Bildschirm an.
Computer, alle Mikrofone am Bildschirm aus.
Computer, mittlere Mikrofone im Mikrofonfeld aus.
Mikrofone im inneren Feld aus.

Computer, Mikrofone im inneren Mikrofonfeld aus.
Computer, Mikrofone im äußeren Mikrofonfeld aus.
Computer, Mikrofone im inneren Mikrofonfeld an.
Computer, Mikrofonfeld nach hinten.
Computer, stopp.
Computer, Mikrofonfeld nach hinten.
Computer, stopp.
Computer, Mikrofone am Bildschirm an.
Computer, Mikrofone am rechten Bildschirmrand aus.
Computer, Mikrofone am oberen Bildschirmrand aus.
Computer, jedes zweite Mikrofon am unteren Bildschirmrand aus.
Computer, alle Mikrofone, die eine Primzahl sind, an.
Computer, alle Mikrofone an.
Computer, Mikrofonfeld in die Mitte.
Computer, Mikrofon 41 und 49 aus.
Computer, Mikrofon 55 aus.
Computer, also mein Tach war an sich sehr schön, ich habe zu Hause mein Mikrofon benutzt. Ich möchte jetzt, dass du mir mal bitte die 10 ausstellst.
Du wolltest mir grade nicht die 10 ausstellen. Jetze stellste mir dafür mal bitte alle von 26 bis 21 aus, danke.
Computer, sag doch mal: kannste mir mal 21 bis 26 die Mikrofone ausstellen.
Computer, sag mal: kannste mir mal die 11 und die 20 ausstellen und danach sofort wieder einstellen.
Computer, jetzt machste mal folgendes: du fährst bitte mal das Mikrofonfeld n Stück nach vorne und dann machste mal hier komplett die Mitte aus.
Computer, das mittlere, ja mittlere Mikrofonfeld willste mir aber ausstellen oder?
Computer, mag mich nicht.
Nagut Computer, jetzt machen wir mal folgendes: Du stellst mal bitte alle am Bildschirm aus, und machst danach auch das Mikrofonfeld aus.

B.2 Proband 2

Computer, die 10 aus.
Computer, die 10 ein.
Computer, die Gruppe um die 1 aus.

Computer, die Gruppe um die 26 aus.
Computer, Gruppe 1 aus.
Computer, Mikrofone oben aus.
Computer, links aus und rechts ein.
Computer, links aus und oben ein.
Computer, alle aus.
Computer, alle aus.
Computer, alle ein.
Computer, 38 bis 44 aus.
Computer, schalte die 54 aus und die 52 aus und die 53 und die 47 aus.
Computer, schalte die 60 aus und die 54 ein.
Computer, fahre, aeh, Computer ne aus.
Computer, fahre die Platte um einen halben Meter vor.
Computer, fahre die Platte um 1,45 Meter zurück.
Schalte 3 und 10 aus.
Computer, schalte 3 und 10 aus.
Computer, schalte die 13 aus.
Computer, schalte 3, 10 ein.
Computer, schalte 3, 10 ein.
Computer, schalte 13 aus.
Computer, schalte 3, 10 und 14 aus.
Computer, schalte 3, 10 und 14 ein.
Computer, schalte 13 und 14 aus.
Computer, schalte 15 aus.
Computer, schalte 3, 10, 14 und 13 ein.
Schalte die 16 aus.
Computer, schalte die 16 aus.
Computer, schalte die 3, 10, 14, 15 und 16 ein.
Computer, schalte außen aus.
Computer, schalte alle ein.
Computer, schalte die Mitte aus.
Computer, schalte die 38 und die 62 ein.
Computer, schalte die 38 und die 62 aus.
Computer, fahre die Decke vor.
Computer, stopp.
Computer, fahre die Decke zurück um 3 cm.

Computer, fahre die Decke 3 cm hinter.
Computer, fahre die Platte zurück.
Computer, stopp.
Computer, fahre die Decke vor und dann zurück.
Computer, fahre die Decke vor und zurück.
Computer, fahre die Decke um 3 cm zurück und mache die 33 und die 57 aus.
Computer, mache eine beliebige aus.
Computer, mache ein beliebiges Mikrofon aus.
Computer, stelle dich aus.
Computer, mache links aus.
Computer, mache die rechte Seite aus.
Computer, turn the left side on.
Computer, mache die Mikrofone aus.
Computer, schalte ein Mikrofon ein.
Computer, aktiviere die 1.
Computer, lösche die 1.
Computer, entferne Mikro 1.
Computer, schalt Mikro 1 aus.
Computer, 2 ein.
Computer, 1 aus, 1 ein, 1 aus.

B.3 Proband 3

Computer, Mikrofon 1 aus.
Computer, Mikrofon 5 und 6 aus.
Computer, Mikrofon 14 und 17 aus.
Computer, Mikrofon 14 und 1 an.
Computer, Mikrofon 26, 27 und 25 aus.
Computer, Mikrofon 31 aus.
Computer, Mikrofon 1 bis 9 aus.
Computer, Mikrofon 1 bis 5 an.
Computer, Mikrofon 23 aus.
Computer, Mikrofon 44, 43 aus.
Computer, Mikrofon 64, 1 und 3 aus.
Computer, Mikrofon 11 bis 20 aus.

Computer, Mikrofon 12 bis 18 an.
Computer, Mikrofon 2 uns 12 aus.
Computer, Mikrofon 39 bis 59 aus.
Computer, Mikrofon 42 und 51 an.
Computer, Mikrofon 39 bis 59 an.
Computer, Mikrofon 1 bis 9 an.
Computer, Mikro 1 bis 3 aus.
Computer, Mikro 2 an.
Computer, Mikro 4 bis 10 aus.
Computer, Mikrofon 32, 29 aus.
Computer, Mikrofon 13 bis 18 aus.
Computer, Mikrofon 1 bis 19 an.
Computer, Mikro 2 bis 8 aus.
Computer, obere Platte bewegen, zurück.
Computer, Mikrofon 39 bis 51 aus.
Computer, Mikrofon 2 bis 9 an.
Computer, Mikrofon 12 bis 20 an.
Computer, obere Platte vor.
Computer, Mikrofon 64 an.
Computer, Mikro 1 bis 10 aus.
Computer, Mikro 1 bis 4 an.
Computer, Mikro 61 und 35 aus.
Computer, Mikro 43 bis 51 an.
Computer, Mikro 36, 55, 62, 56 aus.
Computer, Mikro 36, 62, 56 aus.
Computer, Mikro 55 und 54 aus.
Computer, Mikro 1 bis 13 aus.
Computer, 23, 22 aus.
Computer, 14 bis 19 aus.
Computer, 1 bis 18 an.
Computer, 37, 59, 53 aus.
Computer, 59, 53, 47 aus.
Computer, 62, 55, 61 an.
Computer, 35 bis 42 an.
Computer, 1 bis 10 aus.
Computer, 11 bis 20 aus.

Computer, 1 bis 18 an.
Computer, 13, 15, 17 aus.
Computer, 52 bis 62 aus.
Computer, 50 bis 61 an.
Computer, 56, 35, 51 aus.
Computer, 51 aus.
51 an, Computer, 51 an.
Computer, 56, 35 an.
Computer, 35 bis 51 aus.
Computer, 1 bis 46 aus.
Computer, 1 bis 28 an.
Computer, 18, 13 aus.
Computer, 51 bis 62 an.
Computer, 36 bis 44 an.
Computer, runterfahren.
Computer, aus.
Computer, 35 bis 49 an.
Computer, 35 an.
Computer, 13 an.
Computer, 1 bis 22 aus.
Computer, obere Platte nach zurück fahren zurück.
Computer, obere Platte zurück.
Computer, 35 bis 60 aus.
Computer, 43 bis 59 an.
Computer, obere Platte vor.
Computer, heute ist das Wetter schön. Ich seh zwar kein Wetter weil die Vorhänge davor sind.
Die Vorhänge sind aber blau, das ist hässlich und die beiden Jungs nerven mich ganz schön.
Computer, die Vorhänge sind blau.
Die Vorhänge sind nicht schön. Computer?
Computer, hier stehen 2 Computer.
Computer, hier sitzen 2 Kerle.
Computer, 2 Teller bestehen aus einem Besteck.
Computer, 2 Mikros stehen hier.
Das Wetter ist heut schön, Computer, 2 aus.
Das Wetter ist schön, Computer, 2 aus.

Das Wetter ist schön und 46 aus Wolken.
Computer, das Wetter ist schön und 46 aus.
Computer, das Wetter ist schön und 46 aus.
Computer, das Wetter ist schön und Mikro 46 aus.

B.4 Proband 4

Computer, Mikrofon 2 aus.
Computer, Mikrofone 3, 8 und 7 aus.
Computer, Mikrofone 2 bis 7 an.
Computer, Mikrofone oben aus.
Computer, Mikrofon 49 aus.
Computer, alle Mikrofone aus.
Computer, alle Mikrofone an.
Computer, fahr das Ding mal weiter nach vorne.
Computer, fahr das Ding mal weiter nach hinten.
Computer, Mikrofone rechts aus.
OK, Computer, dann gib mir doch bitte mal die rechten Mikrofone wieder.
Computer, schalt mir Mikrofone 10 bis 30 an.
Computer, fahr die Deckenplatte bitte bis ganz nach vorn.
OK, Computer, schalt mal alles aus.
OK, Computer, fahr die Platte in die Mitte.
Computer, alle Mikrofone wieder an.
Computer, Mikrofon 65 aus.
Computer, schalt mal bitte die linke Hälfte der Mikrofone am Bildschirm aus.
Computer, schalte Mikrofone 13 bis 18 aus.
Computer, invertiere Mikrofone 12 bis 19.
Computer, schalte Mikrofone 13 bis 18 an und Mikrofon 19 aus.
Computer, schalte Mikrofon 25 aus, 31 aus, und fahr die Platte nach vorne.
OK, Computer, alle Mikrofone an und die Platte in die Mitte.
Computer, schalt mal Mikrofon 0 aus.
Na gut, Computer, dann mach halt Mikrofon 1 aus.
Computer, schalte Mikrofon 1 aus.
Computer, schalte alle Mikrofone bis auf 8 aus.
Computer, alle Mikrofone an.

Computer, schalte die Mikrofone 26 bis 21 aus.
Computer, schalte die Deckenmikrofone aus.
Computer, schalte alle Mikrofone an.
Computer, schalte alle Mikrofone aus, bis auf 21 bis 30.
Computer, schalte alle Mikrofone aus, bis auf Mikrofon 22.
Computer, schalte alle Mikrofone aus und danach Mikrofon 22 an.
Computer, alle Mikrofone aus, und 22 an.
Computer, alle Deckenmikrofone an und die Platte nach hinten.
Computer, 1 bis 10 und 12 bis 18 an.
Computer, Mikrofon 8.
Computer, alle Mikrofone an und die Platte in die Mitte.
Computer, was ist dein Lieblingsgericht?
Computer, was ist heute fürn Tag?
Computer, soll ich dir was über mein Leben erzählen?
Computer, wie findest du Erdbeerketchup?
Computer, ich hab ein Mikrofon zu Haus, aber ich sage dir nicht, welche Nummer es hat.
Computer, was soll das? Warum funktionierst du nicht?
Computer, mach das.
OK, Computer, definiere mir Wissen.

B.5 Proband 5

Computer, Mikrofon 1 aus.
Computer, Mikrofon 1 bis 10 aus.
Computer, Mikrofone links aus.
Computer, alle Mikrofone aus.
Computer, Mikrofone unten an.
Computer, Mikrofone 27 bis 40 an.
Computer, Mikrofone an der Decke an.
Computer, Deckenmikrofone einen Meter nach vorne fahren.
Computer, Deckenmikrofone ganz nach hinten fahren.
Computer, Mikrofone mit geraden Nummern an.
Computer, Mikrofone mit geraden Zahlen an.
Computer, Deckenmikrofone wieder in die Mitte fahren.
Computer, äußere Mikrofone an der Decke aus.

Computer, Deckenmikrofone an.
Computer, Mikrofone ganz außen an der Decke aus.
Computer, innere Deckenmikrofone aus.
Computer, mittlere Deckenmikrofone aus.
Computer, alle Mikrofone an.
Computer, linke Deckenmikrofone aus.
Computer, vordere Deckenmikrofone aus.
Computer, innere und äußere Deckenmikrofone aus.
Computer, alle Mikrofone aus.
Computer, 2 bis 5 an.
Computer, 10 bis 15, 50 bis 60 und 30 bis 35 an.
OK, Computer, alle Mikrofone aus.
Computer, 1 bis 5, 35 bis 45 an und Deckenmikrofone einen halben Meter nach vorne.
Computer, 1 bis 10 an.
Computer, 1, 2, 3 aus und 2 an.
Computer, 1, 2, 3 aus.
Computer, 1, 2, 3 an und 3 aus.
Computer, Deckenmikrofone 3 cm nach hinten.
Computer, äußere Deckenmikrofone, innere Deckenmikrofone, Mikrofone links und rechts an.
Computer, mittlere Deckenmikrofone aus.
Computer, 12, 13, 14, 15, 16, 17, 18, 20 an, 1, 2, 3 aus.
Computer, mittlere Deckenmikrofone aus, innere Deckenmikrofone aus, äußere Deckenmikrofone aus, 4, 5, 6, 7, 8, 9 aus, 1 an.
Computer, alle Mikrofone aus.
Computer, alles auf der linken Seite an.
Computer, alle vorderen Mikrofone an.
Computer, ausschalten.
Computer, alles zwischen 50 und 60 an.
Computer, 36, 40, 50 an.
36 aus.
Computer, obere Mikrofone aus, untere Mikrofone aus, obere Mikrofone an.
Computer, links aus, rechts aus, oben aus.
Computer, Decke aus.
Computer, einmal Pommes mit Mayo bitte.
Computer, wie sieht der Wetterbericht aus.
Computer, scheint die Sonne heut eigentlich, hab ich nicht so genau gesehen.

Computer, hast schon was vor heute?
Computer, ich mag Bratwurst.
Computer, ich mag doch keine Bratwurst.

B.6 Proband 6

Mikro 1 bis 10 aus.
Computer, Mikro 1 bis 10 aus.
Gruppiere Mikrofon 1 bis 10 als oben.
Computer, gruppiere Mikrofon 1 bis 10 als oben.
Computer, deaktiviere 1.
Computer, deaktiviere 2 bis 10.
Computer, gruppiere 11 bis 20 als A.
Computer, deaktiviere 45.
Computer, bewege 20 vorwärts.
Computer, aktiviere rechts.
Computer, deaktiviere links.
Computer, rechts aus.
Computer, Mitte an.
Computer, Mitte aus.
Computer, oben an.
Computer, außen aus.
Computer, Mitte aus.
Computer, innen aus.
Computer, Center an.
Computer, rechts an.
Computer, links an.
Computer, alle an.
Computer, ungerade aus.
Computer, ungerade Nummern aus.
Computer, Bildschirm und Mitte aus.
Computer, außen und links an.
Computer, alle an.
Computer, diagonal aus.
Computer, quer aus.

Computer, vertikal aus.
Computer, links, oben, unten und rechts aus.
Computer, 38 bis 61 aus.
Computer, 52 bis 63 an.
Computer, alle aus.
Computer, vorne Mitte an.
Computer, Bildschirm aus.
Computer, Bildschirm zentriert an.
Computer, 11 bis 15 und 6 bis 10 an.
Computer, 1, 23, 26, 30, 27 an.
Computer, modulo 2 aus.
Computer, links blinken.
Computer, 62 bis 14 an.
Computer, 1, 3, 14, 4, 8, 9 aus.
Computer, oben aus.
Computer, Decke aus.
Computer, bewege 24 nach hinten.
Computer, bewege ganz nach vorne.
Computer, Bildschirm halb rechts an.
Computer, die rechte Hälfte an.
Computer, alle an.
Computer, kannst du fliegen?
Computer, alles größer 10 aus.
Computer, größer 10 aus.
Computer, 10 und größer aus.
Computer, alles über 10 aus.
Computer, 10 bis Ende aus.
Computer, 21 bis 32 aus.
Computer, links aus.

B.7 Proband 7

Mikrofon 1 aus.
Computer, Mikrofon 1 aus.
Computer, Mikrofone 2 bis 5 aus.

Computer, Mikrofone 1 bis 2 ein.
Computer, rechte Gruppe aus.
Computer, hintere Gruppe aus.
Computer, obere Gruppe an.
Computer, von mir aus linke Gruppe aus.
Computer, rechte Gruppe an.
Computer, von dir aus linke Gruppe aus.
Computer, Gruppe 46 bis 52 aus.
Computer, alle Mikros anschalten.
Computer, alle Mikros ausschalten.
Computer, Mikros 12 bis 19 an.
Computer, untere Gruppe an.
Computer, obere Gruppe an.
Computer, vordere Gruppe aus.
Computer, vordere Gruppe an.
Computer, Gruppe 19 bis 19 aus.
Computer, Mikro 18 ein und aus.
Computer, Mikro 1 aus.
Computer, Mikros 26 bis 11 aus.
Computer, Mikro 1, 25, 22, 21, 11, 12, 13 an.
Computer, Mikros 27, 28, 30 aus.
Computer, Mikros 3 bis 8 aus.
Computer, alle Gruppen aus.
Computer, linke und rechte Gruppe an.
Computer, obere und rechte Gruppe aus.
Computer, untere und rechte Gruppe an.
Computer, linke und rechte Gruppe aus.
Computer, alle Mikros an.
Computer, untere Gruppe aus.
Computer, untere Gruppe aus.
Computer, rechte Gruppe an und Mikros 18 und 19 an.
Computer, obere Gruppe aus und Mikros 12 und 13 an.
Computer, linke und rechte Gruppe aus, untere Gruppe an.
Computer, obere und rechte Gruppe an.
Computer, 26, 25 aus, 26 an.
Computer, 26 und 22 an.

Computer, 22 ein und aus.
Computer, 21 ein und aus.
Gruppe von 21 bis 19 aus.
Computer, 21 bis 20 aus.
Computer, 10 bis 0 aus.
Computer, Gruppe 49 bis 41 aus.
Computer, Nummer 58 ein und 32 aus.
Computer, 44, 45, 46 ein und 32 ein.
Computer, 43, 42 ein und 1 aus.
Computer, alle Mikros an.
Computer, Gruppe von 48 bis 59 ein.
Computer, Gruppe 48 bis 34 aus.
Computer, 1 bis 3 aus und 8 aus.
Computer, 1 bis 25 aus, 10 bis 32 aus, 27 bis 20 an.
Computer, 9 aus, 12 aus, 1 an, 27 aus, 20 aus, 46 an.
27 bis 20 an, 32 bis 40 aus.
Computer, alles an.
Computer, Gruppe 10 bis 20 aus, Gruppe 21 bis 30 aus.
Gruppe 1 bis 8 aus, Gruppe 28 bis 27 an.
Computer, 1 bis 8 aus, Gruppe 27 bis 20 an.
rechte Gruppe, untere Gruppe, linke Gruppe obere Gruppe aus.
Computer, obere Gruppe aus, rechte Gruppe aus, obere und linke Gruppe aus.
Computer, obere Gruppe aus.
Computer, 46 bis 40 aus.
Computer, obere Gruppe, rechte Gruppe, untere Gruppe, linke Gruppe an.
Computer, untere Gruppe aus, rechte Gruppe aus.
Computer, Gruppe 38 bis 51 aus.
Computer, alles aus.
Computer, 2 ein.
Computer, 1 bis 3 an.
Computer, 1 bis 4 an.
Computer, Gruppe von 2 bis 8 an.
Computer, rechte Gruppe an.
Computer, Gruppe von 27 bis 8 aus.
Computer, 1, 20, 27 an.
Computer, 20, 18, 19 an.

Computer, untere Gruppe außer 13 an.
Computer, untere Gruppe außer die 13 an.
Computer, untere Gruppe an.
Computer, untere Gruppe außer die 13 aus.
Computer, linke Gruppe, rechte Gruppe an.
Computer, 26 bis 27 aus.
Computer, alle Mikrofone einschalten.
Computer, Nummer 52 bis 20 aus.
Computer, linke Gruppe und rechte Gruppe an.
Computer, jedes zweite Mikro aus.
Computer, jedes dritte Mikro aus.
Computer, alle Mikros an.
Computer, Nummer 64 bis 38 aus.
Computer, alle Mikros an.

B.8 Proband 8

Computer, alle Mikrofone an.
Computer, alle Mikrofone anschalten.
Computer, alle Mikrofone ausschalten.
Computer, Mikrofone 1 bis 10 anschalten.
Computer, letzte Handlung rückgängig machen.
Computer, Mikrofone 10, 32, 31, 28, 27 einschalten.
Computer, alle Mikrofone ausschalten.
Computer, alle Deckenmikrofone anschalten.
Computer, Deckenplattform nach vorne fahren.
Computer, fahre die Deckenplattform nach hinten.
Computer, schalte die Deckenmikrofone aus.
Computer, schalte die Mikrofone wieder an.
Computer, fahre die Deckenplattform auf die Ausgangsposition zurück.
Computer, schalte die Mikrofone, ach Scheiße.
Computer, schalte die inneren Deckenmikrofone an.
Computer, schalte die mittleren Deckenmikrofone an.
Computer, schalte die äußeren Deckenmikrofone an.
Computer, die Fernsehmikrofone an und die Deckenmikrofone aus.

Computer, schalte die noch fehlenden Mikrofone an.
Computer, schalte die Deckenmikrofone an und fahre die Plattform nach vorn.
Computer, schalte alle Mikrofone aus, fahre die Plattform nach hinten und schalte dann die Fernsehmikrofone wieder an.
Computer, schalte Gruppe 1 an.
Computer, schalte Mikrofon 64, 44, 51, 52, 35, 61, 36, 42 aus.
Computer, zeichne mir einen Smiley.
Computer, bring mir eine Pizza.
Computer, schalte alle Mikrofone aus und fahre die Plattform wieder in ihre Ursprungsposition.
Computer, mir gehen die Ideen aus.
Computer, schalte die linken Fernsehmikrofone an.
Computer, schalte jetzt die rechten Fernsehmikrofone an.
Computer, invertiere die Mikrofone.
Computer, schalte alle Mikrofone außer die Mikrofone 4, 5, 6, 7 an.
Computer, Mikrofone 13, 14, 15 an.
Computer, Mikrofon 19.
Computer, wechsel den Zustand von Mikrofon 32.
Computer, ich möchte die Mikrofone 51, 45 und 39 benutzen.
Computer, ich schalte die Mikrofone 51, 45 und 39 an und alle anderen aus.
Computer, schalte alle Fernsehmikrofone aus.
Computer, schalte sie wieder an.
Computer, schalte die oberen und die unteren Fernsehmikrofone an.
Computer, schalte die seitlichen Fernsehmikrofone aus.
Computer, schalte die linken Fernsehmikrofone aus.
Computer, schalte die rechten Fernsehmikrofone an.
Computer, schalte die restlichen Fernsehmikrofone an.
Computer, fahre die Deckenplattform nach vorne und dann fahre die Deckenplattform nach hinten.
Computer, fahre die Deckenplattform nach vorn und dann nach hinten.
Computer, schalte die Fernsehmikrofone aus.
Computer, schalte die äußeren Deckenmikrofone an.
Computer, schalte die restlichen Deckenmikrofone an.
Computer, schalte die noch fehlenden Deckenmikrofone an.
Computer, ich brauche die Mikrofone 51 und 58.
Computer, schalte die Mikrofone 37 und 63 ab.
Computer, schalte das Mikrofon 70 an.

Computer, schalte das Mikrofon 65 an.
Computer, schalte das Mikrofon 0 an.
Computer, schalte das Mikrofon 2 plus 3 an.
Computer, schalte das Mikrofon 4, 5, 6, 7, 8 und 9 an.
Computer, schalte die Mikrofone 12 bis 64 an.
Computer, schalte die Mikrofone wieder aus.
Computer, schalte alle Mikrofone aus.
Computer, Deckenplattform auf Ausgangsposition.

B.9 Proband 9

Computer, ah cool.
Computer, schalte Mikrofon 23 bis 27 aus.
Computer, schalte Mikrofon 23 bis 27 wieder an.
Computer, schalte alle Mikrofone aus.
Computer, schalt die Mikrofone wieder an.
Computer, aktiviere alle Mikrofone.
Computer, bewege oberes Mikrofonfeld.
Computer, bewege das obere Mikrofonfeld 30 cm nach vorne und schalte alle Mikrofone aus.
Computer, aktiviere alle Mikrofone.
Computer, schalte die Mikrofone 3 bis 5, 25 bis 21 und 14 bis 19 aus.
Computer, schalte 25, Mikrofon 25 bis 21 an.
Computer, deaktiviere Mikrofon 1.
Computer, deaktiviere Mikrofon 23.
Computer, deaktiviere Mikrofon 10, 23 und 29.
Computer, aktiviere alle Mikrofone.
Computer, deaktiviere alle Mikrofone.
Computer, aktiviere Mikrofon 1, 4, 7, 19 und 46.
Computer, aktiviere Mikrofon 25, 23, 63.
Computer, schalte Mikrofone 2, 23 bis 27 ach ne ach Gott.
Computer, schalte Mikrofon 2, 32 bis 27, 14 bis 18 und 20 ein.
Computer, bewege die obere Mikrofonplattform bis zum Ende.
Computer, fahre den oberen Mikro...ach Scheiße wie hießn das?
Computer, fahre das obere Mikrofonfeld nach, bis zum Bildschirm.
Computer, bewege das Mikrofonfeld nach hinten.

Computer, fahre das Mikrofonfeld nach vorn.
Computer, fahre das Mikrofonfeld in die Mitte.
Computer, fahre das Mikrofonfeld 30 cm nach vorn.
Computer, schalte alle Mikrofone an.
Computer, fahre das Mikrofonfeld 50 cm nach hinten.
Computer, schalte Mikrofon 1 bis 10, 23 bis 30 und 59 bis 64 ab.
Computer, schalte alle Mikrofone ab.
Computer, aktiviere Mikrofon 1.
Computer, aktiviere alle restlichen Mikrofone.
Computer, aktiviere die Mikrofone 2 bis 38.
Computer, aktiviere die Mikrofone 44 bis 64.
Computer, aktiviere die Mikrofone 39 bis 43 und schalte Mikrofon 50 bis 58 ab.
Computer, fahre die Mikrofonplattform nach vorn.
Computer, schalte alle Mikrofone ab.
Computer, aktivieren die Mikrofone 26, 35, 47, 62 und 11.
Computer, fahre die Mikro, das Mikrofonfeld in die Mitte.
Computer, aktiviere alle Mikrofone.
Computer, mach Partylicht.
Computer, schalte Mikrofon 1 ab und an.
Computer, mach mir Essen, ich hab Hunger.
Computer, aktiviere Mikrofon 65.
Computer, tanze.
Computer, spiel Musik ab.
Computer, dreh das Mikrofonfeld.
Computer, schalte Mikrofon 49, 41, 33, 57 ab.

B.10 Proband 10

Computer, Start.
Computer, Mikrofon 1 aus.
Computer, Mikrofon 14 aus.
Computer, Mikrofon 1 an.
Computer, Mikrofongruppe unten aus.
Computer, Gruppe Mikrofon oben aus.
Computer, Mikrofone oben Gr... ach.

Computer, Mikrofongruppe oben aus.
Computer, Mikrofone ach.
Computer, Mikrofongruppe oben Mitte aus.
Computer, alle Mikrofone aus.
Computer, alle Mikrofone an.
Computer, Mikrofone Decke außen aus.
Computer, Mikrofone Bildschirm aus.
Computer, alle Mikrofone Bildschirm an.
Computer, alle Mikrofone Decke aus.
Computer, alle Mikrofone aus.
Computer, Computer aus.
Computer, Mikrofone 1 bis 11 an.
Computer, Mikrofone 28 bis 31 an.
Computer, Mikrofone 22 und 26 an.
Computer, Bildschirm unten alle Mikrofone an.
Computer, neue Gruppe Mikrofon 11 und Mikrofon 10.
Computer, 27 an.
Computer, Mikrofone 64 und 39 an.
Computer, Mikrofone 47 und 48 an.
Computer, alle Mikrofone mit gerader Nummer an.
Computer, alle Mikrofone kleiner 10 an.
Computer, Mikrofon 20 aus.
Computer, Mikrofon mit Zahl kleiner 3 aus.
Computer, Mikrofone oben Mitte aus.
Computer, Mikrofone Bildschirm oben Mitte aus.
Computer, Mikrofon 100 aus.
Computer, Mikrofon 90 aus.
Computer, Mikrofon 45 aus.
Computer, Mikrofon Bildschirm rechts aus.
Computer, Mikrofone Bildschirm links aus.
Computer, Mikrofone Decke an.
Computer, alle Mikrofone aus.
Computer, Mikrofon Nummer 20 an.
Computer, Mikrofon Nummer 1 an.
Computer, Mikrofon Nummer 3 an.
Computer, Mikrofon Nummer 13 an.

Computer, Mikrofon Nummer 21 an.
Computer, Mikrofon Nummer 10 an.
Computer, alle Mikrofone Bildschirm aus.
Computer, Mikrofon Nummer 21 an.
Computer, Mikrofon Nummer 51 an.
Computer, Mikrofone Decke außen an.
Computer, Mikrofone Decke Zentrum an.
Computer, Mikrofone Decke in der Mitte an.
Computer, Mikrofone an der Decke mittig an.
Computer, Mikrofone an der Decke an.
Computer, Mikrofon Nummer 21 an.
Computer, Mikrofone Bildschirm links und rechts an.
Computer, Bildschirm Mikrofone oben und unten an.
Computer, Mikrofone Bildschirm links und rechts aus.
Computer, Mikrofone Decke aus.
Computer, alle Mikrofone an.
Computer, Mikrofone Bildschirm oben und unten aus.
Computer, Mikrofone Bildschirm rechts aus.
Computer, Mikrofone n der Decke in der Mitte aus.
Computer, Mikrofone direkt über mir aus.
Computer, alle Mikrofone an der Decke an.
Computer, alle Mikrofone Bildschirm unten an.
Computer, ein Döner bitte.
Computer, ich mag Züge.
Computer, Computer.
Computer, fahre Decke einen Meter nach vorne.
Computer, fahre Decke ein Inch zurück.
Computer, fahre Decke komplett zurück.
Computer, Decke stopp.
Computer, OK, ich habs verstanden.
Computer, spiel geile Musik ab.
Computer, XBOX go home.
Computer, bitte leuchte rot.
Computer, sei mal fröhlich.
Computer, mach mal was schönes.
Computer, Decke Ausgangsstellung.

Computer, Mikrofone Bildschirm oben an.
Computer, Bildschirmmikrofone diagonal an.
Computer, Bildschirm aus.
Computer, Selbstzerstörung.
Computer, Döner.
Computer, ich möchte Pommes rot weiß.
Wenn ich jetzt Bratwurst sage reagiert er gar nicht.
Computer, blinke bitte bunt.
Computer, mach bitte die Mikrofone alle aus.
Computer, würden Sie höflicherweise alle Mikrofone einschalten.
Computer, bitte seien Sie so nett und schalten Sie alle Mikrofone ein.
Computer, mir fällt nichts mehr ein, kannst du mir weiterhelfen?
Computer, alle Mikrofone an der Decke aus und alle Mikrofone am Bildschirm an.

B.11 Proband 11

Computer, die Mikrofone funktionieren alle über diese Zahlen da?
Ja, mach mal hier die 4 aus.
Computer, mach mal die 4 aus.
Computer, 10 aus.
Computer, 14, 15 und 16 mach aus.
Computer, alle aus.
Computer, alle geraden Zahlen an.
Computer, die vier anschalten.
Computer, die 27 auch.
Computer, verschieb Platte nach vorn.
Computer, stopp.
Computer, schalte an Mikrofone 64, 45, 51, 22, 30 und 27.
Computer, fahr Scheibe einen Meter nach hinten.
Computer, fahr Platte einen Meter nach hinten.
Computer, alle Mikrofone an.
Computer, schalt die Hälfte aus.
Computer, schalte aus Mikrofon 12 bis 25.
Computer, schalte aus Gruppe oben.
Computer, mach an Gruppe links.

Computer, schalte alle aus.
Computer, schalte an Mikrofone 3, 7, 9, 15 und 19.
Computer, alle an.
Computer, Scheibe nach vorn.
Computer, Gruppe 1 aus.
Computer, Gruppe außen aus.
Computer, Gruppe dunkelgrau aus.
Computer, alle Mikrofone an.
Computer, Gruppe links und Rechts aus.
Computer, alle Gruppen aus.
Computer, zwei Gruppen an.
Computer, Mikrofone 10-30 an.
Computer, Scheibe nach hinten.
Computer, stopp.
Computer, Gruppe innen an.
Computer, mach mal an, die 55, 52, 38 und die 37.
Computer, 64 an.
Computer, die 40, die 47 und die 55 an.
Computer, 56 und 53 an.
Computer, Scheibe nach hinten.
Computer, 3 an.
Computer, stopp.
Computer, alle dreißiger aus.
Computer, 39 an.
Computer, kann der auch Englisch?
Lieber Computer, schalte doch mal bitte die 48 und die 56 ein.
Computer, 63 auch.
Computer, 63 an.
Computer, Scheibe zum Ursprung.
Computer, alle Mikrofone an.
Computer, alle Mikrofone aus.
Computer, eins, zwei drei starten.
Computer, Gruppe rechts anschalten.
Computer, Gruppe oben und unten anschalten.
Computer, Gruppe rechts ausschalten.
Computer, Gruppe unten beenden.

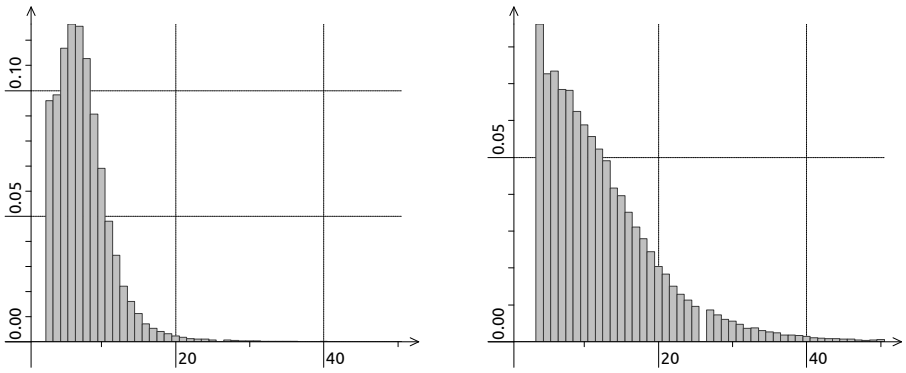
Computer, alle Mikrofone auf Scheibe an.
Computer, alle Mikrofone auf Monitor an.
Computer, schalte doch mal die 26 aus.
Computer, schalte die 26 aus.
Computer, schalte die 45 auch aus.
Computer, 64 aus.
Computer, 64 zurück.
Computer, die 64 an.
Computer, alle Mikrofone aus, bis auf die 3.
Computer, alle Mikrofone aus.
Computer, Gruppe Mitte an.
Computer, Gruppe innen an.
Computer, Scheibe nach hinten.
Computer, stopp.
Computer, Scheibe 10 cm nach vorne.
Computer, Scheibe ganz nach vorne.
Computer, Gruppe oben und unten an.
Computer, mach mal die 32 an.
Computer, tust du bitte die 32 anmachen.
Computer, zu, ne mit zu das kriegt der nicht hin. Mach mal bitte, jetzt ist er schon aus.
Computer, mach die 32 an, jetzt.
Computer, jetzt mach sie aus, die 32.
Computer, 32 aus.
Computer, beenden.
Computer, Scheibe nach hinten.
Computer, innen aus.
Computer, außen an.
Computer, stopp.
Also, wies Wetter ist, weiß ich leider nicht. Ich kann nicht raus gucken.
Ich schätze mal es ist immernoch bewölkt, und die Sonne scheint nicht.
Ja ich bin grad von Arbeit gekommen und durfte heut den ganzen Tag Parkhäuser planen.
Computer, hallo wie gehts dir? Mir gehts gut danke.
Ja, ich weiß jetzt nicht, was ich dir erzählen soll.
Computer, ich werd demnächst hoffentlich das vollendete Programm sehen, was hier geschrieben wird.
Ich kann übrigens keine kurzen Sätze fassen, ich sprech immer so lang.

Computer, magst du Schnitzel.
Computer, wie gehts dir? Wie alt bist du?
Hast du Geschwister?
Computer, bist du alleine?
Der arme, der sitzt hier jeden Tag alleine, nur diese komischen Laptops hier.
Ist das eigentlich ein, das ist doch einfach nur ein Flachbildschirm hier oder?

C Phonemdauerstatistiken

Im folgenden sind die dieser Arbeit zugrunde liegenden Phonem-Dauer-Statistiken dargestellt. Die Histogramme wurden aus der erweiterten deutschen Trainingsstichprobe der Verbmobil Datenbasis (VMX) ermittelt. Auf den Abszissen sind die Phonemdauern in Anzahlen von Merkmalvektoren aufgetragen. Ein Merkmalvektor entspricht einem Zeitframe von 10 ms Länge. Die Ordinaten stellen die relativen Häufigkeiten für die entsprechenden Dauern dar. Die Histogramme zeigen somit eine Schätzung für die Wahrscheinlichkeitsfunktion der Phonem-Dauern.

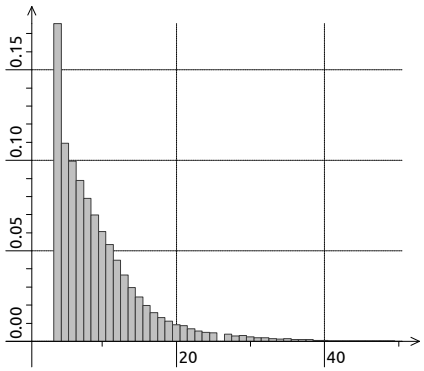
Bemerkenswert ist, dass kein einziges Phonem im Verbmobil-Korpus mit einer Länge von 26 Merkmalvektoren auftritt. Dies bedeutet eine leichte Verzerrung der Ergebnisse, die im Rahmen dieser Arbeit allerdings als unkritisch eingeschätzt und daher ignoriert wurde. Um Nullwerte zu vermeiden wurden die absolut ausgezählten Werte mit einer Add-One-Glättung angepasst.



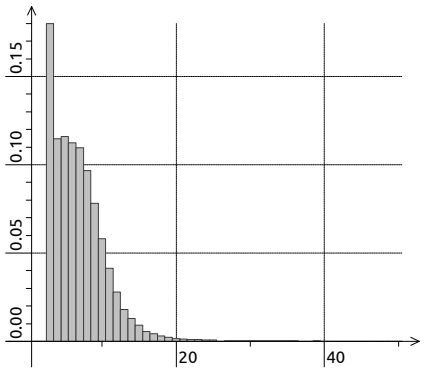
(a) Vokal **a** wie in k**a**nn

(b) Vokal **a:** wie in K**a**hn

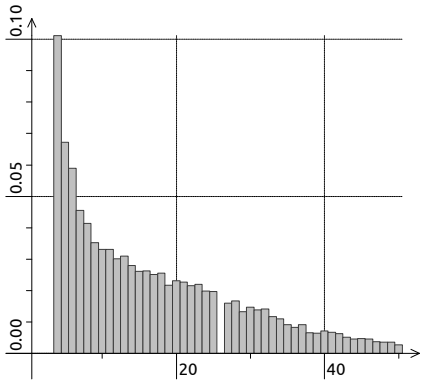
Abbildung C.1: Phonem-Dauer-Statistik der ersten zwei Vokale



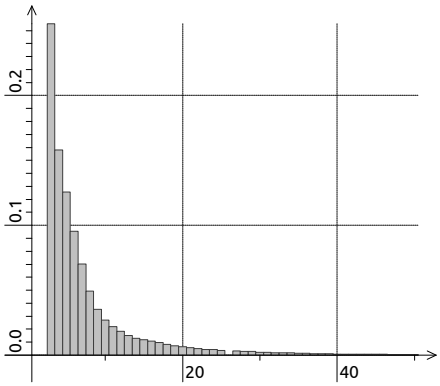
(a) Vokal **e:** wie in B**ee**t



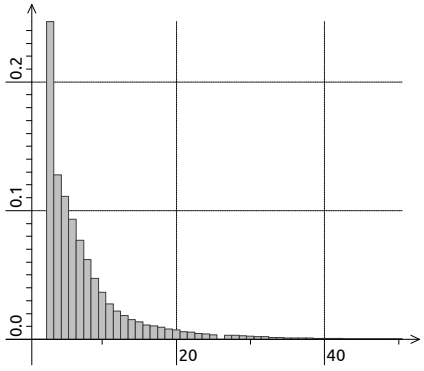
(b) Vokal **E** wie in B**e**tt



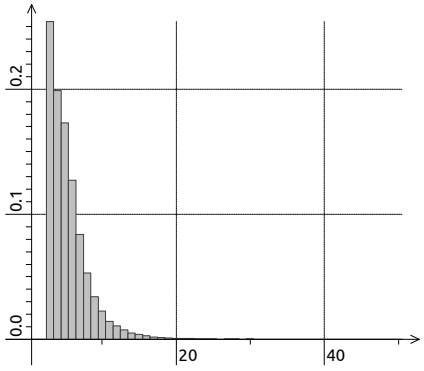
(c) Vokal **E:** wie in K**ä**se



(d) Vokal **@** wie in les**e**n

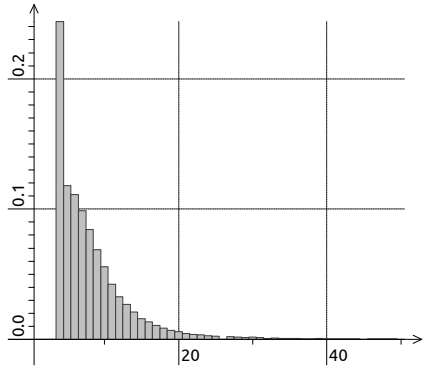


(e) Vokal **6** wie in Les**er**

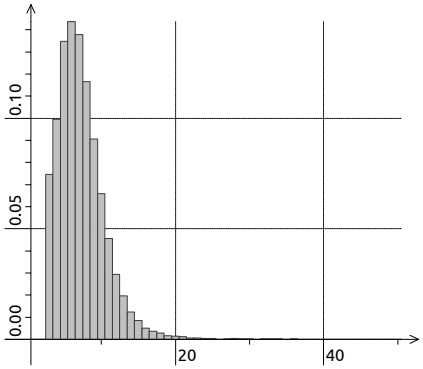


(f) Vokal **I** wie in r**i**tt

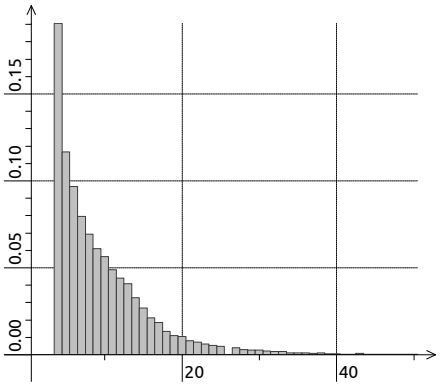
Abbildung C.2: Phonem-Dauer-Statistik von sechs Vokalen



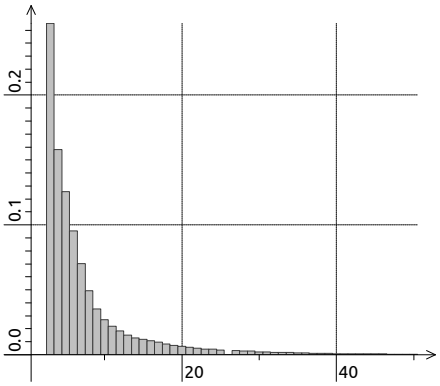
(a) Vokal **i:** wie in r**ie**t



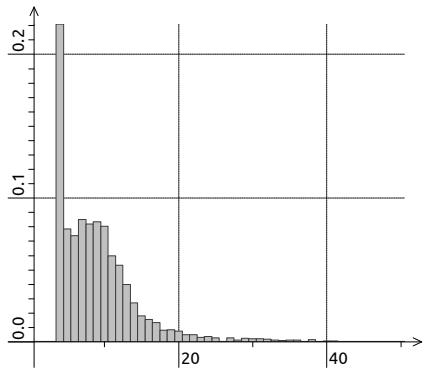
(b) Vokal **O** wie in B**o**ck



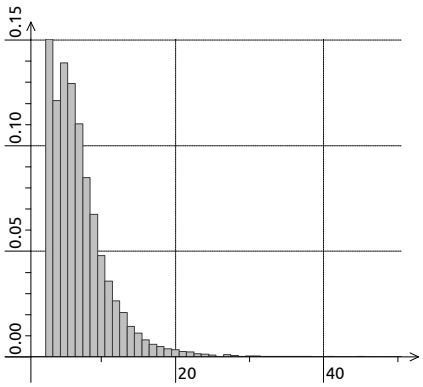
(c) Vokal **o:** wie in b**o**g



(d) Vokal **9** wie in H**ö**lle

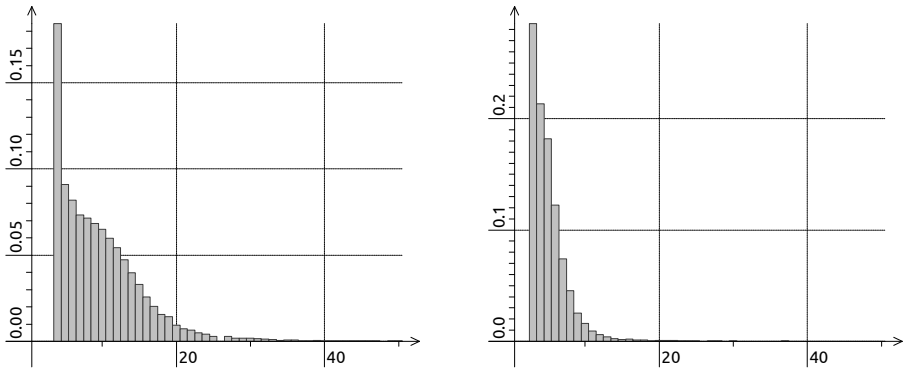


(e) Vokal **2** wie in H**ö**hle



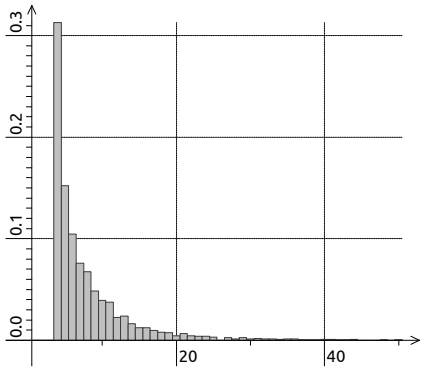
(f) Vokal **U** wie in m**u**ss

Abbildung C.3: Phonem-Dauer-Statistik von sechs Vokalen

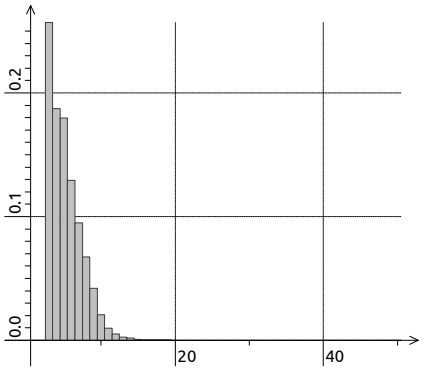


(a) Vokal **u:** wie in M**u**s

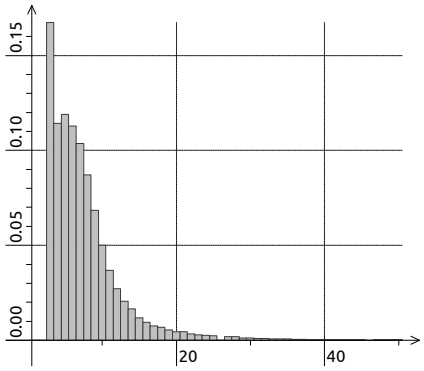
(b) Vokal **Y** wie in H**ü**tte



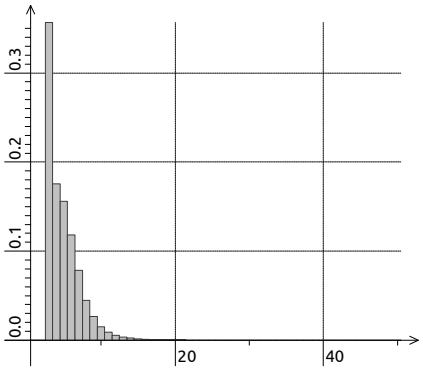
(c) Vokal **y:** wie in H**ü**te



(d) Konsonant **b** wie in **b**ei

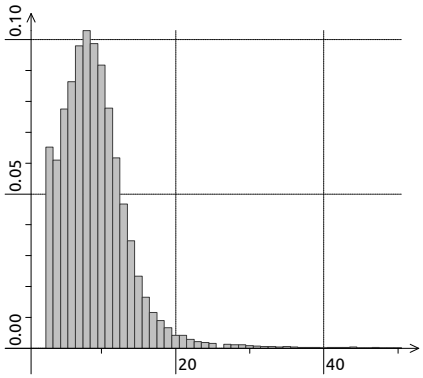


(e) Konsonant **C** wie in di**ch**

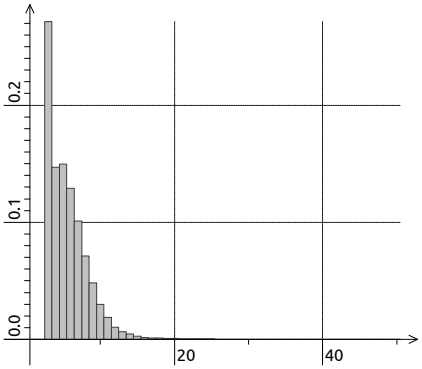


(f) Konsonant **d** wie in **d**u

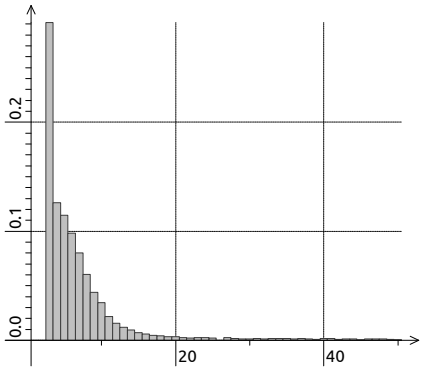
Abbildung C.4: Phonem-Dauer-Statistik von sechs Vokalen und Konsonanten



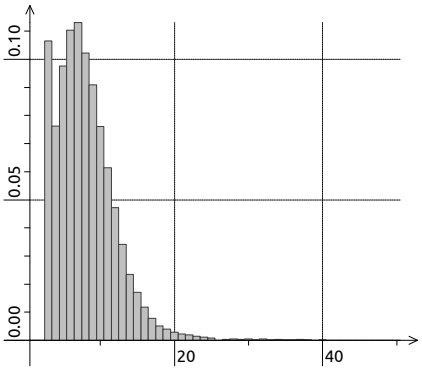
(a) Konsonant **f** wie in ver**f**ahren



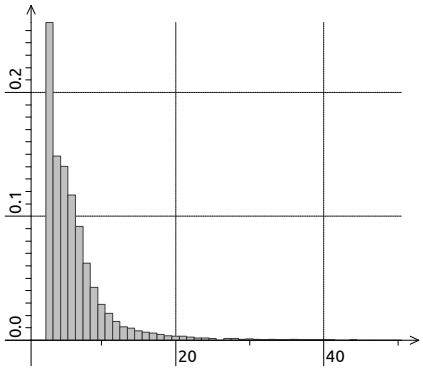
(b) Konsonant **g** wie in **G**ast



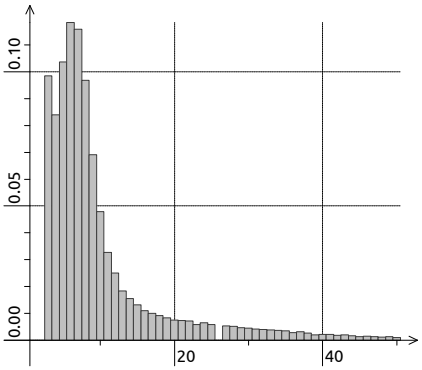
(c) Konsonant **h** wie in **H**ast



(d) Konsonant **k** wie in **K**ahn

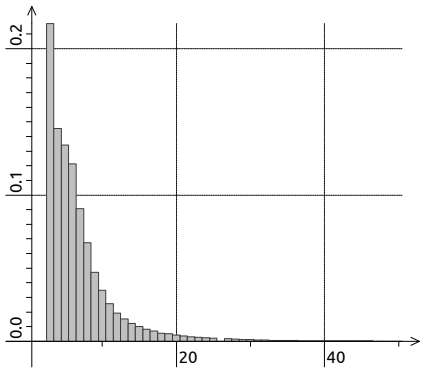


(e) Konsonant **l** wie in **L**icht

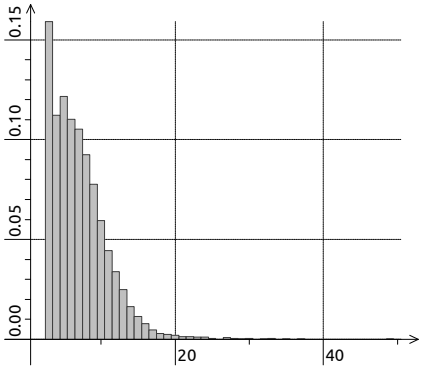


(f) Konsonant **m** wie in **M**ann

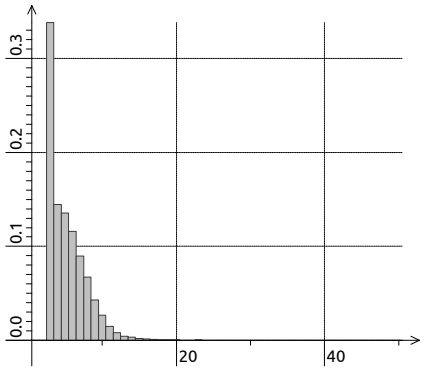
Abbildung C.5: Phonem-Dauer-Statistik von sechs Konsonanten



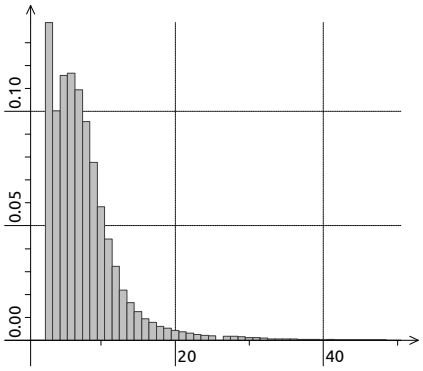
(a) Konsonant **n** wie in ver**n**eun



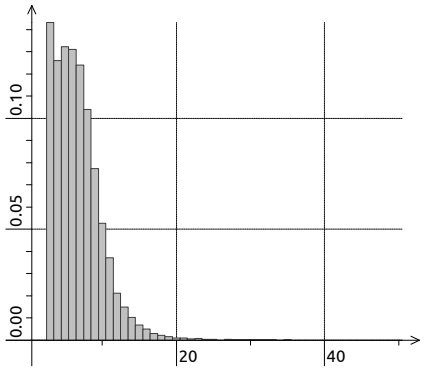
(b) Konsonant **p** wie in **P**latz



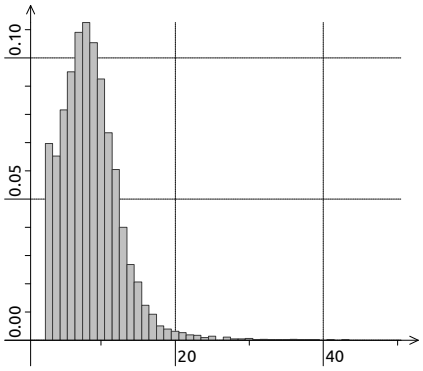
(c) Konsonant **r** wie in **R**auch



(d) Konsonant **s** wie in la**s**

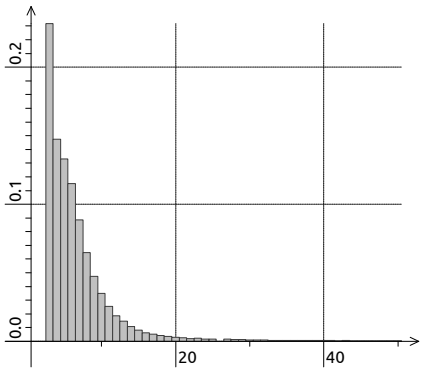


(e) Konsonant **z** wie in le**s**en

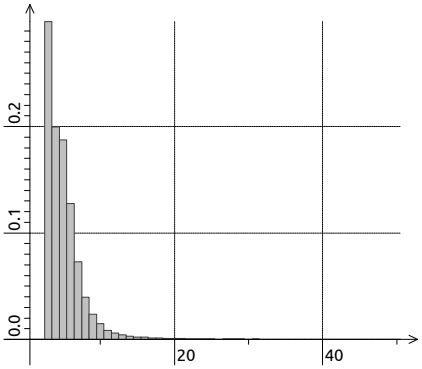


(f) Konsonant **S** wie in Ta**sch**e

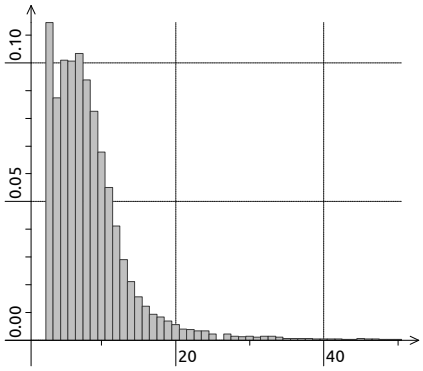
Abbildung C.6: Phonem-Dauer-Statistik von sechs Konsonanten



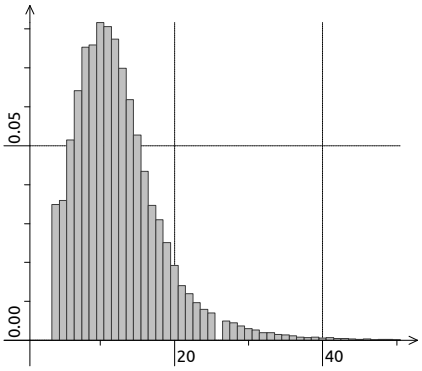
(a) Konsonant **t** wie in **T**orte



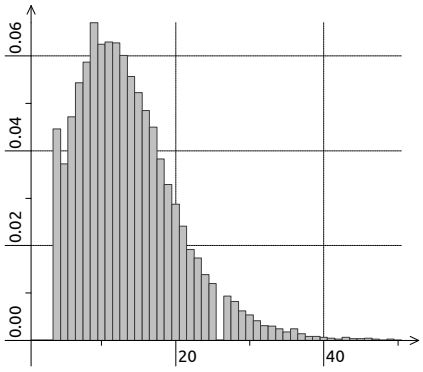
(b) Konsonant **v** wie in **V**ase



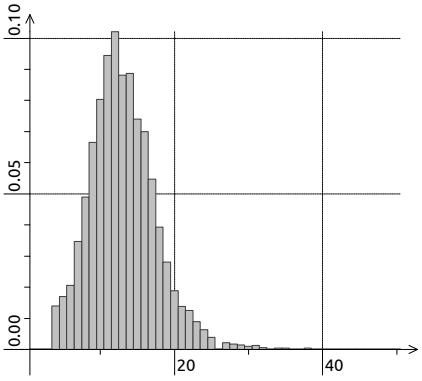
(c) Konsonant **x** wie in Da**ch**



(d) Diphthong **aI** wie in zw**ei**

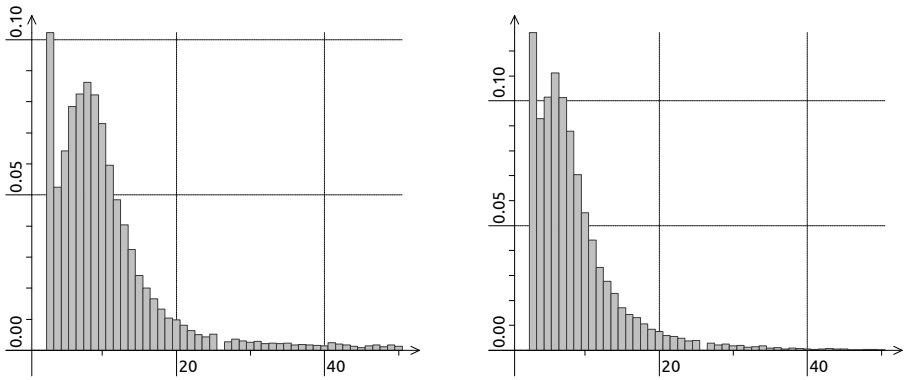


(e) Diphthong **aU** wie in B**au**ch



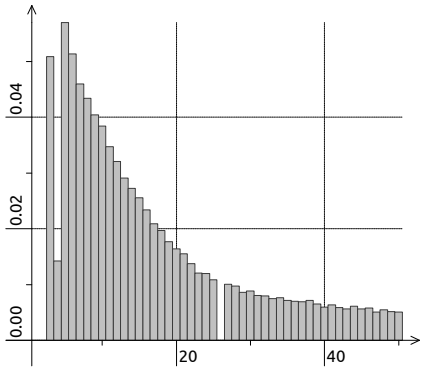
(f) Diphthong **OY** wie in n**eu**

Abbildung C.7: Phonem-Dauer-Statistik von drei Konsonanten und drei Diphthongs

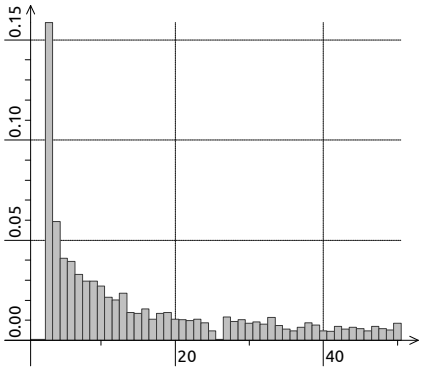


(a) Semivokal **j** wie in **j**a

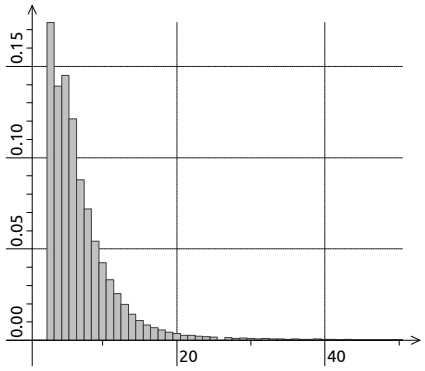
(b) Nasal **N** wie in Ju**ng**e



(c) Stille **.**



(d) Nicht-Sprache #



(e) Glottisschlag **Q**

Abbildung C.8: Phonem-Dauer-Statistik der letzten fünf Phoneme

D Standardisierte Histogramme der Merkmale

Nachfolgend sind die standardisierten Histogramme der einzelnen Konfidenzmerkmale abgebildet. Sie stellen die Verteilung der Erkennerergebnisse in den vier Klassen *OOT* - out of topic (entspricht nicht der Grammatik), *NSP* - no speech (nichtsprachliche Eingaben), *ERR* - error detection (Annahme von anzunehmender Eingabe bei nicht matchender Erkennung) und *COR* - correct (korrekt angenommen und erkannte Eingaben). Die Abszissen stellen den Merkmalswert dar, während auf den Ordinaten die absoluten Häufigkeiten aufgetragen sind. Die einzelnen Klassen sind pro Merkmalswert gestapelt.

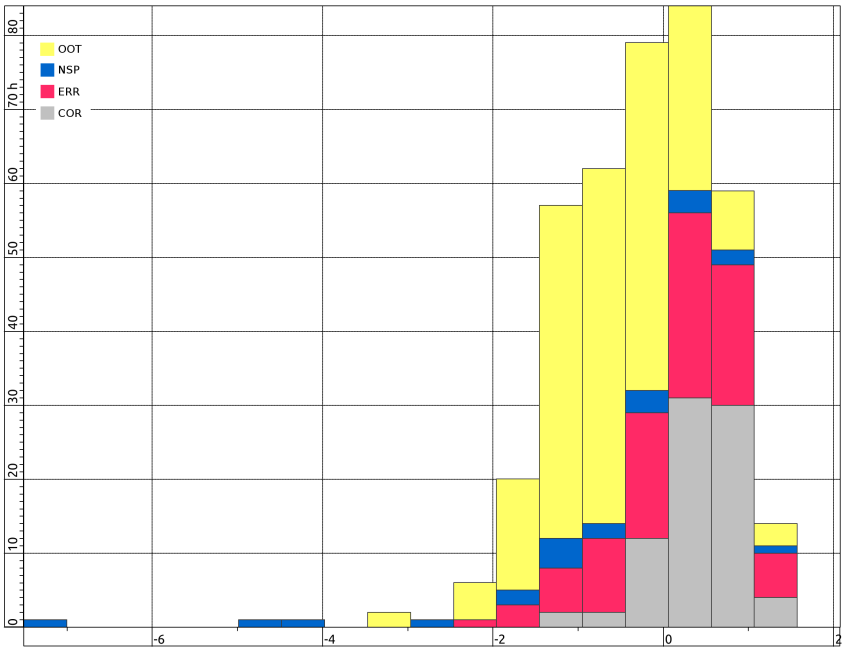


Abbildung D.1: Standardisiertes Histogramm NAD

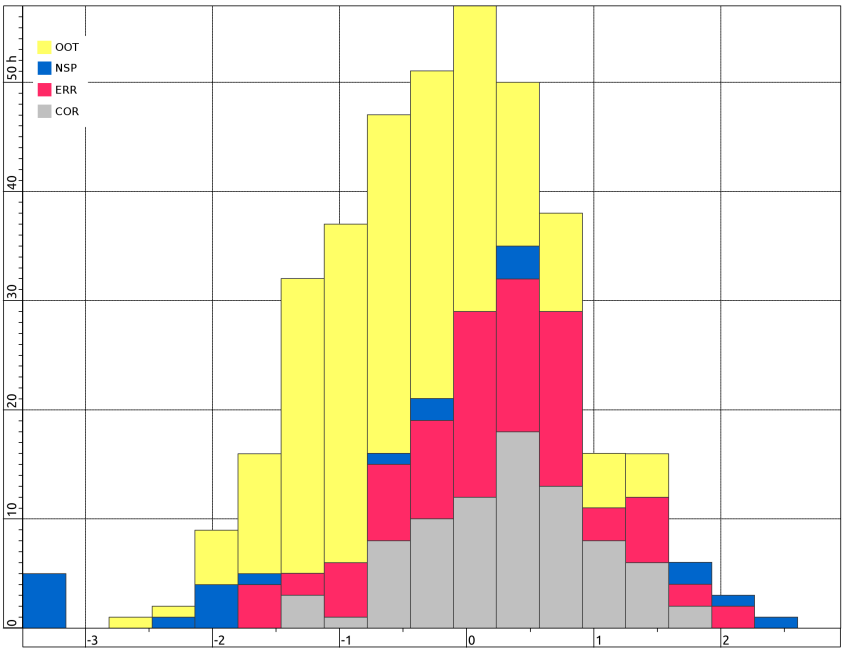


Abbildung D.2: Standardisiertes Histogramm NHD

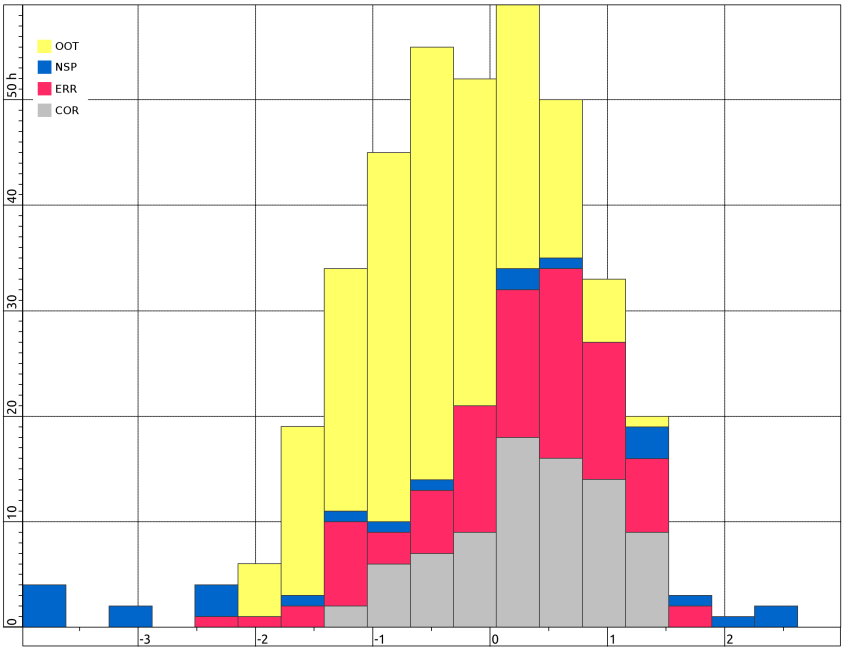


Abbildung D.3: Standardisiertes Histogramm NWHD

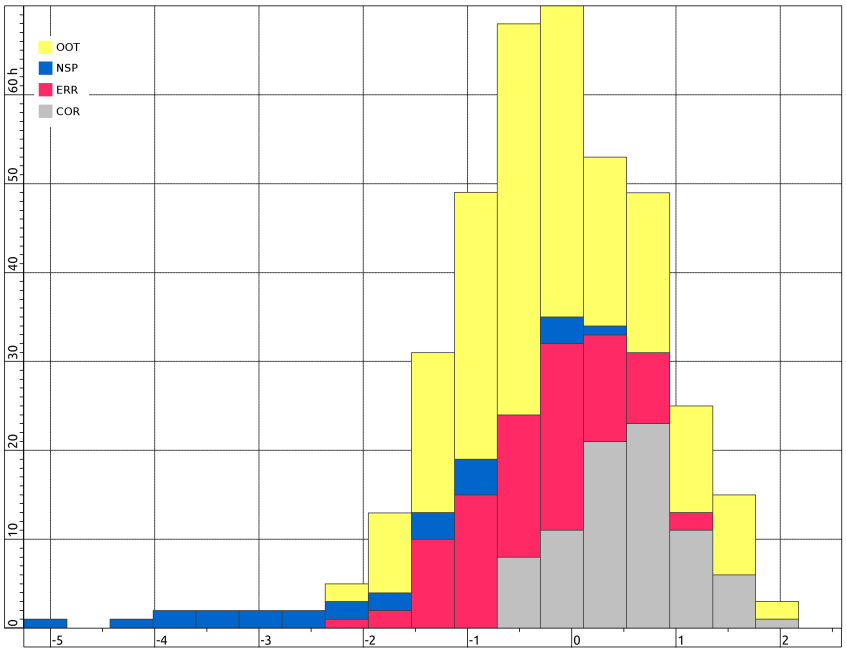


Abbildung D.4: Standardisiertes Histogramm NWLD

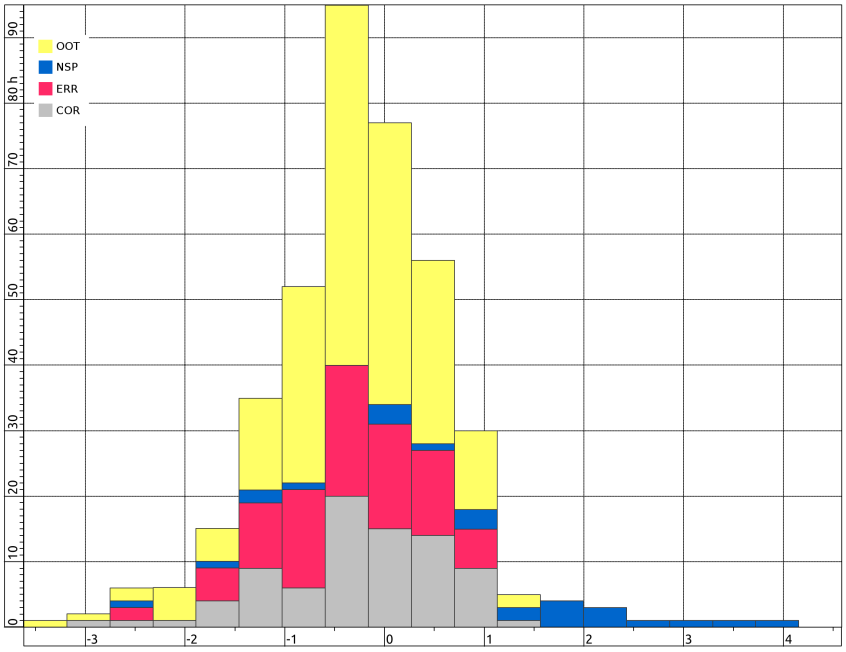


Abbildung D.5: Standardisiertes Histogramm PDL

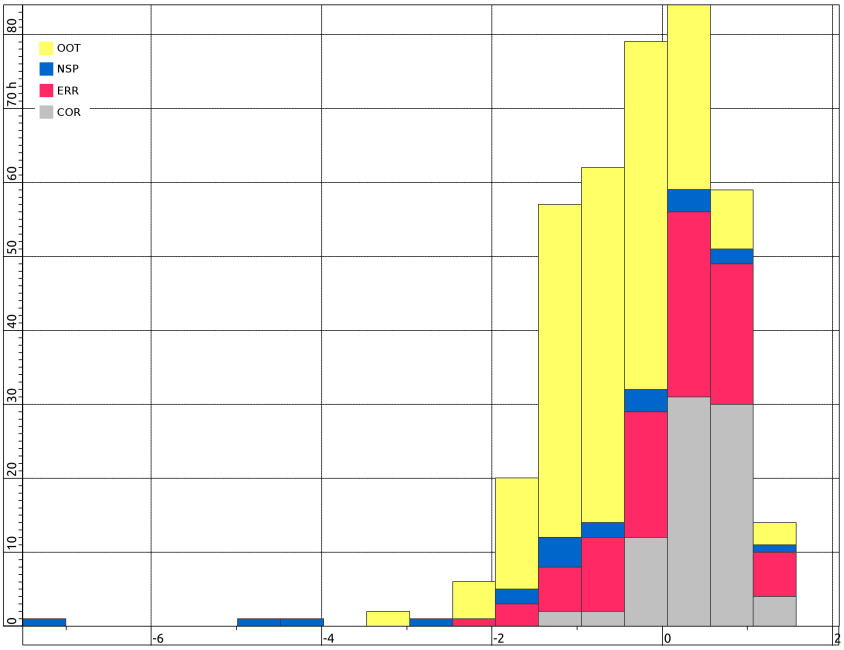


Abbildung D.6: Standardisiertes Histogramm NAD

E Codelisting Konfidenzschätzung

```
#!/usr/bin/env dlabpro
## Unified Approach to Speech Synthesis and Recognition
## - OOT rejection test program
##
## AUTHOR : Matthias Wolff, Leonard Foerster
## PACKAGE: uasr/_work/rejection
##
## ARGUMENTS:
##   $1: Configuration file

## TODO:
## 1. Implement per-frame phoneme duration likelihood score
## 2. Toggle evaluation modes (TP2 to be accepted OR rejected)

"$UASR_HOME/scripts/dlabpro/util/cfg.itp"       include;                        # Include configuration utilities
"$UASR_HOME/scripts/dlabpro/util/fea.itp"       include;                        # Include feature utilities
"$UASR_HOME/scripts/dlabpro/util/fst.itp"       include;                        # Include fst utilities
"$UASR_HOME/scripts/dlabpro/util/histogram.itp" include;                        # Include histogram utilities
"$UASR_HOME/scripts/dlabpro/util/lab.itp"       include;                        # Include label utilities
"$UASR_HOME/scripts/dlabpro/util/sig.itp"       include;                        # Include signal utilities
"$UASR_HOME/scripts/dlabpro/util/uasr.itp"      include;                        # Include UASR utilities
"$UASR_HOME/scripts/dlabpro/util/var.itp"       include;                        # Include UASR variables' utilities

## -- Constants --                                                            # ====================================
var S_WRONGNOFRAMES; "Wrong number of frames." S_WRONGNOFRAMES =;               # Used in error messages

## -- Auxiliary Functions --                                                  # ====================================

## Loads a signal file. This function is used to overload <code>-SIG_get</code>.
##
## @param iF file
##          File list instance.
## @param sSns string
##          -- ignored --
## @param idSig data
##          Will be filled with the signal.
## @global sSigDir              CWR
## @global &lt;uasr.dir.sig&gt; R
## @return <code>TRUE</code> if successful, <code>FALSE</code> otherwise.
function -SIG_get_rr(iF,sSns,idSig)
{
  ".sSigDir" "var" ?instance not if                                            # Is there a global sSigDir? No >>
    var .sSigDir;                                                              #   Create one
    ( "sig" "S" -CFG_get_path ) .sSigDir -sset;                                #   Silently get signal files path
  end                                                                          # <<
  .sSigDir iF.sfile idSig -SIG_import_wav;                                     # Import wave file
  idSig -is_empty not return;                                                  # Return success
}

## Pretty-prints an evaluation count or quota.
```

```
51 function -print_quota(sName,sHits,sTrials,idEvl)
52 {
53   function -sum(sKeys,idEvl)
54   {
55     data idKeys;
56     var  i;
57     var  nC;
58     var  sum;
59     { sKeys } "+" "split" idKeys -strop;
60     i idKeys.dim < while
61       ( :idKeys[0,i]: idEvl -find_comp ) nC =;
62       nC 0 < if "Count \"${idKeys[0,i]}\" invalid" 1 -WARNING; end
63       :sum+=idEvl[0,nC];
64       i ++=;
65     end
66     sum return;
67   }
68
69   "\n   - " ( sName 3 -MSG_pad ) + " = " +                                -MSG; # Protocol
70   var C; ( sHits idEvl -sum ) C =;                                              # Count hits
71   sTrials not if                                                                # Print a count >>
72     ( C 5 0 -FMT_f )                                                       -MSG; #   Protocol
73   else                                                                          # << Print a quota >>
74     var N;   ( sTrials idEvl -sum )        N   =;                               #   Count trials
75     var c95; :2*sqrt{(C/N-C*C/N/N)/(N-1)}: c95 =;                               #   Compute 95% confidence
           interval
76     ( :C/N*100: 5 1 -FMT_f ) " +-" + ( :c95*100: 5 1 -FMT_f ) + " %" +   -MSG; #   Protocol
77   end                                                                           # <<
78 }
79
80 ## Creates (and logs) a confidence score histogram.
81 ##
82 ## @idScores data
83 ##    One record of scores per evaluated turn. The last component is expected to
84 ##    contain the label of a histogram class.
85 ## @sCname string
86 ##    The score to create the histogram for; a component name im
87 ##    <code>idScores</code>.
88 ## @idHist data
89 ##    Filled with the histogram (may be <code>NULL</code>).
90 ##    <code>idHist.descr1</code> is set to the empirical mean,
91 ##    <code>idHist.descr2</code> is set to the empirical standared deviation,
92 ##    and <code>idHist.rtext</code> ist set to the histogram name
93 ##    <code>sCname</code>.
94 function -CONF_histogram(idScores,sCname,idHist)
95 {
96   histogram iH                                                                  # Score histogram
97   data      idH;                                                                # Score histogram data
98   data      idLtb;                                                              # Histogram label table
99   data      idScores2;                                                          # Copy of selected scores
100  data      idAux;                                                              # Auxiliary data
101  var       sLogDir;   ( "log" "S" -CFG_get_path ) sLogDir =;                   # Get log folder
102
103  idHist NULL != if idHist -reset; end                                          # Reset target object
104  :idScores["$[sCname]"]: idScores2 =;                                          # Copy selected scores
105  idScores2 -is_empty if leave; end                                             # Got nothing --> forget it
106  "CLS" 16 idScores2 -addcomp; idScores idScores2 -copy_labels;                 # Copy score labels
107  :idScores2[1]: idLtb -copy; idLtb 0 idLtb -sortup; idLtb 0 idLtb -compress;   # Make histogram label table
108  :idScores2[0]: 255 idAux /force -tconvert;                                    # Convert scores to strings
109  idAux { "nan" } 0 0 idAux /noerror -gen_index; :idAux.==0: idScores2 -dmark;  # Find NaNs
110  idScores2 0 0 idScores2 /mark -delete;                                        # Remove NaNs
111  idScores2 NULL 0 "max" idAux /rec -aggregate; :idAux[0,0]: iH -set max;       # Auto minimum
112  idScores2 NULL 0 "min" idAux /rec -aggregate; :idAux[0,0]: iH -set min;       # Auto maximum
113  20 iH -set bins;                                                              # 20 bins
114  :iH.max+(iH.max-iH.min)/iH.bins: iH -set max;                                 # HACK: adjust right-most bin
115  idScores2 :idLtb[0]: iH -update_i;                                            # Fill histogram
116  iH "" idH -HIST_to_data;                                                      # Get histogram as data
117  idScores2 NULL 0 "mean" idAux /rec -aggregate; :idAux[0,0]: idH -set descr1;  # Store mean as data
         descriptor
```

```
118   idScores2 NULL 0 "sdev" idAux /rec -aggregate; :idAux[0,0]: idH -set descr2;  # Also store standard
          deviation
119   sCname idH -set rtext;                                                        # Store histogram name
120
121   "$[sLogDir]/$[sCname]_histogram.dn3" idH -save;                               # HACK: log histogram
122   idHist NULL != if idH idHist -copy; end                                       # Copy result
123 }
124
125 ## Post-processes a recognition result.
126 ##
127 ## @param itRR fst
128 ##          The recognition result (with grammar).
129 ## @param itRRr fst
130 ##          The reference recognition result (phoneme loop).
131 ## @param idNld data
132 ##          The neg. log. density array of the recognition.
133 ## @param iSi object
134 ##          The recognizer session information.
135 ## @return
136 ##   A data instance containing extactly one record per speech frame. The
137 ##   components are:
138 ##   <table>
139 ##     <tr><th>Name</th><th>Description</th></tr>
140 ##     <tr><td>GMM </th><th>GMM indices of speech recognition</th></tr>
141 ##     <tr><td>GMMr</th><th>GMM indices of reference recognition</th></tr>
142 ##     <tr><td>PHN </th><th>Phoneme indices of speech recognition</th></tr>
143 ##     <tr><td>PHNr</th><th>Phoneme indices of reference recognition</th></tr>
144 ##     <tr><td>NLL </th><th>Neg. log. likelihoods of speech recognition</th></tr>
145 ##     <tr><td>NLLr</th><th>Neg. log. likelihoods of reference recognition</th></tr>
146 ##     <tr><td>PLB </th><th>Phoneme labels of speech recognition</th></tr>
147 ##     <tr><td>PLBr</th><th>Phoneme labels of reference recognition</th></tr>
148 ##   </table>
149 function -RR_postprocess(itRR,itRRr,idNld,iSi)
150 {
151   ## Local functions and instances                                              #
          ------------------------------------
152   function -relabel(itGP,itRes)                                                 # Recover per-frame phoneme
          labels
153   {                                                                             # >>
154     fst itAux;                                                                  #   Auxiliary transducer
155     data idAux;                                                                 #   Auxiliary data
156     itRes itAux -copy;                                                          #   Copy recognition result
157     0 itAux -invert;                                                            #   Invert
158     itAux itGP 0 0 itAux -compose;                                              #   Compose with labeling
            transducer
159     itAux 0 1 -1 itAux -best_n;                                                 #   Best path (if solution
            not unique)
160     :itAux.td["~TOS"]: idAux -copy;                                             #   Copy recovered labels
161     idAux.nrec itRes.td.nrec != if "Re-label: " .S_WRONGNOFRAMES + -ERROR; end  #   Wrong # of recovered
            labels
162     0 "~PHN" idAux -set_cname;                                                  #   Name label component
163     idAux itRes.td -join;                                                       #   Join to recognition
            result
164   }                                                                             # <<
165   function -epsremove(idSrc,idDst)                                              # Removes epsilon transitions
166   {                                                                             # >>
167     # HACK: Inaccurate because epsilon transition weights get lost!             #   #
168     idSrc idDst -copy;                                                          #   Copy source
169     :idDst["~TIS"].<0: idDst -dmark;                                            #   Mark epsilon transitions
170     idDst 0 0 idDst /mark -delete;                                              #   Delete marked records
171   }                                                                             # <<
172   function -reweight(idNld,idRes)                                               # Reweight with GMM neg. log.
          densities
173   {                                                                             # >>
174     data idAux;                                                                 #   Auxiliary data
175     idNld.nrec idRes.nrec != if "Re-weight: " .S_WRONGNOFRAMES + -ERROR; end    #   Check number of records
176     "~RID" -1 idRes -rindex;                                                    #   Add record index
            component
```

```
177     :idRes["~RID"]: 0 :idRes["~TIS"]: 0 idNld idAux -lookup_2;                # Get GMM neg. log.
              densities
178     0 "~NLL" idAux -set_cname; idAux idRes -join;                             # Add to recognition result
179     idRes ( "~RID" idRes -find_comp ) 1 idRes -delete;                        # Delete record index
              component
180   }                                                                           # <<
181   data idRR;                                                                  # Per-frame recognition
            result
182   data idRRr;                                                                 # Per-frame reference recog.
            result
183   data idRes;                                                                 # Post-processed recognition
            result
184   data idAux;                                                                 # Auxiliary data
185
186   ## Recover per-frame phoneme labels of recognition results                 #
            -------------------------------------
187   iSi.itGP itRR  -relabel;                                                    # ... of speech recognition
            result
188   iSi.itGP itRRr -relabel;                                                    # ... of reference
            recognition result
189
190   ## Remove epsilon transitions                                              #
            -------------------------------------
191   itRR.td  idRR  -epsremove;                                                  # ... from speech recognition
             result
192   itRRr.td idRRr -epsremove;                                                  # ... from reference
            recognition result
193
194   ## Recover pure acoustic neg. log. likelihoods                             #
            -------------------------------------
195   ## (i. e. remove the time-invariant HMM, lexicon and grammar weights)      #
196   idNld idRR  -reweight;                                                      # ... of speech recognition
            result
197   idNld idRRr -reweight;                                                      # ... of reference
            recognition result
198
199   ## Collect useful data into recognition result                            #
            -------------------------------------
200   :idRR["~TIS"]:  idRes -copy;                                                # GMM indices of speech recog
            . result
201   :idRRr["~TIS"]: idRes -join;                                                # GMM indices of ref. recog.
            result
202   :idRR["~PHN"]:  idRes -join;                                                # Phoneme indices of speech
            recog. res.
203   :idRRr["~PHN"]: idRes -join;                                                # Phoneme indices of ref.
            recog. res.
204   :idRR["~NLL"]:  idRes -join;                                                # Neg.log.likelihoods of sp.
            rec. res.
205   :idRRr["~NLL"]: idRes -join;                                                # Neg.log.likelihoods of ref.
             rec. res.
206   :idRR["~PHN"]:  0 iSi.itRNr.os 0 1 idAux -lookup; idAux idRes -join;        # Phoneme labels of speech
            recog. rec.
207   :idRRr["~PHN"]: 0 iSi.itRNr.os 0 1 idAux -lookup; idAux idRes -join;        # Phoneme labels of ref.
            recog. rec.
208   { "GMM" "GMMr" "PHN" "PHNr" "NLL" "NLLr" "PLB"  "PLBr" } '                 # |
209     0 idRes -set_cnames;                                                      # Rename result components
210
211   idRes return;                                                               # Return post-processed recog
            . result
212 }
213
214 ## Labels a vector sequence with the recognition result.
215 ##
216 ## @param idRes data
217 ##          The post-processed recognition result as returned by
218 ##          {@link -RR_postprocess}.
219 ## @param idVs data
220 ##          The vector sequence to label (content will be changed).
221 ## @return nothing
```

```
222 function -RR_label(idRes,idVs)
223 {
224   :idRes["PLB"]:  idVs -join;                                          # Phoneme labels of speech
          recognition
225   :idRes["PLBr"]: idVs -join;                                          # Phoneme labels of reference
           recog.
226 }
227
228 ## Baseline confidence/rejection implementation.
229 ##
230 ## @param idRes data
231 ##          The post-processed recognition result as returned by
232 ##          {@link -RR_postprocess}.
233 ## @param iSi object
234 ##          The recognizer session information.
235 ## @global nNadT R
236 ## @global nNhdT R
237 ## @global nNadW R
238 ## @global nNhdW R
239 ## @return
240 ##   The recognizer confidence ranging from -1 for "sure to reject" to 1 for
241 ##   "sure to accept". If a positive value is returned, the recognition result
242 ##   should be accepted.
243 function -RR_confidence_baseline(idRes,iSi)
244 {
245   var  nNad; ( idRes iSi -REJ_CNFS_nad ) nNad =;                       # Normalized acoustic
          distance
246   var  nNhd; ( idRes iSi -REJ_CNFS_nhd ) nNhd =;                       # Normalized Hamming distance
247
248   ( "$[nNad]" "nan" == ) ( "$[nNhd]" "nan" == ) || if -1 return; end   # Nan scores -> to be
          rejected
249   :.nNhdW*max(1-nNhd/.nNhdT,-1)+.nNadW*max(1-nNad/.nNadT,-1): return;  # Compute and return
          confidence
250 }
251
252 ## Improved confidence/rejection implementation.
253 ##
254 ## @param idRes data
255 ##          The post-processed recognition result as returned by
256 ##          {@link -RR_postprocess}.
257 ## @param iSi object
258 ##          The recognizer session information.
259 ## @param idXtra data
260 ##          Extra result data will be joined to this object (can be
261 ##          <code>NULL</code>)
262 ## @return
263 ##   The recognizer confidence ranging from -1 for "sure to reject" to 1 for
264 ##   "sure to accept". If a positive value is returned, the recognition result
265 ##   should be accepted.
266 function -RR_confidence(idRes,iSi,idExtra)
267 {
268   data idS;                                                            # List if confidence scores
269   data idAux;
270   data idAux2;
271   var  nScore;                                                         # Confidence score
272   var  i;                                                              # Loop counter
273   var  m;                                                              # Score mean
274   var  s;                                                              # Score Standard deviation
275   var  t;                                                              # Score threshold
276   var  w;                                                              # Score weight
277   var  wsum;                                                           # Score weight sum
278   var  nConf;                                                          # Confidence
279
280   ## Compute a variety of confidence scores                           #
          ------------------------------------
281   { "nad" "nhd" "pdur" "nwhd" "nwld" } ' idS =;  0 "ID" idS -set_cname;  # List of confidence scores
          ...
282   "RAW" ( -type double ) idS -addcomp;                                 # ...and the respective raw
          ...
```

```
283   "STD" ( -type double ) idS -addcomp;                                    # ...and standardized values
284   0 i =; i idS.nrec < while                                               # Loop over confidence scores
            >>
285     ( idRes iSi -REJ_CNFS_${idS[i,0]} ) nScore =;                         #   Invoke score function
286     ( "rej.${idS[i,0]}.mean"  0 "S" -CFG_get_ex ) m =;                    #   Get score mean
287     ( "rej.${idS[i,0]}.sdev" -1 "S" -CFG_get_ex ) s =;                    #   Get score standard
              deviation
288     "\n     - rej.${idS[i,0]}.mean=$[m], rej.${idS[i,0]}.sdev=$[s],"   3 -MSGX; #   Protocol
289     :idS[i,"RAW"]=nScore; :idS[i,"STD"]=-(nScore-m)/s;                    #   Store score and
              standardized score
290     " ${idS[i,0]}="      ( :idS[i,"RAW"]: 0 2 -FMT_f ) +                -MSG2; #   Protocol
291     ", ${idS[i,0]}.STD=" ( :idS[i,"STD"]: 0 2 -FMT_f ) +               3 -MSGX; #   Protocol
292     i ++=;                                                                #   Next score
293   end                                                                     # <<
294   idExtra NULL != if                                                      # Collecting score histograms
            >>
295     ( 1 idS.nrec zeros ) 0 { "S_" } 0 1 idAux -lookup;                    #   Reshape score matrix -->
              idAux2
296     :idS[0]: idAux -join; idAux "" "ccat" idAux -strop;                   #   ...
297     :idS["STD"]: idAux -join; idS 0 2 idAux2 -select; idAux idAux2 -cat;  #   ...
298     idAux2 ' idExtra -join;                                               #   Join to committed extra
              info object
299   end                                                                     # <<
300
301   ## Compute the confidence                                              #
          -------------------------------------
302   0 i =; i idS.nrec < while                                               # Loop over confidence scores
            >>
303     ( "rej.${idS[i,0]}.thrs" 0 "S" -CFG_get_ex ) t =;                     #   Get score threshold
304     ( "rej.${idS[i,0]}.wght" 0 "S" -CFG_get_ex ) w =;                     #   Get score weight
305     :wsum += w;                                                           #   Summ up weights
306     "\n     - rej.${idS[i,0]}.thrs=$[t], rej.${idS[i,0]}.wght=$[w]"    3 -MSGX; #   Protocol
307     :idS[i,"STD" ]: nScore =; "$[nScore]" "nan" == if -1 return end       #   Reject if any score is
              NaN
308     :nScore = w*min(max(nScore-t,-1),1);                                  #   Compute confidence
              contribution
309     ", ${idS[i,0]}.CONF=" ( nScore 0 2 -FMT_f ) +                      3 -MSGX; #   Protocol
310     :nConf += nScore;                                                     #   Aggregate confidence
311     i ++=;                                                                #   Next score
312   end                                                                     # <<
313   :round(wsum*1000)!=1000: if
314     "Weight sum is " ( wsum 0 3 -FMT_f ) + " (should be 1)" +       1 -WARNING;
315   end
316   nConf return;                                                           # Return confidence
317 }
318
319 ## -- Worker Functions --                                                #
        ====================================
320
321 ## Performs a rejection evaluation.
322 ##
323 ## @param sFlist string
324 ##          File list identifier: "all", "dev" or "test".
325 ##
326 ## @global nRNU                              R
327 ## @global &lt;uasr.am.model&gt;             R
328 ## @global &lt;uasr.dir.log&gt;              R
329 ## @global &lt;uasr.dir.model&gt;            R
330 ## @global &lt;uasr.dir.sig&gt;              R
331 ## @global &lt;uasr.flist.&lt;sFlist&gt;&gt; R
332 ## @return
333 ##   The rejection evaluation result; a data instance containing extactly one
334 ##   record. The components are:
335 ##   <table>
336 ##     <tr><th>Name</th><th>Description</th></tr>
337 ##     <tr><td>TN </th><th>True negative count</th></tr>
338 ##     <tr><td>FN </th><th>False negative count</th></tr>
339 ##     <tr><td>FP </th><th>False positive count</th></tr>
340 ##     <tr><td>TP1</th><th>True positive 1 count (recognition correct)</th></tr>
```

```
##     <tr><td>TP2</th><th>True positive 2 count (recognition wrong)</th></tr>
##   </table>
function -REJ_evl(sFlist)
{
  "\n\n// Rejection evaluation"                                          -MSG; # Screen protocol

  ## Define instances                                                          #
       -------------------------------------
  data      idPfv;                                                           # Primary feature vector
       sequence
  data      idSfv;                                                           # Secondary feature vector
       sequence
  data      idNld;                                                           # GMM neg. log. density array
  data      idRes;                                                           # Recognition result (post-
       processed)
  data      idEvl;                                                           # Evaluation result
  data      idAux;                                                           # Auxiliary data
  data      idAux2;                                                          # Auxiliary data #2
  data      idScores;                                                        # Extra data (confidence
       scores)
  data      idScoresLog;                                                     # List of confidence scores
  fst       itRR;                                                            # Recognition result (with
       grammar)
  fst       itRRr;                                                           # Reference recognition
       result
  fstsearch iRR;                                                             # Speech decoder (with
       grammar)
  fstsearch iRRr;                                                            # Reference decoder phoneme
       loop)
  var       nConf;                                                           # Recognition confidence
  var       sWlb;                                                            # Recognized word label
       string
  var       sWlbr;                                                           # Reference word label string
  var       bOot;                                                            # Recognition result is <OOT>
        or <NSP>
  var       i;                                                               # Auxiliary (integer)
       variable
  var       s;                                                               # Auxiliary (string) variable

  ## Get paths and recognition file list                                      #
       -------------------------------------
  var  sAmMod;  ( "am.model" "3_20" "S" -CFG_get_ex ) sAmMod =;              # Acoustic model identifier
  var  sModDir; ( "model" "" -CFG_get_path  ) sModDir =;                     # Get model folder
  var  sLogDir; ( "log"   "" -CFG_get_path  ) sLogDir =;                     # Get log folder
  var  sSigDir; ( "sig"   "" -CFG_get_path  ) sSigDir =;                     # Get signal folder
  file iF;      ( sFlist  "" -CFG_get_flist ) iF -set flist;                 # Recognition file list
  "\n   - Feature info  : $[sModDir]/feainfo.object"                       -MSG; # Screen protocol
  "\n   - Session info  : $[sModDir]/sesinfo.object"                        -MSG; # Screen protocol
  "\n   - GMMs          : $[sModDir]/$[sAmMod].gmm"                         -MSG; # Screen protocol

  ## Load models                                                              #
       -------------------------------------
  object iFi;   "$[sModDir]/feainfo.object"  iFi  -restore;                  # Feature information
  object iSi;   "$[sModDir]/sesinfo.object"  iSi  -restore;                  # Recognizer session
       information
  gmm    iGmm;  "$[sModDir]/$[sAmMod].gmm"   iGmm -restore;                  # Gaussian mixture models

  ## Initialize decoders                                                      #
       -------------------------------------
  "tp" iRR -set algo; "tp" iRRr -set algo;                                   # Use token passing algorithm
  "t"  iRR -set bt;   "t"  iRRr -set bt;                                     # Output frame labels
  FALSE iRR -set stkprn;

  ## Initialize evaluation result                                             #
       -------------------------------------
  ( -type long ) 5 1 idEvl -array;                                           # Allocate evaluation result
       record
  { "TN" "FN" "FP" "TP1" "TP2" } ' 0 idEvl -set_cnames;                      # Name components

```

```
392   ## Loop over speech files                                                    #
            ------------------------------------
393   "\n\n   Recognizing $[iF.len] turn(s) ..."                                  -MSG; # Screen protocol
394   0 1 -PBAR;                                                                  # Begin progress bar
395   iF -reset; iF -next while                                                   # Get next file from file
            list >>
396     "\n   - $[iF.nfile 1 +]/$[iF.len] $[iF.sfile]: ..."                      -MSG2; #   Screen protocol (verbose
              level 2)
397     :(iF.nfile+1)/iF.len: 1                                                   -PBAR; #   Display progress
398     idScores -reset;                                                            #   Clear extra result data
399     ( 0 1 iF.recfile -fetch ) sWlbr =;                                          #   Get reference word label
              string
400
401     ## - Do speech recognition                                                 #   - - - - - - - - - - - - - -
                - - - - -
402     ( iF NULL "" idPfv -FEA_get ) not if  " SKIP" -MSG2; continue; end          #   Do primary feature
              analysis
403     idPfv iFi iGmm.mean.dim idSfv NULL -FEA_sfa;                        "." -MSG2; #   Do secondary feature
              analysis
404     idSfv NULL idNld iGmm /neglog -density;                             "." -MSG2; #   Compute neg. log.
              densities
405     iSi.itRN  .nRNU idNld itRR  iRR  -search;                           "." -MSG2; #   Recognition with grammar
406     iSi.itRNr 0     idNld itRRr iRRr -search;                           "." -MSG2; #   Reference recognition
407     ( itRR itRRr idNld iSi -RR_postprocess ) idRes =;                   "." -MSG2; #   Post-process recognition
              result
408     idPfv.rinc idRes -set rinc;                                                 #   Copy record increment
              from features
409     :idRes["NLL"]-idRes["NLLr"]: idRes -join;                                   #   Add NLL difference
              component...
410     :idRes.dim-1: "dNLL" idRes -set_cname;                                      #   ... and name it
411
412     ## - Log                                                                   #   - - - - - - - - - - - - - -
                - - - - -
413     "$[sLogDir]/$[iF.sfile]" s =;                                               #   Prefix of log file name
414     #idRes idPfv -RR_label; "$[s].pfv.dn3" idPfv -save;                 "L" -MSG2; #   DEBUG - log primary
              features
415     #idRes idSfv -RR_label; "$[s].sfv.dn3" idSfv -save;                 "L" -MSG2; #   DEBUG - log secondary
              features
416     #idRes idNld -RR_label; "$[s].nld.dn3" idNld -save;                 "L" -MSG2; #   DEBUG - log neg. log.
              densities
417     "$[s].res.dn3" idRes -save;                                        "L" -MSG2; #   DEBUG - recognition
              result
418
419     ## - Rejection evaluation                                                   #   - - - - - - - - - - - - - -
                - - - - -
420     :itRR.td["~TOS"]: 0 iSi.idLMtos 0 1 idAux -lookup;                          #   Get word labels of speech
               recog.
421     idAux " " "rcat" idAux -strop; ( 0 0 idAux -fetch ) sWlb =;                 #   Concat. to recognized
              word string
422     " WLB=\"" sWlb + "\" REF=\"" + sWlbr + "\"" +                           -MSG2; #   Screen protocol
423
424     #( idRes iSi -RR_confidence_baseline ) nConf =;                              #   Baseline confidence
              implementation
425     ( idRes iSi idScores -RR_confidence  ) nConf =;                             #   New confidence
              implementation
426     " CONF=" ( nConf 0 2 -FMT_f ) + -MSG2;                                      #   Protocol
427
428     ## Asses TP, TN, FP, and FN                                                #   - - - - - - - - - - - - - -
                - - - - -
429     ( sWlbr "<OOT>" == ) ( sWlbr "<NSP>" == ) || bOot =;                        #   Result is <OOT> or <NSP>
430     ## --- This -->
431 #    bOot ( sWlbr sWlb != ) || if                                               #   To be rejected >>
432 #      nConf 0 <= if "TN" else "FP" end                                         #     Count TN or FP
433 #    else                                                                       #   << To be accepted >>
434 #      nConf 0 <= if "FN" else "TP1" end                                        #     Count FN or TP
435 #    end s =;                                                                   #   <<
436 #    :idEvl[0,"$[s]"]++;                                          " > $[s]" -MSG2; #   Protocol
437     ## <-- or that -->
438     bOot ( nConf 0 <= ) && if                                                   #   True negative >>
```

```
439       :idEvl[0,"TN"]++;                                          " > TN" -MSG2; #     Count and so screen
              protocol
440     end                                                                        #   <<
441     bOot ( nConf 0 > ) && if                                                   #   False positive >>
442       :idEvl[0,"FP"]++;                                          " > FP" -MSG2; #     Count and so screen
              protocol
443     end                                                                        #   <<
444     bOot not ( nConf 0 > ) && if                                               #   True positive >>
445       sWlbr sWlb == if                                                         #     Recognition correct >>
446         :idEvl[0,"TP1"]++;                                     " > TP1" -MSG2; #       True positive 1
447       else                                                                     #     << Recognition wrong >>
448         :idEvl[0,"TP2"]++;                                     " > TP2" -MSG2; #       True positive 2
449       end                                                                      #     <<
450     end                                                                        #   <<
451     bOot not ( nConf 0 <= ) && if                                              #   False negative >>
452       :idEvl[0,"FN"]++;                                          " > FN" -MSG2; #     Count and so screen
              protocol
453     end                                                                        #   <<
454     ## <--
455
456     ## Classify confidence scores and keep list                                #   - - - - - - - - - - - - - -
              - - - - -
457     sWlbr s =; bOot not if ( sWlbr sWlb == if "COR" else "ERR" end ) s =; end  #   Make histogram class name
458     { s } idAux =; 0 "type" idAux -set_cname; idAux "<>" "trim" idAux -strop;   #   Trim '<' and '>'
459     idAux idScores -join; idScores idScoresLog -cat;                           #   Label scores and keep
              scores list
460
461   end                                                                          # <<
462   "\n   " -MSG2; "done\n"                                          -MSG; # Finish progress bar
463
464   ## Aftermath (Plot classified confidence score histograms)                  #
          ------------------------------------
465   0 i =; i idScoresLog.dim < while                                             # Loop over score components
          >>
466     ( i idScoresLog -get_comp_type ) 255 <= if i ++=; continue; end            #   Skip string components
467     idScoresLog ( i idScoresLog -get_cname ) idAux -CONF_histogram;            #   Compute (and log) score
              histogram
468     { idAux.rtext idAux.descr1 idAux.descr2 } idAux2 -cat;                     #   Store name, mean and std.
              dev.
469     i ++=;                                                                     #   Next component
470   end                                                                          # <<
471   { "ID" "MEAN" "SDEV" } ' 0 idAux2 -set_cnames;                               # Name components
472
473   ## Protocol                                                                 #
          -------------------------------------
474   "\n   Score Standardization Data"                                  -MSG; # Protocol
475   "\n   -------------------------"                                   -MSG; # Protocol
476   "\n   ID           Mean    SDev."                                  -MSG; # Protocol
477   0 i =; i idAux2.nrec < while                                                 # Loop over records >>
478     "\n   - " ( :idAux2[i,0]: -6 -MSG_pad ) + ":" +                    -MSG; #   Protocol
479     ( :idAux2[i,1]: 8 3 -FMT_f ) ( :idAux2[i,2]: 8 3 -FMT_f ) +          -MSG; #   ...
480     i ++=;                                                                     #   Next record
481   end;                                                                         # <<
482   "\n   -------------------------\n"                                  -MSG; # Protocol
483
484   idEvl return;                                                                # Return evaluation result
485 }
486
487 ## -- Confidence Score Functions --                                           #
          ====================================
488
489 ## Computes the normalized acoustic distance (NAD) score of a recognition
490 ## result.
491 ##
492 ## @param idRes data
493 ##          The post-processed recognition result as returned by
494 ##          {@link -RR_postprocess}.
495 ## @param iSi object
496 ##          The recognizer session information.
```

```
497 ## @return The NAD score.
498 function -REJ_CNFS_nad(idRes,iSi)
499 {
500   data idRes2;                                                          # Copy of recognition result
501   data idAux;                                                           # Auxiliary data
502   data idAux2;                                                          # Auxiliary data 2
503   var  nNad;                                                            # Normalized acoustic
          distance
504
505   idRes idRes2 -copy;                                                   # Copy recognition result
506   :(idRes2["PHN"].==.nSilMod).||(idRes2["PHNr"].==.nSilMod): idRes2 -dmark;     # Mark pause labels
507   idRes2 0 0 idRes2 /mark -delete;                                      # Delete
508   :(idRes2["PHN"].==.nGbgMod).||(idRes2["PHNr"].==.nGbgMod): idRes2 -dmark;     # Mark garbage labels
509   idRes2 0 0 idRes2 /mark -delete;                                      # Delete
510   idRes2 -is_empty if NAN return; end                                   # No speech frames left ->
          reject
511
512   :abs(idRes2["NLL"]-idRes2["NLLr"]): NULL 0 "sum" idAux  /rec -aggregate;      # Sum absolute NLL diff's. of
           frames
513   :idRes2["NLL"]:                    NULL 0 "sum" idAux2 /rec -aggregate;      # Sum NLLs of speech
          recognition res.
514   :nNad = idAux[0,0]/idAux2[0,0];                                       # Compute normalized acoustic
           distance
515   nNad return;                                                          # Return NAD score
516 }
517
518 ## Computes the normalized Hamming distance (formerly normalized edit distance,
519 ## NED) score of a recognition result.
520 ##
521 ## @param idRes data
522 ##          The post-processed recognition result as returned by
523 ##          {@link -RR_postprocess}.
524 ## @param iSi object
525 ##          The recognizer session information.
526 ## @return The NHD score.
527 function -REJ_CNFS_nhd(idRes,iSi)
528 {
529   data idRes2;                                                          # Copy of recognition result
530   data idAux;                                                           # Auxiliary data
531   data idAux2;                                                          # Auxiliary data 2
532   var  nNhd;                                                            # Normalized acoustic
          distance
533
534   idRes idRes2 -copy;                                                   # Copy recognition result
535   :(idRes2["PHN"].==.nSilMod).||(idRes2["PHNr"].==.nSilMod): idRes2 -dmark;     # Mark pause labels
536   idRes2 0 0 idRes2 /mark -delete;                                      # Delete
537   :(idRes2["PHN"].==.nGbgMod).||(idRes2["PHNr"].==.nGbgMod): idRes2 -dmark;     # Mark garbage labels
538   idRes2 0 0 idRes2 /mark -delete;                                      # Delete
539   idRes2 -is_empty if NAN return; end                                   # No speech frames left ->
          reject
540
541   :idRes2["PHN"].!=idRes2["PHNr"]: NULL 0 "sum" idAux /rec -aggregate;          # Sum Hamming distance of
          frame labels
542   :nNhd = idAux[0,0]/idRes2.nrec;                                       # Compute normalzed edit
          distance
543   nNhd return;                                                          # Return NED score
544 }
545
546 ## Computes the negative phoneme duration likelihood score of a recognition
547 ## result.
548 ##
549 ## @param idRes data
550 ##          The post-processed recognition result as returned by
551 ##          {@link -RR_postprocess}.
552 ## @param iSi object
553 ##          The recognizer session information.
554 ## @global idPhnDUR CRW
555 ## @global sModDir  R
556 ## @return The negative phoneme duration likelihood score.
```

```
557 function -REJ_CNFS_pdur(idRes,iSi)
558 {
559   data idDur;
560   data idAux;
561
562   ".idPhnDur" "data" ?instance not if                                    # No phoneme duration
          probabilities >>
563     data .idPhnDur;                                                     #   Persistently create it
564     "$[.sModDir]/phon_dur_probs.dn3" .idPhnDur -restore;                #   Load probability
            functions
565     :idPhnDur=-.ln(idPhnDur);                                           #   Compute neg. log.
            probabilities
566   end                                                                   # <<
567
568   :idRes["PHN"]: 0 idDur -compress;                                     # Get recognized phoneme
          durations
569   idDur 1 1 idDur -delete; 1 "DUR" idDur -set_cname;                    # ...
570   :(idDur["PHN"].==.nSilMod).||(idDur["PHN"].==.nGbgMod): idDur -dmark; # Remove silence and garbage
571   idDur 0 0 idDur /mark -delete;                                        # ...
572
573   :idDur["DUR"]: 0 :idDur["PHN"]: 0 .idPhnDur idAux -lookup_2;          # Lookup single phoneme
          duration NLLs
574   idAux NULL 0 "mean" idAux /rec -aggregate;                            # Compute mean
575   :idAux[0,0]: return;                                                  # Return result
576 }
577
578 ## Computes the normalized weighted Levenshtein divergence between the
579 ## phonetic recognition and the reference recognition results. Convenience
580 ## function. Invokes <code>idRes iSi "L" -REJ_CNFS_nwpd_int</code>.
581 ##
582 ## @param idRes data
583 ##          The post-processed recognition result as returned by
584 ##          {@link -RR_postprocess}.
585 ## @param iSi object
586 ##          The recognizer session information.
587 ## @see -REJ_CNFS_nwpd_int
588 function -REJ_CNFS_nwld(idRes,iSi)
589 {
590   ( idRes iSi "L" -REJ_CNFS_nwpd_int ) return;
591 }
592
593 ## Computes the normalized weighted Hamming divergence between the phonetic
594 ## recognition and the reference recognition results. Convenience function;
595 ## invokes <code>idRes iSi "H" -REJ_CNFS_nwpd_int</code>.
596 ##
597 ## @param idRes data
598 ##          The post-processed recognition result as returned by
599 ##          {@link -RR_postprocess}.
600 ## @param iSi object
601 ##          The recognizer session information.
602 ## @see -REJ_CNFS_nwpd_int
603 function -REJ_CNFS_nwhd(idRes,iSi)
604 {
605   ( idRes iSi "H" -REJ_CNFS_nwpd_int ) return;
606 }
607
608 ## Computes the normalized weighted phonetic divergence between the recognition
609 ## and the reference recognition results. As the negative logarithmic phoneme
610 ## confusion probabilities are used as weights, the function is not a metric
611 ## (the identity and symmetry axioms do not hold).
612 ##
613 ## @param idRes data
614 ##          The post-processed recognition result as returned by
615 ##          {@link -RR_postprocess}.
616 ## @param iSi object
617 ##          The recognizer session information.
618 ## @param sMode string
619 ##          The operation mode: "H" Hamming divergence, "L" Levenshtein
620 ##          divergence.
```

```
621 ## @global idPhnCmx CRW
622 ## @global sModDir  R
623 ## @return The negative phoneme duration likelihood score.
624 function -REJ_CNFS_nwpd_int(idRes,iSi,sMode)
625 {
626   data idCmx;
627   fst  itAux;
628   data idAux;                                                    # Auxiliary data
629   data idAux2;                                                   # Auxiliary data #2
630
631   ".idPhnCmx" "data" ?instance not if                            # No phoneme confusion matrix
          >>
632     data .idPhnCmx;                                              #   Persistently create it
633     var sAmMod;  ( "am.model" "3_20" "S" -CFG_get_ex ) sAmMod  =;#   Acoustic model identifier
634     "$[.sModDir]/$[sAmMod]_hmm-frm-cmx.dn3" .idPhnCmx -restore;  #   Load per-frame phoneme
            conf. matrix
635     :.idPhnCmx += 1;                                             #   Add-one smoothing
636     .idPhnCmx NULL 0 "sum" idAux -aggregate;                     #   Aggregate smoothed counts
637     :.idPhnCmx = -.ln(.idPhnCmx/idAux);                          #   Normalize and log
638   end                                                            # <<
639
640   sMode "H" == if                                                # Hamming mode >>
641     idRes idCmx -copy;                                           #   Copy recognition result
642     :(idCmx["PHN"].==.nSilMod).||(idCmx["PHNr"].==.nSilMod): idCmx -dmark;#   Mark pause labels
643     idCmx 0 0 idCmx /mark -delete;                               #   Delete
644     :(idCmx["PHN"].==.nGbgMod).||(idCmx["PHNr"].==.nGbgMod): idCmx -dmark;#   Mark garbage labels
645     idCmx 0 0 idCmx /mark -delete;                               #   Delete
646     idCmx -is_empty if NAN return; end                           #   No speech frames left ->
            reject
647   else sMode "L" == if                                           # << Levenshtein mode >>
648     :idRes["PLB"]:  0  idAux  -compress; idAux  "" "rcat" idAux  -strop;#   Get recognized phoneme
            string
649     :idRes["PLBr"]: 0  idAux2 -compress; idAux2 "" "rcat" idAux2 -strop;#   Get reference phoneme
            string
650     idAux2 idAux -cat; idAux 0 1 idAux -select;                  #   Collect phoneme strings
651     idAux 0 -1 :.idPhnCmx[.idPhnCmx.dim-1]: itAux -compile;      #   Compile phoneme strings
652     itAux itAux 0 1 NULL NULL NULL itAux -FST_lvnstn_ex;         #   Best path (eps -> last
            symbol!)
653     itAux.td 2 2 idCmx -select; { "PLB" "PLBr" } ' 0 idCmx -set_cnames;#   Copy and name aligned
            sequences
654   else                                                           # << Unknown mode >>
655     "sMode=\"$[sMode]\" invalid in function -CONF_nwld" 1 -WARNING;#   Warning
656     NAN return;                                                  #   All we can do...
657   end end                                                        # <<
658
659   :idCmx["PHN"]: 0 :idCmx["PHNr"]: 0 .idPhnCmx idAux -lookup_2;  # Lookup single phoneme
          confusion NLLs
660   idAux NULL 0 "mean" idAux /rec -aggregate;                     # Compute mean
661   :idAux[0,0]: return;                                           # Return result
662 }
663
664 ## -- MAIN PROGRAM --                                           #
        ====================================
665
666 ## Magic constants                                             #
       --------------------------------------
667 var sCfgFile; "$1"      sCfgFile =;                            # The UASR configuration file
668 var nRNU;     1         nRNU     =;                            # Recognition network unit to
         use
669 var nNadT;    0.06      nNadT    =;                            # Norm. acoustic distance
       threshold
670 var nNhdT;    0.45      nNhdT    =;                            # Norm. edit distance
       threshold
671 var nNadW;    1.0       nNadW    =;                            # NAD interpolation weight
       1)
672 var nNhdW;    0.0       nNhdW    =;                            # NED interpolation weight
       1)
673                                                                 # 1) weights must sum up to
                                                                        1!
```

```
674 ## Initialize UASR session                                                    #
          ------------------------------------
675 "\n// $__SFILE__.xtp revision " -UASR_version + "/" + -version +          -MSG; # Screen protocol
676 "\n// Process         : $HOSTNAME/" -pid +                                  -MSG; # Screen protocol
677 "\n// Configuration   : $[sCfgFile]"                                       -MSG; # Screen protocol
678 /disarm -SIG_get_rr /disarm -SIG_get =;                                         # Overload signal import
          function
679 sCfgFile TRUE -CFG_init;                                                    # Configure session
680 "$UASR_HOME/scripts/dlabpro/util/uasr_session.itp" include;                  # Include UASR session
          startup script
681 "\n\n// OOT REJECTION TEST PROGRAM"                                        -MSG; # Screen protocol
682
683 ## Global variables                                                         #
          ------------------------------------
684 var .nSilMod; ( "am.sil" -1 "S" -CFG_get_ex   ) .nSilMod =;                 # Silence model (HMA ID)
685 var .nGbgMod; ( "am.gbg" -1 "S" -CFG_get_ex   ) .nGbgMod =;                 # Garbage model (HMA ID)
686 var .sModDir; ( "model" "S"     -CFG_get_path ) .sModDir =;                 # Model directory
687
688 ## Do rejection evaluation                                                  #
          ------------------------------------
689 data idEvl;                                                                 # Rejection evaluation result
690 ( "rej" -REJ_evl ) idEvl =;                                                 # Invoke rejection evaluation
691
692 ## Pretty-printing of results                                              #
          ------------------------------------
693 "\n   Evaluation Results:"                                                  -MSG; # Protocol
694 "N"   "TN+FN+FP+TP1+TP2"    NULL idEvl -print_quota;                        # Protocol
695 "TN"  "TN"                  NULL idEvl -print_quota;                        # Protocol
696 "FN"  "FN"                  NULL idEvl -print_quota;                        # Protocol
697 "FP"  "FP"                  NULL idEvl -print_quota;                        # Protocol
698 "TP1" "TP1"                 NULL idEvl -print_quota;                        # Protocol
699 "TP2" "TP2"                 NULL idEvl -print_quota;                        # Protocol
700 "ERR" "FP+FN" "TN+FN+FP+TP1+TP2" idEvl -print_quota;                        # Protocol
701 "FAR" "FP"              "FP+TN" idEvl -print_quota;                        # Protocol
702 "FRR" "FN"          "TP1+TP2+FN" idEvl -print_quota;                        # Protocol
703
704 "\n\n// $__SFILE__.xtp completed (${.__UTL_nErrors} errors)."             -MSG; # Protocol
705 quit                                                                        # Terminate dLabPro
706
707 ## EOF
```

F Inhalt der DVD

Auf der dieser Arbeit beigelegten DVD finden sich sämtliche Daten um die dargelegten Resultate zu reproduzieren. Die DVD ist in folgende Ordner eingeteilt:

Arbeitsumgebung enthält alle nötigen Daten um dLabPro und UASR zur Evaluation der Rückweisung in Betrieb zu nehmen.

flists enthält die einzelnen Listen von Spracheingaben pro Proband, und eingeteilt in Entwicklungs- und Teststichprobe.

Grammatik enthält die Beschreibung der aus dem Wizard-of-Oz-Experiment entstandenen neuen Grammatik, sowie deren Graphen.

PDF enthält diese Arbeit im PDF-Format, sowie die Foliensätze zu Zwischenverteidigung und Verteidigung.

scripts enthält alle `bash`-Scripte zur Kalibrierung des Systems und zur Evaluation der einzelnen Konfidenzmerkmale.

Soundfiles enthält alle Tonaufzeichnungen die im Zuge des Wizard-of-Oz-Experiments erstellt wurden. Diese Tonaufnahmen bilden den verwendeten Testkorpus.

F.1 Einrichtung dLabPro und UASR

Die auf dieser DVD bereitgestellten Versionen von dLabPro und UASR sind getestet unter einem 64-bit Linux-System und auf einem solchen lauffähig.

Um die Entwicklungs- und Evaluationsumgebung in Betrieb zu nehmen müssen lediglich die beiden Ordner `dlabpro` und `uasr` aus dem Verzeichnis `Arbeitsumgebung` von der DVD in das `$HOME`-Verzeichnis des Nutzers kopiert werden. Danach muss der Inhalt der Datei `bash rc additions` der Datei `.bashrc` im Homeverzeichnis des Nutzers hinzugefügt werden. Dies fügt dem System die nötigen Umgebungsvariablen und Konfigurationen hinzu um dLabPro und UASR nutzen zu können.

F.2 Installation Testkorpus

Um den in dieser Arbeit verwendeten Testkorpus in das System zu integrieren, müssen verschiedene Daten von der DVD in das UASR-Verzeichnis kopiert werden. Der Inhalt des Ordners `flists` muss hierbei in den Ordner `$HOME/uasr/_work/rejection/data/rej/common/flists` kopiert werden. Der Inhalt des Ordners `Soundfiles` wird nach `$HOME/uasr/_work/rejection/data/rej/co` kopiert. Nun kann das System mit der richtigen Konfiguration auf dem Testkorpus arbeiten.

F.3 Nutzung des eingerichteten Systems

Im Ordner `scripts` auf der DVD finden sich in den einzelnen Unterordnern verschiedene Scripte um die Arbeit mit dem wie in Abschnitt F.1 und F.2 beschrieben eingerichteten System zu erleichtern.

Mit den Scripten aus dem Unterordner `calibration` kann die Standardisierung der einzelnen Merkmale kalibriert werden. Hierfür müssen die entsprechenden Scripte lediglich nach `$HOME/uasr/_work/rejection` kopiert werden und dort ausgeführt werden. Die Ergebnisse der Evaluation werden in entsprechende Log-Files geschrieben, und können später ausgewertet werden.

Das folgende Codebeispiel zeigt den Code zum Kalibrieren der Standardisierung des Merkmals NAD.

```
1  #!/bin/bash
2
3  #for i in 0.0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95 1.0
4  #do
5  date >> nadlog.txt
6
7      weight1=1.0
8      for j in 0.0585 0.059 0.0595 0.06 0.0605 0.061 0.0615
9      do
10         for l in 0.0535 0.054 0.0545 0.055 0.0555 0.056 0.0565
11             do
12             echo "nad.mean = $j, nad.sdev = $l, nad.wght = $weight1" >> nadlog.txt
13             dlabpro REJ.xtp REJ.cfg -Prej.nad.dev=$j -Prej.nad.sdev=$l -Prej.nad.wght=$weight1 >> nadlog.txt
14         done
15     done
16
17 date >> nadlog.txt
18 #done
```

Zeile 8 und 10 beschreiben die Werte auf denen gerechnet werden soll. Hier können beliebige Werte als Parameter eingetragen werden um einen zu untersuchenden Teilraum zu bestimmen. Das Symbol » leitet die Ausgabe der verschiedenen Aufrufe in die dahinter spezifizierte Datei um. Hier kann eine beliebige Datei angegeben werden. Existiert die Datei noch nicht, so

wird sie automatisch angelegt. Falls die Datei schon existiert, werden die neuen Einträge angehängt.

Die Scripte in den Ordnern `evaluation`, `evaluation dual` und `evaluation single` arbeiten nach dem gleichen Prinzip wie die zur Kalibrierung. Mit den enthaltenen for-Schleifen wird der Suchraum begrenzt und mit dem Aufruf von dLabPro mit entsprechenden Parametern eine dLabPro-Instanz erzeugt, die mit den übergebenen Parametern arbeitet. Je nach Anzahl der vorhandenen Recheneinheiten, kann das starten von mehreren Scripten gleichzeitig eine bessere Auslastung des Systems schaffen.

Die Scripte im Unterordner `log parse` helfen bei der Weiterverarbeitung der Ergebnisse. Mit dem Script `getres.sh` können die Resultatzeilen aus einem Log eines Evaluationsscriptes extrahiert werden. Das zu bearbeitende Log wird als Parameter übergeben, zum Beispiel `./getres.sh log.txt`. Das Script `sort.sh` sortiert die einzelnen Parameter in einer Ergebniszeile und erzeugt eine Ausgabedatei welche als CSV in Tabellenkalkulationsprogramme importiert werden kann.

G Abkürzungsverzeichnis

BTU Brandenburgische Technische Universität

TUD Technische Universität Dresden

HMM Hidden Markov Modell

UASR Unified Approach to Speech Synthesis and Recognition

TP true positive

TN true negative

FP false positive

FN false negative

KFR Konfidenzfehlerrate

FAR False-Acceptance-Rate

FRR False-Rejection-Rate

EER Equal-Error-Rate

DET Detection-Error-Tradeoff

NAD Normalized Acoustic Distance

NHD Normalized Hamming Distance

NWHD Normalized weighted Hamming Divergence

NWLD Normalized weighted Levenshtein Divergence

PDL Phone Duration Likelihood

PDLW Weighted Phone Duration Likelihood

Literaturverzeichnis

[1] Jiang, Hui: *Confidence measures for speech recognition: A survey.* Speech communication, 45(4):455–470, 2005.

[2] Berton, André: *Konfidenzmaße und deren Anwendungen in der automatischen Sprachverarbeitung.* Web-Univ.-Verlag, 2004.

[3] Kemp, Thomas und Thomas Schaaf: *Estimating confidence using word lattices.* In: *Proc. Eurospeech*, Band 2, Seiten 827–830. Rhodes, Greece: ESCA, 1997.

[4] Goronzy, Silke: *Robust adaptation to non-native accents in automatic speech recognition*, Band 2560. Springer, 2002.

[5] Wessel, Frank, Klaus Macherey und Ralf Schluter: *Using word probabilities as confidence measures.* In: *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, Band 1, Seiten 225–228. IEEE, 1998.

[6] Wessel, Frank, Ralf Schluter, Klaus Macherey und Hermann Ney: *Confidence measures for large vocabulary continuous speech recognition.* Speech and Audio Processing, IEEE Transactions on, 9(3):288–298, 2001.

[7] Kelley, John Falk: *Natural Language and computers: Six empirical steps for writing an easy-to-use computer application.* Doktorarbeit, Johns Hopkins University, 1983.

[8] Pfister, Beat und Tobias Kaufmann: *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung.* Springer DE, 2008.

[9] Wolff, Matthias: *Akustische Mustererkennung.* TUDpress, 2011.

[10] Wolff, Matthias, Constanze Tschöpe, Ronald Römer und Günther Wirsching: *Subsymbol-Symbol-Transduktoren.* In: *Elektronische Sprachsignalverarbeitung 2013, Studientexte zur Sprachkommunikation, Band 65*, Seiten 197–204. TUDpress, 2013.

[11] Rinne, Horst: *Taschenbuch der Statistik*. Harri Deutsch Verlag, 2008.

[12] Wolff, Matthias: *UASR - Recognition Confidence*. `https://www.tu-cottbus.de/kommunikationstechnik/wiki/index.php/UASR_-_Recognition_Confidence`, last visited: February 2014.

[13] Jurafsky, Daniel und H James: *Speech and language processing an introduction to natural language processing, computational linguistics, and speech*. 2000.