Visualisierung kontinuierlicher, multimodaler Schmerz Scors am Beispiel akustischer Signale

Masterarbeit

Franz Anders
HTWK Leipzig

Januar 2017

Abstract

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

8

2 Grundlagen der medizinischen Schrei-Forschung

2.1 Schmerz Scores

Bei erwachsenen Menschen wird der Schmerzgrad typischerweise durch eine Selbsteinschätzung des Patienten unter der Leitung gezielter Fragen des Arztes vorgenommen. Bei Kindern unter 3 Jahren ist diese Selbsteinschätzung nicht möglich. Schmerz drückt sich in Veränderungen des psychologischen, körperlichen und biochemischen Verhaltens des Säuglings aus. Die für den Arzt am leichtesten feststellbaren Verhaltensänderungen sind von außen wahrnehmbaren Merkmale, wie zum Beispiel ein Verkrampfen des Gesichtsausdrucks, erhöhte Körperbewegungen oder lang anhaltendes Weinen. Um eine weitesgehend objektive Schmerzfeststellung zu ermöglichen, wurden sogenannte *Pain-Scores* entwickelt, die durch ein Punktesystem den insgesamten Schmerzgrad des Babies quantifizieren.[22] Es existieren *eindimensionale* Pain-Scores, die den Schmerz nur Aufgrund der Beobachtung eines Merkmals beurteilen, so wie beispielsweise die reine Beurteilung des Gesichtsausdruckes. *Mehrdimensionale* (auch *multimodale*) Pain-Scores beziehen mehrere Faktoren in das Scoring mit ein.[1]. Tabelle 2.1 zeigt das Scoring-System „Neonatal Infant Pain Scale“(NIPS) als Beispiel für eine multimodale Pain-Score. Der Säugling wird anhand der aufgeführten Kategorien bewertet und alle vergebenen Punkte aufsummiert. Ein insgesamter Wert von > 3 zeigt Schmerz an, ein Wert von > 4 großen Schmerz.[10]

Tabelle 2.1: NIPS-Scoring

NIPS	**0 points**	**1 point**	**2 points**
Facial Expr.	Relaxed	Contracted	-
Cry	Absent	Mumbling	Vigorous
Breathing	Relaxed	Different than basal	-
Arms	Relaxed	flexed/stretched	-
Legs	Relaxed	flexed/stretched	-
Alertness	Sleeping	uncomfortable	-

In den meisten mehrdimensionalen Scoring-Systeme werden die Schreigeräusche mit einbezogen. Tabelle 2.2 zeigt eine Übersicht über eine ausgewählte Menge an multimodalen Pain-Scores. Alle Pain-Scores sind für Kleinkinder bis 3 Jahren gedacht. In der Übersicht wird nicht wiedergegeben, welche weiteren Merkmale jeweils in das Scoring mit einbezogen werden, oder welche Insgesamtpunktzahlen auf welche Schmerzintensität hinweisen. Es soll an dieser Stelle nur verdeutlicht werden, welche unterschiedlichen Ansätze zur Bewertung des Schreiens aus medizinischer Sicht im Zusammenhang mit Pain-Scores existieren. Folgende Beobachtungen lassen sich aus der Übersicht ziehen:

1. Die zu beobachtenden Eigenschaften des Weinens werden mit subjektiv behafteten Werten charakterisiert. Beispielsweise wird im N-PASS-System ist ein Schmerz-Schrei

als „High-pitched or silent-continuous crying“ beschrieben. Es wird nicht fest definiert, was als „crying“ gilt oder welche Tonhöhe als „high-pitched“ ist. Auch die Erstquellen geben keine festen Definitionen.

2. Es gibt verschiedene Ansätze zur Bewertung des Weinens. Bei CRIE ist die Tonhöhe, bei BIIP die Länge und bei COMFORT die Art des Weinens entscheidend.

3. Die Beschreibungen sind kurz und prägnant gehalten, der Arzt hat in keinem der Modelle auf mehr als drei Parameter des Schreiens zu achten. Die Begründung liegt darin, dass bei allen Modellen a.) das Schreien nur eines von mehreren Faktoren ist, und b.) Die Schmerzbestimmung in einem vorgegebenen Zeitrahmen durchführbare sein muss.

System	**P.**	**Description**
FLACC[28]	0	No cry (awake or asleep)
	1	Moans or whimpers; occasional complaint
	2	Crying steadily, screams or sobs, frequent complaints
N-PASS[23]	-2	No cry with painful stimul
	-1	Moans or cries minimally with painful stimuli
	0	Appropiate Crying
	1	Irritable or Crying at Intervals. Consolable
	2	High-pitched or silent-continuous crying. Not consolable
BIIP[8]	0	No Crying
	1	Crying <2 minutes
	2	Crying >2 minutes
	3	Shrill Crying >2 minutes
CRIES[3]	0	If no cry or cry which is not high pitched
	1	If cry high pitched but baby is easily consoled
	2	If cry is high pitched and baby is inconsolable
COVERS[13]	0	No Cry
	1	High-Pitched or visibly crying
	2	Inconsolable or difficult to soothe
PAT[11]	0	No Cry
	1	Cry
DAN[4]	0	Moans Briefly
	1	Intermittent Crying
	2	Long-Lasting Crying, Continuous howl
COMFORT[19]	0	No crying
	1	Sobbing or gasping
	2	Moaning

	3	Crying
	4	Screaming
MBPS[21]	0	Laughing or giggling
	1	Not Crying
	2	Moaning quiet vocalizing gentle or whimpering cry
	3	Full lunged cry or sobbing
	4	Full lunged cry more than baseline cry

Tabelle 2.2: Übersicht über Pain-Scores

2.2 Schmerz-Schrei aus medizinischer Sicht

Die Frage ist: Woher kommen diese unterschiedlichen Bewertungen des Weinens in Tabelle 2.2? Gibt es eine Pain-Score, die aus wissenchaftlicher Sicht „recht hat“? Dieser Fragestellung unterliegen unterliegen zwei grundlegendere Fragen: 1.) Ist es überhaupt möglich, anhand der akustischen Eigenschaften den Grund für den Schrei abzuleiten, also beispielsweise Hunger, Einsamkeit oder Schmerz? Anders formuliert: Gibt es überhaupt so etwas wie einen Schmerz-Schrei? 2.) Ist es möglich, anhand der akustischen Eigenschaften den Schweregrad des Unwohlseins abzuleiten (also beispielsweise den Grad des Schrei-Versursachenden Schmerzes)?

Die Annahme, dass es möglich ist, aus dem Schreien den Grund abzuleiten, wird als „Cry-Types Hypothesis“ bezeichnet. Die berühmtesten Befürworter dieser Hypothese ist eine skandinavische Forschungsgruppe, auch bezeichnet als „Scandinavian Cry-Group“, die diese Idee in dem Buch „Infant Crying: Theoretical and Research Perspectives“[2] publik machte. Die Annahme ist, dass die verschiedenen Ursachen *Hunger, Freude, Schmerz, Geburt und Anderes* klare Unterschiede hinsichtlich ihrer akustischen Merkmale aufweisen, welche an einem Spektogramm ablesbar seien. Entsprechende Beispiele werden in dem Buch gegeben. Nur einige Jahre Später zeigte Müller et al [6] in einem Paper, dass bei leichter Veränderung der Bedingungen der Experimente die Unterscheidung nicht möglich ist. Die Gegenhypothese ist, dass Weinen „nichts als undifferenziertes Rauschen“ sei. 50 Jahre später liegt kein anerkannter Beweis für die eine oder andere Hypothese vor. Es gibt nur starke Hinweise dafür, dass die Plötzlichkeit des Eintretens des Schreigrundes hörbar ist. Ein plötzliches Ereignis, wie ein Nadelstich oder ein lautes Geräuch, führen auch zu einem plötzlich beginnenden Schreien. Ein langsam einretendes Ereignis, wie ein langsam immer stärker werdender physischer Schmerz oder langsam eintretender Hunger führen auch zu einem langsam eintretenden Weinen. Da keine Einigung herrscht, wird empfohlen, den Grund aus dem Kontext abzuleiten.[27]

Die Zweite Frage nach der Ableitung der Stärke des Unwohlseins aus den akustischen Eigenschaften des Geschreis wird in der Fachsprache unter dem Begriff *Cry as a graded Signal* subsumiert. Je „stärker“ das Weinen, desto höher das Unwohlsein (*Level of Distress (LoD)*) des Säuglings. Tatsächlich bemessen wird dabei der von dem Beobachter vermutete Grad des Unwohlsein des Babies, und nicht der tatsächliche Grad, da dieser ohne die Möglichkeit der direkten Befragung des Kindes nie mit absoluter Sicherheit bestimmt werden kann. Dieser vermutete LoD wird entweder durch das subjektive Empfinden der Beobachter oder durch Pain-Scores festgestellt. Ein hohes Level of Distress hat vor allem

eine schnelle Reaktion der Aufsichtspersonen zur Beruhigung des Babies zur Folge, womit dem Geschrei eine Art Alarm-Funktion zukommt. Es gibt starke Hinweise darauf, dass das Level of Distress anhand objektiv messbarer Eigenschaften des Audiosignals bestimmt werden kann. So herrscht beispielsweise weitesgehend Einigung darüber, dass ein „lang“ anhaltendes Geschrei auf einen hohen Level of Distress hinweist. Insofern aus dem Kontext des Schreiens Schmerz als wahrscheinlichste Ursache eingegrenzt werden kann, kann aus einem hohen Level of Distress ein hoher Schmerz abgeleitet werden. [27] und [26]

Es herrscht wiederum keine Einigung darüber, welche akustischen Eigenschaften im Detail ein hohes Level of Distress anzeigen. Carlo V Bellieni et al [4] haben festgestellt, dass bei sehr hohem Schmerz in Bezug auf die DAN-Scala (siehe Tabelle 2.2) die Tonhöhe des Geschreis steigt. Qiaobing Xie et al [26] haben festgestellt, dass häufiges und „verzerrtes“ Schreien (ohne feststellbares Grundfrequenz, da der Ton stimmlos erzeugt wird) auf einen hohen Level of Distress hinweist.[27] Diese Uneinigkeit hat wahrscheinlich zu den verschiedenen Bewertungen in den Pain-Scores geführt. 2.2.

2.3 Phyisio-Akustische Modellierung des Weinens

Das Ziel dieses Kapitels ist die Schaffung eines einheitlichen Vokabulares, auf den sich bezogen wird, um das Schreien eines Babys zu beschreiben. Die hier vorgestellten Begriffe stammen sowohl aus dem Buch „A Physioacoustic Model of the Infant Cry “ H Golub und M Corwin [12] als auch aus dem Paper „Rythmic organization of the Sound of Infant Cry “ von Zeskind et al.[24]

Die Lautäußerung eines Neugeborenen, umgangssprachlich auch als „Weinen“ oder „Schreien“ bezeichnet, lässt sich im allgemeinen beschreiben als das „rythmische Wiederholen eines beim ausatmen erzeugen Geräusches, einer kurzen Pause, einem Einatmungs-Geräusch, einer zweiten Pause, und dem erneuten Beginnen des Ausatmungs-Geräusches.“[33].

Das Vokabular, welches insbesondere von H Golub und M Corwin geschaffen wurde, ist sehr umfassend. An dieser Stelle wird eine Auswahl grundlegender Begrifflichkeiten vorgestellt, die in dieser Arbeit gebraucht werden. Sie werden in Abbildung 2.1 veranschaulicht.

Expiration beschreibt den Klang, der bei einem einzelnen, ununterbrochenem Ausatmen mit Aktivierung der Stimmbänder durch das Baby erzeugt wird. [24]. Der von Golub et al [12] verwendete Begriff **Cry-Unit** wird in dieser Arbeit synonym verwendet. Umgangssprachlich ist handelt es sich um einen einzelnen, ununterbrochenen *Schrei*.

Inspiration beschreibt den Klang, der beim Einatmen durch das Baby erzeugt wird.

Burst beshreibt die Einheit von einer Expiration und der darauf folgenden Inspiration. Das heisst, dass die zeitliche Dauer eines Bursts sowohl das Expiration-Geräusch, das Inspiration-Geräusch als auch die beiden Pausen zwischen diesen Geräuschen umfasst. Praktisch ergibt sich das Problem, dass vor allem bei stärkerem Hintergrundrauschen die Inspiration-Geräusche häufig weder hörbar noch auf dem Spektrogramm erkennbar sind. Daher wird die Zeitdauer eines Bursts oder Cry-Unit vom Beginn einer Expiration bis zum Beginn der darauf folgenden Expiration definiert und somit allein von den Expirations auf die Bursts geschlossen. Implizit wird somit eine Inspiration zwischen zwei Expirations angenommen.

Cry die insgesamte klangliche Antwort zu einem spezifischen Stimulus. Eine Gruppe mehre-

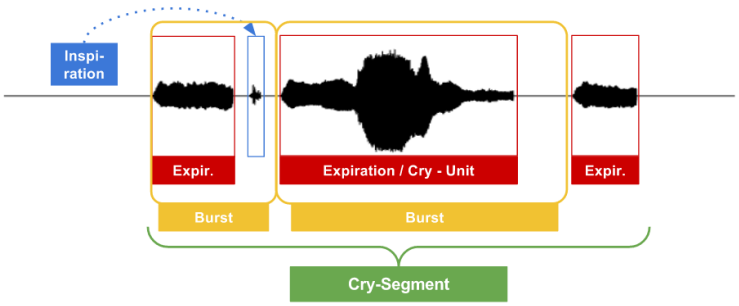

Abbildung 2.1: Veranschaulichung des Grundvokabulars

rer Cry-Units.[12] In dieser Arbeit wird ein *Cry* auch als **Cry-Segment** bezeichnet, um Verwechslungen zu vermeiden.

Weiterhin wurden von H Golub und M Corwin [12] Cry-Units in eine der folgenden drei Kategorien eingeführt:

Phonation beschreibt eine Cry-Unit mit einer „vollen Vibration der Stimmbänder“ mit einer Grundfrequenz zwischen 250 und 700 Hz. Entspricht umgangssprachlich einem Weinen mit einem „klaren, hörbaren Ton“.

Hyper-Phonation beschreibt eine Cry-Unit mit einer „falsetto-artigem Vibration der Stimmbänder“ mit einer Grundfrequenz zwischen 1000 und 2000 Hz. Entspricht umgangssprachlich einem Weinen mit einem „sehr hohen, aber klaren, hörbaren Ton“.

Dysphonation beschreibt eine Cry-Unit ohne klar feststellbare Tonhöhe, produziert durch Turbulenzen an den Stimmbändern. Entspricht umgangsprachlichen dem „Brüllen oder Krächzen“.

Eine Cry-Unit gehört dabei mindestens einer dieser Kategorien an, kann aber auch in seinem zeitlichen Verlauf die Kategorie wechseln. H Golub und M Corwin [12] stellen weiterhin eine Reihe an charakteristischen Eigenschaften vor, die in Bezug auf ein Cry-Segment berechnet werden.

Latency-Period beschreibt die Dauer zwischen dem zufügen eines Schmerz-Stimulus und dem beginn des ersten Cry-Bursts des Segmentes

Duration beschreibt die insgesamte Zeitdauer des Cry-Segmentes. Es wird keine genaue Definition gegeben, wodurch Beginn und Ende definiert werden. Das Segment endet dort, wo es „scheint, aufzuhören“.

Maximum-Pitch beschreibt die höchste festgetellte Grunfrequenz des Segmentes.

... und viele weitere, die in [12] nachgelesen werden können, aus Platzgründen an dieser Stelle jedoch nicht vollständig genannt werden.

3 System zur Visualisierung akustischer Schmerz-Scores

Das Ziel dieser Arbeit ist die Ableitung des Schmerz-Scores aus einem Audiosignal sowie die darauf folgende Visualisierung dieser Schmerz-Scores. Folgende Anforderungen werden an das System gestellt:

1. Das System muss dazu in der Lage sein, aus den akustischen Eigenschaften des Weinens die Schmerz-Score abzuleiten
2. Das System muss dazu in der Lage sein, die Schmer-Score zu visualiseren.
3. Die Verarbeitungspipeline muss genug Flexibilität bieten, um beliebige Pain-Scores einzubinden.
4. Die Analyse muss auch bei nicht-optimalen akustischen Bedingungen Einsatzfähig sein.
5. Die Methoden müssen kontinuierlich eingesetzt werden können. Das heißt, dass zu einem Analysezeitpunkt nur Informationen verwendet werden können, die nicht in der Zukunft liegen.

Im folgenden wird ein Überblick über bereits verföffentlichte Ansätze zur Analyse von akustischen Signalen Neugeborener oder sonstiger automatisierter Systeme zur Ableitung der Pain-Score gegeben.

3.1 Literatur-Übeblick

Bei der Aufgabenstellung handelt es sich grob betrachtet um einen Klassifikations/Regressions-Aufgabe, bei der aus den Eigenschaften des Audiossignals mit den Kindlichen Lautäußerungen eine Schluss gezogen werden soll. In dieser Aufgabenstellung ist der Schluss eine Pain-Score. An dieser Stelle wird ein Überblick über Veröffentlichungen gegeben, in denen ähnliche Aufgabenstellungen bearbeitet worden.

Der Großteil der Veröffentlichungen stellt Systeme Klassifikation einzelner Cry-Units vor, entweder bezüglich der Wein-Ursache (Hunger, Angst, Schmerz...) oder zur Diagnose bestimmer Krankheiten. Diese Methoden sind nicht für die kontinuierliche Analyse geeignet, sondern haben das Ziel, bezüglich einer bereits vorliegenden Cry-Unit eine möglichst hohe Klassifizierungs-Accuracy zu erzielen. Probleme wie Hintergrundrauschen, Berechnungsaufwand oder kontextuelle Informationen werden selten mit in Betracht gezogen. Beispiele für solche Systeme sind die von Abdulaziz et al [34] oder Furh et al [30].

Várallyay stellt in seiner Dissertation „Analysis of the Infant Cry with Objective Methods“ [32] Methoden zur automatisierten Analyse kindlicher Lautäußerungen vor. Das eigentliche Ziel der Dissertation ist die Erforschung der Unterschiede zwischen den Lautäußerungen gesunder und tauber Neugeborener. Die automatisierte Verarbeitungs-Pipeline der Audiosi-

gnale ist dabei ein „Nebenprodukt“ zur schnelleren Auswertung der Signale. Die Auswertung muss nicht kontinuierlich erfolgen. In der vorgestellten Verarbeitungspipeline wird das Eingangssignal in Zeitfenster weniger Millisekunden zerlegt und jedes Fenster auf Basis der Fenstereigenschaften als Stimmhaft oder nicht-Stimmhaft klassifiziert. Die stimmhaften Signalfenster werden zu *Segmenten* zusammengefasst (in Kapitel 2.3 als Cry-Unit bezeichnet). Auf Basis der Segmente werden Auswertungen bezüglich der Zeit-Bereiches (Durchschnittliche Segmentlänge, Pausenlängen etc.), des Frequenz-Bereiches (Grund-Frequenz, Formanten-Frequenzen etc.) und des Melodie-Verlaufes (Melodie-typ) angestellt. Analysiert wurden Signale mit einer Länge von 10 bis 100 Sekunden, die Lautäußerungen von Babies mit oder ohne Hörbehinderung beinhalten. Aus den Auswertungsergebnisse stellt Várallyay die wichtigsten Unterscheidungsmerkmale zwischen tauben und gesunden Babies fest. In der Dissertation [32] wird ein Überblick über das Vorgehen und die Ergebnisse gegeben. Die Verarbeitungsschritte werden detailllierter in einzelnen Veröffentlichungen beschrieben, auf die der Autor dieser Arbeit jedoch kein Zugriff gewährt wurde.

Cohen et al haben 2012 in dem Paper „Infant Cry Analysis and Detection “ [5] ein System zur Analyse der akustischen Signale von Neugeborenen vorgestellt. Dieses System klassifziert die Audio-Signale in eine der drei Klassen *Cry, No Cry* und *No Activity*. Mit *Cry* sind Lautäußerungen gemeint, die eine potentiell Gefahr für das Baby anzeigen, wie z.B. wie Schmerz oder Hunger. *No Cry* meint, dass das Baby zwar Laute von sich gibt, diese aber keine potentielle Gefahr anzeigen. emphNo Activity meint keinerlei Lautäußerung. Die Verarbeitungs-Pipeline wird detailliert vorgestellt und ist für die kontinuierliche Verarbeitung mit einer gewissen Verzögerungszeit spezialisiert. Das Signal wird in überlappende *Segmente* à 10 Sekunden zerlegt. Die Stimmaktivität in dem Segment wird algorithmisch festgestellt. Wenn Aktivität vorliegt, wird das Segment in Sections à 1 Sekunden zerlegt und die Stimmaktivität für jede Section analysiert. Wird genügend Stimmaktivität für eine Section festgestellt, wird die Section in *Frames*) à 32 Millisekunden zerlegt und Features für jedes Signalfenster errechnet. Mit Hilfe eines Predictors werden die Frames in *Cry, No-Cry, No-Activity* klassifiziert, wobei Kontextuelle Informationen der umliegenden Frames mit einbezogen werden. Aus den Klassen der Frames wird auf die Klasse der Section geschlossen, und aus den Klassen der Sections auf die Klasse des 10 Sekunden langen Segments. Das System hat in Bezug auf diese Arbeit den Vorteil, dass ebenfalls die kontinuierliche Verarbeitung im Vordergrund steht. Der Nachteil an dieser Methode ist, dass die zeitliche Einheit, für die die Klassifizierung vorgenommen wird, auf unflexibel auf 10 Sekunden festgelegt ist. Daher müsste diese Verarbeitungspipeline abgewandelt werden, um Anstelle der Ableitung der drei genannten Klassen einer Pain-Score zu verwenden, die einen längeren Beobachtungszeitraum als 10 Sekunden benötigt.

Pal et al haben 2006 in dem Paper „Emotion detection from infant facial experessions and cries“ [25] ein System zur Emotions-Detektion bei Neugeborenen aus Aufnahmen des Gesichtsausdruck und akustischen Aufnahmen des Weinens vorgestellt. Die zu erkennenden Emotionen sind *Traurigkeit, Wut, Hunger, Angst und Schmerz*. Es wird nicht erwähnt, ob die Analyse kontinuierlich oder nicht-kontinuierlich erfolgt. Bei der Verarbeitung der akustischen Signale werden die Features *Grund-Tonhöhe* und die *Frequenz der ersten drei Formanten* extrahiert und mit einem Klassifikations-Algorithmus klassifiziert. Es werden keinerlei Details genannt, inwiefern die Features aus kurzen Signalfenstern oder längeren Signalabschnitten errechnet werden, welche Vorverarbeitungsschritte angewandt werden und ob die Klassfizierung auf Ebene der Signalfenster oder über längere Zeitabschnitte

hingweg geschieht. Die Veröffentlichung liefert Ideen über mögliche Features, bietet jedoch keinen Einblick in die Verarbeitungspipeline.

Zamzi et al haben 2016 in dem Paper „An Approach for Automated Multimodal Analysis of Infants' Pain" [9] ein System zur automatisierten und kontinuierlichen mutlimodalen Analyse von Neugeborenen zur Ableitung des Schmerzes vorgestellt. Das System trägt den Namen *MPAS*. Der Insgesamte Schmerzgrad wird aus den Analyseergebnissen der monomdaler Schmerzindikatoren für *Gesichtsausdruck, Körperbewegung, Vitalfunktionen und Weinen* errechnet. Das Ziel des Projektes kommt der Aufgabenstellung dieser Masterarbeit am nächsten, da es ebenfalls um die Ableitung von Schmerz in einem multimodalen Verbund geht. Es wird jedoch nicht die Anfoderung gestellt, Flexibilität in der Wahl der Pain-Score zu gewährleisten . Während in der Veröffentlichung die Analyse der ersten drei genannten Schmerzindikatoren angekündigt wird, werden daraufhin die Methoden zur Analyse der akustischen Signale *nicht* erläutert. Auch die ersten Validierungs-Ergebnisse beziehen sich nur auf den Gesichtsausdruck, Körperbewegung und Vitalfunktionen. Es ist nicht klar, ob die Miteinbeziehung akutischer Signale fallen gelassen wurde. Die Ausführungen konzentrieren sich dazu vermehrt auf die Methoden zur Kombination der Auswertungsergebnisse der monomodalen Schmerzindikatoren. Die Verarbeitungs-Pipelines der monomdalen Schmerzindikatorn werden nur grob vorgestellt.

3.2 Verarbeitungs-Pipeline

In Kapitel 3.1 wurden verschiedene Systeme vorgestellt, deren Problemstellungen dem Thema dieser Masterarbeit ähneln. Keine der präsentierten Verarbeitungs-Pipelines eignet sich, um mit nur leichten Anpassungen übernommen werden zu können: Entweder sind die Verarbeitungsschritte nicht für die kontinuierliche Verarbeitung konzipiert [34] [30] [32], nicht genügen abstrahiert, um für andere Klassifizierungen als die ursprünglich geplanten abgewandelt werden zu können [5], oder stellen die Verarbeitungs-Pipeline nicht vor [25] [9].

In dieser Arbeit wird die folgende Verarbeitungs-Pipeline vorgestellt. Sie wird in in Abbildung 3.1 visualisiert.

1. **Pre-Processing**. Vorverarbeitung des Signals. An dieser Stelle geschieht eine Anpassung der Lautstärke mit Hilfe eines Audiocompressors zur besseren Kontrolle der Signalenergie. Das Pre-Processing wird in Kapitel 3.3 vorgestellt.
2. **Voice-Activity-Detection**. Das Audiosignal wird in einander überlappende Zeitfenster weniger Millisekunden zerschnitten. Mit Hilfe eines Klassifizierungs-Algorithmus werden die Zeitfenster in als *Stimmhaft* oder *nicht Stimmhaft* markiert. Ununterbrochene Reihen von Stimmhaften Signalfenstern werden zu *Cry-Units* zusammengefasst. Das Ergebnis der Voice-Activity-Detection sind Markierungen der Anfangs- und Endzeitpunkte *Cry-Units*, die die Basis aller darauf folgenden Auswertungen bilden. Diese Idee ist aus der Dissertation von Várallyay [32, S. 16 - 17] übernommen, welcher Cry-Units als *Segments* bezeichnet. Die Voice-Activity-Detection wird in Kapitel 3.4 vorgestellt.
3. **Segmentierung** (engl *Segmenting*), das Zusammenfassen mehrer Cry-Units zu Segmenten, welche in Kapitel 2.3 als *Cry* bezeichnet werden. Dieser Schritt ist notwendig, weil die Ableitung der Schmerz-Scores nicht aus den Informationen einer Cry-Unit, sondern aus dem Verbund mehrerer Cry-Units geschieht. Keine der in Kapitel 3.1 vorgestellten Veröffentlichungen beschreibt ein Verfahren, welches adaptiert werden können, entweder

weil der Input der Algorithmen bereits auf die Länge der Segmente beschnitten wurde, oder weil ein eventuell verwendetes Verfahren nicht beschrieben wird. Daher wird ein simpler Algorithmus für die Segmentierung vorgeschlagen, welcher für die kontinuierliche Auswertung implementiert werden kann. Die Segmentierung wird in Kapitel 3.5 vorgestellt.

4. **Feature-Extraction**, das heißt die Berechnung von Eigenschaften für jedes Segment, die für die Ableitung der Pain-Scores von Interesse sind. Diese Eigenschaften weden in Regelmäßigen Zeitintervallen innheralb des Segmentes abgefragt. Diese Feature-Extraktion ist ein notwendiger Vorbereitungsschritt für die Anschließende Klassifikation/Regression, welcher in allen in Kapitel 3.1 vorgestellten Veröffentlichungen durchgeführt wird. Die Feauture-Extraction wird in Kapitel 3.6.1 vorgestellt.
5. **Ableitung der Pain Score** aus den Features des Segmentes. Während es sich in allen in Kapitel 3.1 vorgestellten Veröffentlichungen um Klassifikationsaufgaben handelte, wird hier eine Regression vorgenommen. Die Feature-Extraction wird in Kapitel 3.6 und 3.6.2 vorgestellt.
6. **Visualisierung** der errechneten Pain-Score. In dieser Arbeit werden mehrere Versionen eines Systems vorgeschlagen, welche den zeitlichen Verlauf auf Ampel-Farben abbilden, welche die höhe der Schmerz-Score codieren. die Visualisierung wird in Kapitel 3.7

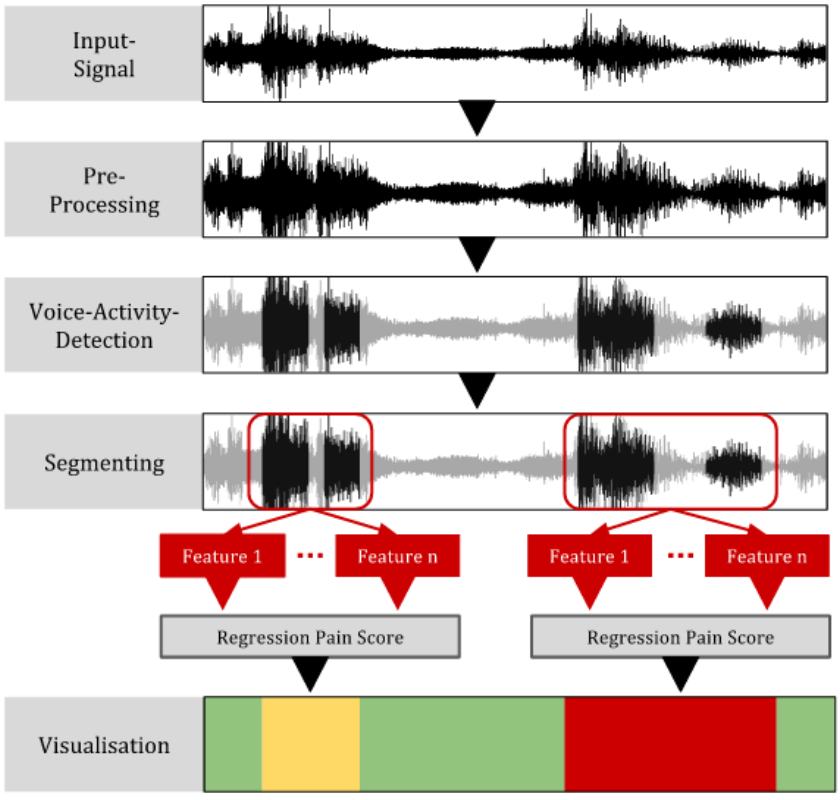

Abbildung 3.1: Die Verarbeitungs-Pipeline des vorgestellten Systems

3.3 Preprocessing

Beim Preprocessing wird das Signal so vorverarbeitet, dass Störeinflüsse auf die darauf folgenden Verarbeitungsschritte von vorneherein minimiert werden. Welches Pre-Processing durchgefüht wird, ist Abhägig von der konrketen Aufgabenstellung. So werden beispielsweise bei einigen Algorithmen zur Voice-Activity-Detection, also dem markieren stimmhafter Signalabschnitte, Tiefpass, Hochpass- und Bandpassfilter eingesetzt, um diejenigen Frequenzanteile herauszufiltern, die von der Stimme nicht produziert werden können [17] [29] [16]. Bei einigen Pitch-Detection-Algorithmen wird *Centerclipping* eingesetzt, also das 0-Setzen von Samples mit $x[i] < 0.5 \cdot$ Maximalaussteuerung.[7]

In dieser Arbeit wurde sich für eine Vorverarbeitung entschieden, bei der das Signal hinsichtlich seiner Dynamik im Zeitbereich eingegrenzt wird. Dies ist ein typischer Vorverarbeitungschritt bei Sprachaufnahmen. Hintergrund ist, dass sehr kurz, aber sehr laute Pegelspitzen weit über dem Durchschnittspegel des Gesamtsignals den Maximalwert des Signals unnötig begrenzen und die Signalenergie so gering halten. Da die Testsignale, die in dieser Arbeit verwendet werden, aus inhomgenen Quellen stammen und sehr unterschiedliche Lautstärken haben, wird so gewährleistet, dass sie zumindest ähnliche Energien haben. An dieser Stelle werden (noch) keine Frequenanteile herausgefiltert, um keine Frequenzen zu verlieren, die in den späteren Verareitungsschritten wieder Voice-Activity-Detection 3.4 oder der Feature-Extraction eventuell noch benötigt werden.

Die Dynamikeinschränkung wird mit Hilfe eines Audiokompressor umgesetzt. Ein Audiokompressor verringert Signalspitzen, die über einen festgelegten *Schwellwert (Threshold)* liegen, um ein festgelegtes *Verhältnis (Ratio)*. Ein Threshold von 0.3 mit Ratio von 0.5 bedeutet beispielsweise, dass alle Signalspitzen, die den Wert 0.3 überschreiten oder -0.3 unterschreiten, um 50% verringert werden. Ein Kompressor kann auf die Überschreitung des Thresholds erst nach einer als *Attack* bezeichneten Verzögerung reagieren, und bei erneuten Verlassen des Thresholdes mit einer als *Release* bezeichneten Verzögerung nachwirken. Signalspitzen werden so verringert und die Lautstärke-Dynammik eingeschränkt. Die tatsächliche Erhöhung der Signalenergie geschieht im Anschluss durch die Anhebung der insgesamten Signallautstärke, wie Beispielsweise der Normalisierung des Signals auf den Maximalpegel. Abbildung 3.2 zeigt die Parameter eines solchen Audio-Kompressors.

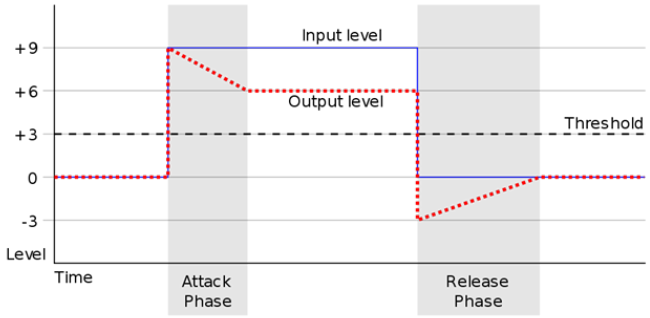

Abbildung 3.2: Parameter eines Audio-Kompressors

Der entwickelte Kompressor automatisiert die Einstellung von Threshold und Ratio auf Grundlage des Root-Mean-Square (RMS) des Signales x der Länge N. Der RMS-Wert ist ein Maß für die durchschnittliche Signalenergie und wird wie nach Formel 3.1 berechnet.

Threshold und Ratio werden nach den Formeln 3.2 und 3.3 berechnet, wobei der Parameter r_a den Ziel-RMS-Wert anbgibt und mit dem Wert.

$$\mathrm{RMS}(x) = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} x[n]^2} \tag{3.1}$$

$$\mathrm{THold}(x) = \left[\frac{\mathrm{RMS}(x[])}{r_a}\right]^2 \tag{3.2}$$

$$\mathrm{Ratio}(x) = \left[\frac{\mathrm{RMS}(x[])}{r_a}\right]^2 \tag{3.3}$$

Abbildung 3.3 zeigt das ein Signal vor und nach dem Preprocessings. Zu sehen ist, dass die Lautstärke der einzelnen Schrei-Einheiten nach der Anpassung einheitlicher ist.

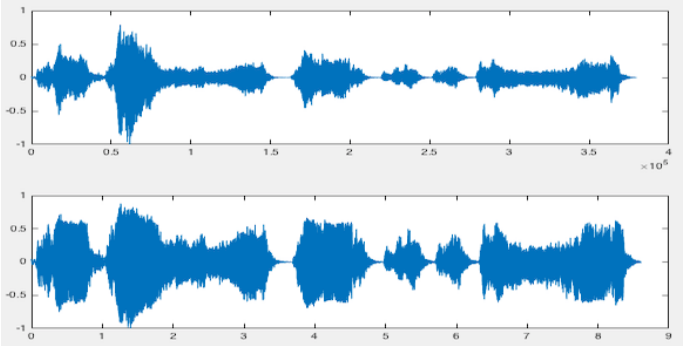

Abbildung 3.3: Ergebnis des Preprocessings

3.4 Voice Activity Detection

Das Ziel ist, in einem Audiosignal diejenigen Stellen zu markieren, in denen Stimme enthalten ist. Abbilung 3.4 visualisiert ein Beispiel für eine solche Markierung: Zu sehen ist der Zeitbereich eines Audiosignales mit drei klar erkennbaren Cry-Units. Die rote Linie, die das Signal überspannt, bildet die Zeiteinheiten des Eingangssignales in die binären Kategorien *Stimmhaft* und *Stille* ab.

Die Erkennung des Vorhandenseins von Stimme in einem Signal wird als *Voice Activity Detection (VAD)* oder auch *Speech Detection* bezeichnet. Das Ziel ist die Unterscheidung von denjenigen Zeiträumen im Signal, in denen Stimme enthalten ist, von den Zeiträumen

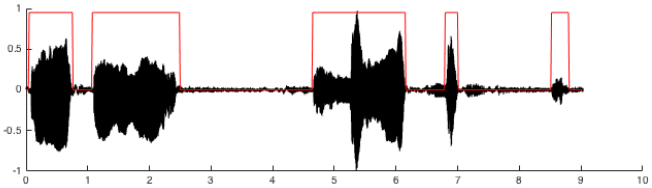

Abbildung 3.4: Markierung von Schreigeräuschen im Audiosignal. Schwarz: Das Eingangssignal $x[\,]$. Rot: Klassifizierung in Stimmhaft/Stille

ohne Stimme. Die größte Herausforderung für VAD-Algorithmen ist die robuste Erkennung bei Signalen mit Rauschen unbekannter Stärke und Natur. [16, S. 1] [31, S. 1]

Der Grundlegende Aufbau eines VAD-Algorithmus ist wie folgt:

1. **Windowing**: Unterteilung des Signals in (einander überlappende) Fenster, für die Entscheidung durchgeführt werden soll.
2. **Feature-Extraction** aus den einzelnen Fenstern
3. **Thresholding / Klassifizierung** über die Präsens oder Nicht-Präsens von Stimme für jedes Zeitfenster auf Grundlage der Extrahierten Features mit Hilfe von Entscheidungsregeln wie Grenzwerten.
4. **Decision-Smoothing**, das nachträgliche Hinzufügen oder Entfernen von Entscheidungen mit Hilfe von kontextuellen Informationen der umliegenden Entschiedungen.[14, S. 8 - 9] [16, S. 1 - 2]

Der an dieser Stelle entwickelte Ansatz ist eine Kombination aus den Ideen, die von Moattar et al [18], Kristjansson et al [31], Waheed et al [17], Ahmadi et al [29] und Shen et al[15] vorgestellt wurden.

3.4.1 Windowing

Das Signal $x[\,]$ wird nach den in Kapitel **??** beschriebenen Verfahren nach Gleichung **??** in die Signalfenster $x_0[\,] \ldots x_m[\,]$ zerlegt, bezeichnet als „Windowing“. Die Signalfenster werden zunächst im Zeitbereich belassen. Es wurde sich für die Waheed et al [17] vorgestellte Fensterlänge von 25 ms entschieden, als Kompromiss zwischen den von Moattar et al[18] empfohlenen 10 ms und den von Ahmadi et al [29] empfohlenen 40 ms. Die Fenster überlappen einander um 50%, das heisst 12.5 ms.

3.4.2 Feature Extraction

Für jedes Signalfenster $x_0[\,]...x_m[\,]$ à 25 ms werden die folgenden Features aus den Kategorien **Zeit-Bereich**, **Frequenz-Bereich**, **Cesptrum** und **Auto-Korrelation** berechnet.

Zeit-Bereich

Im Zeit-Bereich werden die beiden Features *Root-Mean-Square*-Wert *[RMS]* und *Zero-Crossing-Rate [ZCR]* berechnet.

Moattar et al [18] bezeichnen den Energiegehalt eines Signals als das für die VAD am häufigsten Angewandte Feature. Daher wird der RMS-Wert eines Signalfensters nach Gleichung 3.1 verwendet. Hintergrund ist, dass der Energiegehalt eines Stimmsignals typischerweise Höher ist als der des Hintergrundrauschens. Bei geringen Signal-to-Noise-Ratios ist diese Bedingung jedoch nicht immer gegeben. Als zweites Feature des Zeitbereiches wird die in verwendete *Zero-Crossing-Rate* berechnet. Die ZCR nach Formel 3.4 gibt an, wie häufig ein Vorzeichenwechsel im Signal vorkommt. Eine höhere ZCR weist auf Stille hin, da Rauschen typischerweise einen höheren ZCR als Signale mit einer Periodizität aufweist.

Problematisch ist dieses Kriterium bei Signalen, bei denen gar kein Hintergrundrauschen vorliegt, da solche Signalfenster eine ZCR von 0 aufweisen. [29]

$$\mathrm{ZCR}(x_i[\,]) = \sum_{1}^{N-1} |\mathrm{sng}(x_i[n]) - \mathrm{sng}(x_i[n-1])| \tag{3.4}$$

Autokorrelation

Neben den in Kapitel 3.4.2 geannten „einfachen“ Features des Zeitbereiches wird zur VAD die Autokorrelation verwendet. Wie in Kapitel **??** ausgeführt, weisen stimmhafte Signale eine höhere Periodizität als das Hintergrundrauschen auf. Daher eignet sich die in Kapitel **??** vorgestellte Autokorrelation, um diese Periodizität festzustellen. Es werden die Features *Maximum Autocorrelation Peak* [$aMax$] und (*Autocorrelation Peak Count*) [$aCount$] berechnet.

Beide Features werden von Kristjansson et al [31, S. 1 - 2] zur VAD erprobt. Die *höchste Magnitude der Autokorrelation* (*Maximum Autocorrelation Peak*) wird nach der Formel 3.5 definiert und bestimmt die höchste Magnitude im Autokorrelations-Signal. Eine höherer [$aMax$]-Wert spricht für eine dominante Grundfrequenz im Signal. Das zweite Feature ist die *Anzahl an Autokorrelations-Spitzen* nach Formel 3.6. Ein höherer [$aCount$]-Wert spricht für das vorhandensein dominanter Obertonwellen im Signal. Aus Kapitel 2.3 geht hervor, dass die Grundfrequenz von Neugeborenen zwischen $200 - 2000\,\mathrm{Hz}$ liegt, weshalb auch nur in Lags dieses Bereichs die Autokorrelation durchgeführt wurde.

$$\mathrm{aMax}(x_i[\,]) = \max_k \mathrm{mag}\{\mathrm{NA\text{-}Corr}_k(x_i[\,])\} \tag{3.5}$$

$$\mathrm{aCount}(x_i[\,]) = \operatorname*{count}_k \mathrm{mag}\{\mathrm{NA\text{-}Corr}_k(x_i[\,])\} \tag{3.6}$$

Frequenz-Bereich

Aus dem **Frequenz-Bereich** werden die drei Features *unnormalisierte spektrale Entropie* [$SEnt_u$], *normalisierte spektrale Entropie* [$SEnt_n$] und *dominanteste Frequenzkomponenten* [f_{Dom}] berechnet.

Als Vorbereitungsschritt werden die Signalfenster des Zeit-Bereiches $x_0[\,] \dots x_m[\,]$ zunächst mit der in Kapitel **??** vorgestellten Short Time Fourier Transformation in die *Frequenz-Fenster* $X_0[\,] \dots X_m[\,]$ transformiert. Das heißt, dass $X_i[\,] = \mathrm{DFT}(w[\,] \cdot x_i[\,])$. Es wurde eine 2048 Punkte Lange FFT und eine Hamming-Window als Fensterfunktion verwendet.

Kristjansson et al [31, S. 2] verwenden die *spektrale Entropie* Voice Activity Detection. Dabei wird das Spektrum des Frequenzfensters X_i als Wahrscheinlichkeitsverteilung betrachtet. Die Entropie als Maß zur „Unreinheit“ wird in Kapitel **??** erläutert. Die *normalisierte spektrale Entropie* wird nach der Formel 3.8 berechnet. Das Signal $px_i[\,]$ ergibt sich durch die Normalisierung des N-Punkte langen Spektrums nach Formel 3.7. Neben der in [31] vorgestellten normalisierten spektralen Entropie wird zusätzlich die *unnormalisierte Spektrale Entropie* nach Formel 3.9 berechnet. Bei dieser wird das Spektrum nicht

normalisiert, das heißt, es gilt $px_i[f] = X_i[f]$. Somit hat Energie des Signals einen größeren Einfluss die höhe des Features. Bei der normalisierten spektralen Entropie ist zu erwarten, dass Frequenzfenster mit Hintergrundrauschen eine höhere Entropie haben als Fenster mit Stimme . Bei der unnormalisierten spektralen Entropie ist zu erwarten, dass Signalfenster mit Stimme eine höherer Spektrale Entropie haben als Fenster mit Stille.[1]

In die Berechnungen wurden nur die Frequenzen im Bereich von 200 - 8000 Hz mit einbezogen, da aus Kapitel **??** die tiefst Mögliche Frequenz kindlicher Lautäußerung bei 200 Hz liegt und nach Shen et al [15] Stimme keine Informationen oberhalb von 8000 Hz übertragen.

$$px_i[n] = \frac{X_i[n]}{\sum_{k=1}^{N} X_i[k]} \tag{3.7}$$

$$\mathrm{SEnt}_n(px_i[\,]) = -\sum_{k=1}^{N} px_i[k] \cdot \log(px_i[k]) \tag{3.8}$$

$$\mathrm{SEnt}_u(X_i[\,]) = -\sum_{k=1}^{N} X_i[k] \cdot \log(X_i[k]) \tag{3.9}$$

Moattar et al [18, S. 2550] stellen die *dominanteste Frequenzkomponente* zur Voice-Activity-Detection vor. Für jedes Frequenzfenster $X_i[\,]$ wird diejenige Frequenz nach Formel 3.10 berechnet, welches die höchste Amplitude hat. Es wird dabei, im Gegensatz zur spektralen Entropie, der gesamte Frequenzraum betrachtet. Ein stimmhaftes Signal hat typischerweise eine höhere f_{Dom} als ein nicht stimmloses Signal, bedingt durch die hohe Amplitude der Grundfrequenz.

$$f_{Dom}(X_i[\,]) = \arg\max_k \{X_i[k]\} \tag{3.10}$$

Cepstrum

In Kapitel **??** wurde das Cepstrum vorgestellt und Erläutert, wie Peaks im oberen Quefrency-Bereich auf das Vorhandensein eines periodischen, obertonreichen Signals, wie die Stimme eines ist, hinweist. Aus dem Cepstrum-Bereich werden die Features *Upper Cepstrum Peak* $[Ceps_{mag}]$ und *Upper Cepstrum Peak Location* $[Ceps_{loc}]$ berechnet.

Ahmadi et al [29] sowie Kristjansson et al[31] schlagen vor, die *höchste Magnitude im oberen Quefrency-Bereich* (Upper Cepstrum Peak) als Feature zu verwenden. Formel 3.11 definiert die Berechnung. $c_i[\,]$ ist das Cepstrum des i-ten Frequenzfenster $X_i[\,]$. Wie in Kapitel 2.3 erläutert, liegt die Grundfrequenz bei kindlichen Lautäußerungen zwischen 200 und 2000 Hz, was einem Quefrency-Bereich von 5 - 40 ms entspricht. Folglich werden bei der Berechnung nach Formel 3.11 nur Quefrency-Werte in diesem Bereich betrachtet. Eine hoher $Ceps_{mag}$-Wert weist auf das Vorhandensein von Stimme für das aus dem Fenster $x_i[\,]$ Berechneten Cepstrum $c_i[\,]$ hin. Als zweites Features wird die Quefrency der höchsten

[1] Kristjansson et al [31, S. 2] verwenden zur Entropie-Berechnung den Logarithmus zur Basis 10, anstatt zur Basis 2. Es ist nicht klar, ob es sich dabei um einen Fehler handelt. Zur Featureberechnung in dieser Arbeit wurde, wie in dem Paper beschrieben, ebenfalls der Logarithmus zur Basis 10 verwendet!

Amplitude des Cepstrum (Upper Cepstrum Peak Location) nach Formel 3.12 berechnet. Bei Signalfenstern mit Stille ist es wahrscheinlicher, dass sich die höchste Ampltiude am Mindest- oder Maximum-Wert des durchsuchten Quefrency-Bereiches befindet.

$$Ceps_{mag}(c_i) = \max_k \text{mag}\{c[k]\} \tag{3.11}$$

$$Ceps_{loc}(c_i) = \arg\max_k \{c[k]\} \tag{3.12}$$

Abbildung 3.5 visualisiert alle vorgestellten Features, die für die Voice Activity Detection eingesetzt werden. Der oberste Plot zeigt das Audiosignal aus Abbildung 3.4 mit einem Signal/Rauschabstand von 20 dB. Der rote Graph über den Plot klassifiziert die Zeitbereiche in 1 = *stimmhaft* und 0 = *nicht stimmhaft*. Alle darunter eingezeichneten Plots zeigen den zeitlichen Verlauf der entsprechenden Features. Bei jedem Feature ist eine Korrelation mit der stimmhaftigkeit oder nicht-stimmhaftigkeit der entsprechenen Signalabschnitte zu sehen, welche bei einigen Feature stärker ausfällt als bei anderen..

Konstruktion des Feature-Raumes

Abbildung 3.6 zeigt in (A) des zeitlichen Verlauf des *RMS*-Features eines Signals mit einem Signal-Rausch-Abstand (SNR) von 50 dB. Die Zeiträume mit Stille haben einen weitaus niedrigeren RMS-Wert als die Zeiträume mit Stimme. In (B) ist das selbe Signal mit einem Signal-Rausch-Abstand von 3 dB zu sehen. Nun liegen die RMS-Werte der stimmlosen Bereiche nur noch knapp unter denen des Sprachsignals. Zu sehen ist, dass starkes Hintergrundrauschen ähnlich hohe Feature-Werte erzeugen kann wie die Stimme.

Moattar et al [18] und Waheed et al [17] präsentieren die Idee, den Wert des jeweiligen Features zu messen, der in den stimmlosen Bereichen durch das Hintergrundrauschen erzeugt wird. So kann davon ausgegangen werden, dass die ersten Signalfenster zunächst noch keine Stimme enthalten, und der Feature-Wert des Rauschens somit anhand der ersten Signalfenster bestimmt werden. Bei einer langanhaltenden und kontinuierlichen Analyse können sich die Signal/Rausch-Verhältnisse jedoch ständig ändern, weshalb der Feature-Werte der stimmlosen Signalbereiche regelmäßig aktualisiert werden müssem. Es kann jedoch davon ausgegangen werden, dass die Länge einer Cry-Unit eine bestimmte Länge t_{max} nicht überschreiten kann, bevor das Babie Luft holen muss und somit ein Zeitfenster mit Stille entsteht. So haben Zeskind et al [24] diesen Wert mit $t_{max} = 4.75\,\text{s}$ festgstellt. In einem Zeitbereich $t > t_{max}$ muss somit zumindest ein Feature-Wert enthalten sein, der durch stimmlose Signalteile erzeugt wird. Auf Basis dieser Überlegung wird das *Differenz-Feature* $\text{Diff}(Feat(x_i[\,]))$ nach Formel 3.13 definiert als die Differenz des aktuell gemessenen Feature-Wertes und des geringsten Feature-Wertes, welcher im vergangenen Zeitbereich t gemessen wurde. $Feat(x_i[\,])$ ist dabei ein beliebiger Feature-Wert des Signalfensters $x_i[\,]$, t_{xi} die Länge eines Signalfensters in Sekunden x_i ist (in diesem Fall 25 ms), und t der in der Vergangenheit zu durchsuchende Zeitbereich in Sekunden, welcher größer als t_{max} gewählt wird. In Abbildung 3.6 wird in (C) das Differenz-Feature für den RMS-Wert gezeigt.

$$\text{Diff}_t(Feat(x_i[\,])) = Feat(x_i[\,]) \; - \min_{k=i-z\ldots i}(Feat(x_k[\,])), \qquad z = \frac{2t}{t_{xi}} \tag{3.13}$$

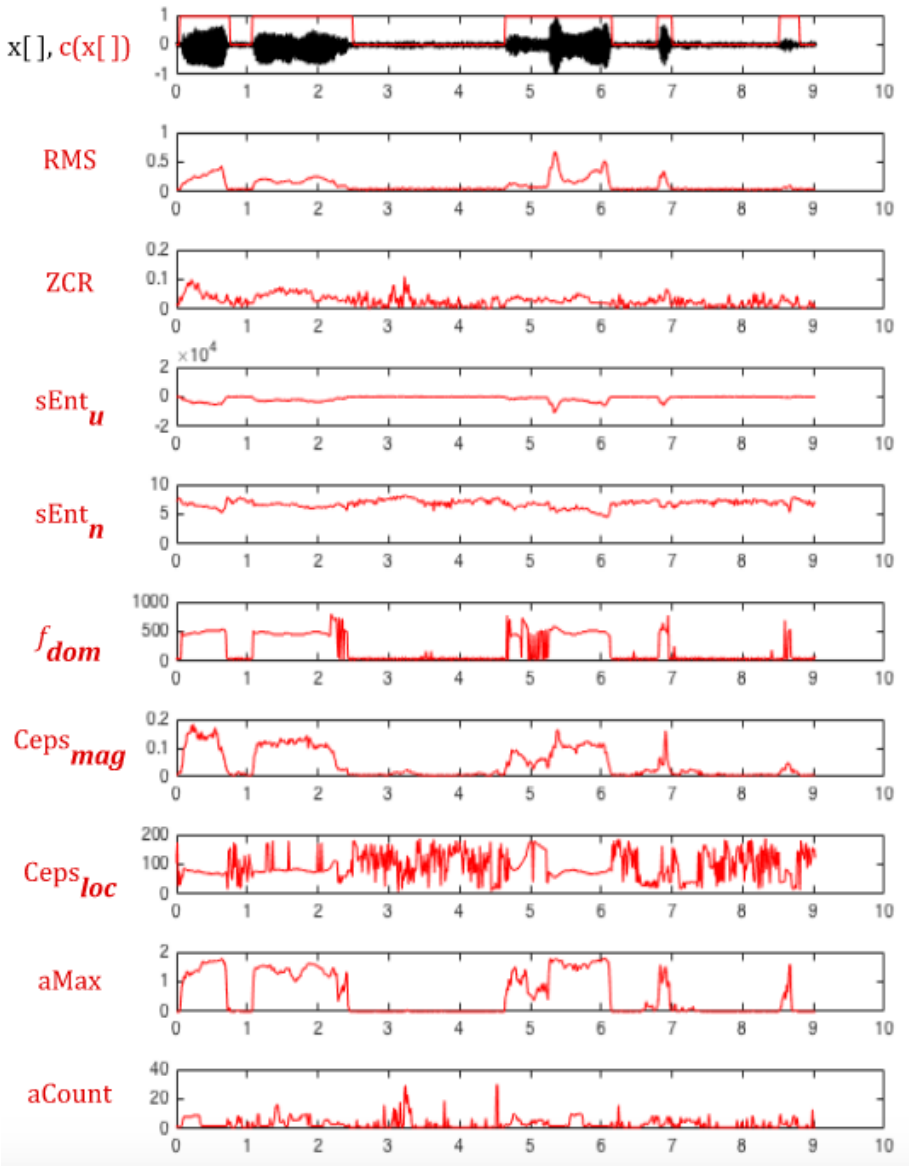

Abbildung 3.5: Übersicht über alle Features, die für die Voice Activity Detection verwendet werden.

Der Feature-Raum setzt sich schlussendlich folgendermaßen zusammen: Die ersten 9 Features bilden die in Kapitel 3.4.2 - 3.4.2 insgesamt 9 vorgestellten Features *RMS, ZCR,* $SEnt_u$*,* $SEnt_n$*,* f_{Dom}*,* $Ceps_{mag}$*,* $Ceps_{loc}$*, aMax* und *aCount*. Weiterhin wird für jedes Feature nach Formel 3.13 das Differenz-Feature mit $t = 5\,\mathrm{s}$ berechnet. Die Features *ZCR,* $SEnt_u$ und *aCount* wurden vor der Berechnung des Differenz-Features bezüglich ihres Vorzeichens invertiert, da bei Ihnen ein hoher anstatt ein niedriger Wert stimmhafte Signalteile anzeigt. Das einzige Feature, welches nicht als Differenzfeature dem Feautrevektor beigefügt wurde, ist der *Upper Cepstral Peak Location*-Feature [$Ceps_{loc}$], da es bei Stille sowohl einen höheren als auch einen niedrigeren Wert annehmen kann. Der Feature-Raum umfasst somit

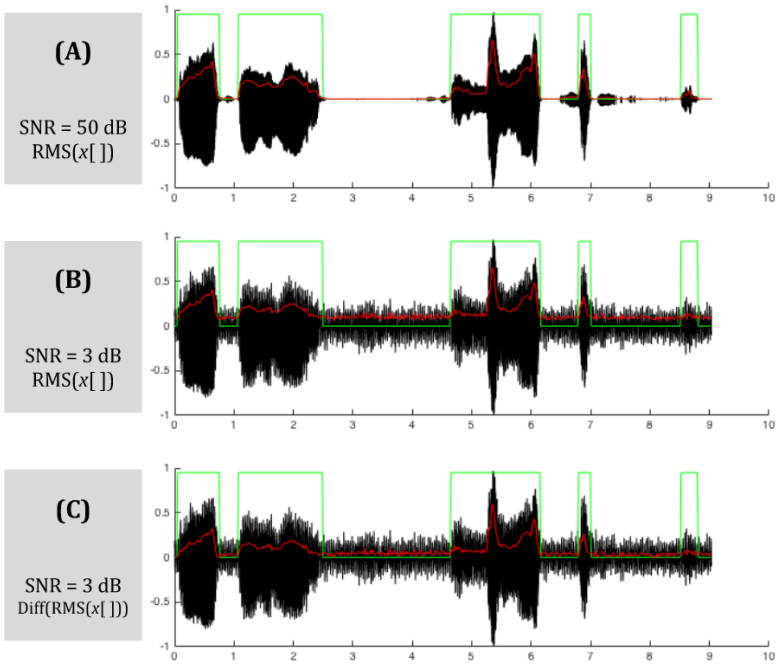

Abbildung 3.6: Das RMS-Feature bei verschiedenen Signal/Rausch-Abständen. Schwarz: Eingangs-Signal $x[\,]$. Grün: Klassifizierung in Stimmhaft/Stille. Rot: Feature-Wert.

insgesamt $9 + 8 = 17$ Dimensionen. Gleichung 3.14 verdeutlicht die Zusammensetzung des Feature-Vektors v_i, der aus dem Signalfenster $x_{[\,]}$ berechnet wird.

$$v_i = \Big(\mathrm{RMS}(x_i[\,]), ..., \ \mathrm{aCount}(x_i[\,]), \mathrm{Diff}_t(\mathrm{RMS}(x_i[\,]))....\mathrm{Diff}_t(-\ \mathrm{aCount}(x_i[\,]))\Big) \quad (3.14)$$

3.4.3 Thresholding

Finden der Grenzwerte

Wird die Voice-Activity-Detection für ein Signal $x[\,]$ durchgeführt, wird es in die Signalfenster $x_1[\,]...x_n[\,]$ zerlegt die Featurevektoren $v_1...v_n$ berechnet. Das Ziel ist nun, Grenzwerte für die Features zu finden, bei deren Über- oder Unterschreitung das Signalfenster als *stimmhaft* kategorisiert wird. Abbildung 3.7 verdeutlicht das Prinzip für das Feature *RMS*. Diese Entscheidung nach einem Grenzwert ist ein klassisches Vorgehen bei der Voice-Activity-Detection. Eine binäre Klassifizierung nach dem Muster $C(x_i) = \{1, \text{wenn } \mathrm{RMS}(x_i[\,]) \geq 0.18, \quad 0 \text{ sonst}\}$ würde auf den in diesem Fall für eine weitgehend richtige Klassifizierung vornehmen.

Eine Methode zum Finden der optimalen Grenzwerte ist der in Kapitel **??** vorgestellte *C4.5*-Algorithmus. Da der *C4.5* Entscheidungsbäume erstellt, kann die Entschiedung aufgrund der Verkettung von Grenzwerten mehrerer Features gefällt werden. Ein Beispiel wird in Listing 3.1 dargestellt, bei dem die Klasse eines Signalfensters hierarchisch zuerst nach einem Grenzwert für Ceps_{mag} und danach für den RMS-Wert entschieden wird.

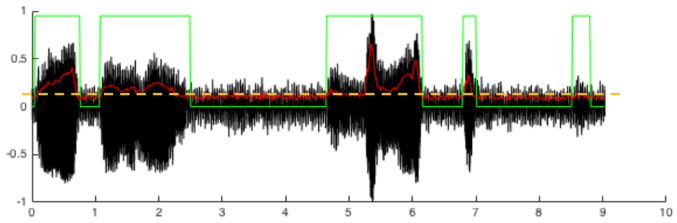

Abbildung 3.7: Thresholding eines Feature-Signales. Schwarz: Das Signal $x[\,]$. Grün: Klassifizierung in Stimmhaft/Stille. Rot: RMS-Feature. Orange: Beispiel-Grenzwert

Listing 3.1: Beispiel eines CART-Entscheidungsbaums

```
if Ceps_mag(x_i[]) > 0.2
|    if RMS(x_i[]) < 0.13
|    |    C(x_i[]) = 0
|    |else
|    |    C(x_i[]) = 1
|else
|     C(x_i[]) = 1
```

Trainings- und Testdatensätze

Zur Training und zur Evaluation des REPTree muss ein Datensatz D erstellt werden, dessen Erzeugung in diesem Kapitel beschriebn wird.

Es wurden sechs Audioaufnahmen mit Weinen von Babies von der freien Online-Sound-Bibliothek `https://www.freesound.org/` heruntergeladen und zu Segmenten à 10 Sekunden beschnitten. Es handelt sich um weitgesgehend rauschfreie Aufnahmen, die von verschiedenen Babys stammen. In den Audiosignalen wurden manuell die Zeitbereiche markiert, welche Stimme enthalten. Es wurden *keine* Geräusche markiert, bei denen es sich offensichtlich um Einatumungs-Geräusche handelt. Geräusche, bei denen nur Anhand der Aufnahme nicht mit Sicherheit festgestellt werden konnte, es ob sie durch Einatmungs- oder Ausatmungs-Geräusche handelt, wurden als Stimme markiert. Weiterhin wurden drei verschiedene Rauschsignale heruntergeladen. Es handelt sich um „realistische“ Atmosphären von Krankenhäusern. Jedes der sechs Audioaufnahmen der Babys wurde mit jedem der drei Rauschsignale überlagert, einmal mit einem Signal/Rausch-Abstand von 50 dB („fast unhörbares Rauschen“), und einmal mit einem Signal/Rausch-Abstand von 3 dB („starkes Rauschen“). Außerdem wurde ein siebte Aufnahme eines Babies heruntergeladen, welches mit einem vierten Rauschsignale mit einem SNR von 7 dB überlagert wurde. Dieses Signal spielt eine Sonderrolle, da es bei der Konstruktion der Entscheidungsbäume nur zur Verifikation verwendet wird. So wurden vier Mengen an Audiosignalen erzeugt:

$\mathbf{A}_{50\,\mathrm{dB}}$ enthält $3 \cdot 6 = 18$ Audiosignale, bei dem alle sechs Baby-Aufnahmen mit den drei Rauschsignalen bei einem Signal-Rausch-Abstand von 50 dB überlagert wurden

$\mathbf{A}_{3\,\mathrm{dB}}$ enthält $3 \cdot 6 = 18$ Audiosignale, bei dem alle sechs Baby-Aufnahmen mit den drei Rauschsignalen bei einem Signal-Rausch-Abstand von 3 dB überlagert wurden

$\mathbf{A}_{\mathbf{50+3\,dB}} = \{A_{50\,\mathrm{dB}} \cup A_{3\,\mathrm{dB}}\} = 32$ Audiosignale

$\mathbf{A}_{7\,\mathrm{dB}*}$ enthält 1 Audiosignal, bei dem eine siebte Baby-Aunfahme mit einem vierten Rauschsignal bei einem Signal-Rausch-Abstand von 7 dB überlagert wurde.

Im nächsten Schritt werden die eigentlichen Datensätze $D_{SNR,Feats}$ gebildet, in dem Audiosignale dieser Signalmengen (1) wie in Kapitel 3.3 beschrieben vorverarbeitet werden, (2) wie in Kapitel 3.4.1 in die Signalfenster à 25 ms zerlegt werden und (3) für jedes Signalfenster der durch Gleichung 3.14 definierte Featurevektoren berechnet wird. Außerdem wird jedem Featurevektor die Klasseninformation *Stimme/Stille* zugewiesen.

Es ist rechnerisch zu aufwendig, alle genannten Features in einem kontinuierlichen System zur Voice Activity Detection zu berechnen. Daher werden die Datensätze in Untermengen bezüglich der verwendeten Features eingeteilt. Das Ziel ist es, diejenige Untermenge an Features zu finden, die sich am besten für die Voice-Activity-Detection sowohl bei niedrigem als auch bei starkem Hintergrundrauschen eignet. Die Untermengen werden in Bezug auf die Methode gebildet, durch die die Features berechnet werden. Das heißt, dass beispielsweise die Untermenge *Zeit* die in Kapitel 3.4.2 beschriebenen Features *RMS* und *ZCR* sowie die dazugehören Differenzfeatures *Diff$_t$(RMS)* und *Diff$_t$(ZCR)* beinhaltet.

Die 9 Untermengen sind: { Zeitbereich, Frequenzbereich, Cepstrum, Autokorrelation, Zeit + Frequenzbereich, Zeit + Cepstrum, Zeit + Autokorrelation, Frequenz + Cepstrum, Frequenz + Autokorrelation }. Cepstrum- und Autokorrelation werden nicht gemeinsam in eine Untermenge hinzugefügt, da Sie die Rechnerisch aufwendigsten sind. So enthält beispielsweise der Datensatz $D_{3\,\mathrm{dB},Zeit}$ die Featurevektoren des Zeitbereiches für die Audiosignale mit einem Signal-Rausch-Abstand von 3 dB. Alle Audiosignal-Mengen [$\mathrm{A}_{50\,\mathrm{dB}}$], [$\mathrm{A}_{3\,\mathrm{dB}}$], [$\mathrm{A}_{50+3\,\mathrm{dB}}$] und [$\mathrm{A}_{7\,\mathrm{dB}}$] wurden in Datensätze umgewandelt. Es wurden schlussendlich $4 \cdot 9 = 36$ Datensätze gebildet.

Training

Das Ziel ist, mit Hilfe des C4.5-Algorithmus einen Entschidungsbaum zu finden, der auf Basis einer möglichst geringen Feature-Menge eine möglichst hohe Klassifkationsgenauigkeit für sowohl niedrige als auch hohe Signal/Rausch-Abstänge erzielt. Die Frage ist, ob ein Entschiedungsbaum, der auf Basis von Signalen mit einem niedrigen SNR gebildet wird, auch für hohe SNR eine hohe Klassifikationsgenauigkeiten erzielt, oder ob der umgedrehte Fall zutreffend ist. Daher werden die Entscheidungsbäume sowohl auf Basis verschiedener SNRs als auch verschiedener Feature-Untermengen gebildet. Die Entschäudungsbäume werden daraufhin gegen die Signale mit den verschiedenen SNRs evaluiert. Wird also beispielsweise der Datensatz $D_{50\,\mathrm{dB},Zeit}$ zum Training und der Datensatz $D_{3\,\mathrm{dB}}$ verwendet, so wird berechnet, wie gut sich der Klassifikator unter Verwendung der Zeit-Features zur Klassifizierung niedriger SNRs eignet, obwohl er für hohe SNRs entworfen wurde. Dabei ist unerheblich, welche Features der Test-Datensatz verwendet, da es bei der Evaluation nur auf die Klasseninformation der Instanzen ankommt.

Die Implementierung, die für den *C4.5* verwendet wurde, ist der *REPTree*-Algorithmus [2] der Open Source Data-Mining-Bibliothek *Weka*[3]. Die Implementierung hat den Vorteil, dass die maximale Tiefe des Entscheidungsbaumes festlegbar ist und somit die Komplexität des Baumes begrenzt werden kann und Overfitting vermieden wird.

Es wurden insgesamt $3 \cdot 9 = 27$ Trainings-Datensätze erzeugt ([3 SNR-Werte: 3 dB, 50 dB und 50+3 dB] × [9 Feature-Untermengen]. Der Datensatz mit einem SNR von 7 dB wurde

[2] Dokumentation von REPTree: `http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/REPTree.html`

[3] Download von WEKA: `http://www.cs.waikato.ac.nz/ml/weka/`

nicht zum Training verwendet). Mit diesen 27 Trainingsdatensätze wurden mit Hilfe des *REPTree*-Algorithmus 27 Klassifikationsbäume erzeugt. Jeder Klassifikationsbaum wurde gegen die 3 Testdatensätze $D_{3\,dB}$, $D_{50\,dB}$ und $D_{7\,dB^*}$ evaluiert und die Accuracy berechnet. Das Signal $A_{7\,dB^*}$ erfüllt dabei eine Sonderrolle, da es nicht in den Trainingsdatenstäzen enthalten war und somit der Kontrolle dient, ob Overfitting vorliegt. Da jeder Datensatz ungefähr dreimal mehr Stimmhafte Examples als nicht-Stimmhafte enhthielt, wurde jede Stimmlose Instanz eines Datensatzes eingefügt. Somit wurde für jeden Datensatz ein ausgewogenes Verhältnis zwischen positiven und negativen Examples gewährleistet. Um die Komplexitiät des Entscheidungsbaumes zu verringern eine Nutzung von möglichst wenig Features zur Klassifizierung zu erzwingen, wurde die maximale Tiefe des REPTree auf 2 gesetzt.

Ergebnis

Die Evaluations-Ergebnisse sind in Tabelle .1 zu sehen. Für jeden Trainingsdatensatz mit einem bestimmten SNR und einer Feature-Untermenge wird die Accuracy für den jeweilgen Test-Datensatz mit einem SNR von 3 dB, 50 dB und 7 dB* verwendet.[4]. Außerdem wird der Durchschnittswert aller drei Accuracy-Werte angegeben.

Die Features, welche zu den höchsten Accuracy-Werten führten, sind die des *Cepstrum*-Bereiches, genauer gesagt das $\mathrm{Diff_t(Ceps_{mag})}$-Feature, da es vom REPTree als einziges Feature dieses Bereiches für die Entscheidungsbäume ausgewählt wurde. Die Entscheidungsbäume, die mit dem $\mathrm{Diff_t(Ceps_{mag})}$-Feature entworfen wurden, erreichten eine durchschnittliche Accuracy von mindestens 91,45%. Der nächstbeste Entscheidungsbaum mit einer Accuracy von 86,96% wurde unter Verwendung der Features des Zeitbereiches und der Autokorrelation auf dem Datensatz $D_{50+3\,dB,Zeit+Correlation}$ entworfen. Sobald der Cepstrum-Bereich in Verbindung mit den Features anderer Bereiche verwendet wurde, wurde das $\mathrm{Diff_t(Ceps_{mag})}$-Feature vom REPTree-Algorithmus bevorzugt und die Features der anderen Bereiche nicht mehr verwendet.

Auf Basis der Datensätze $D_{3\,dB,Ceps}$, $D_{3\,dB,Zeit+Ceps}$, $D_{3\,dB,Freq+Ceps}$, $D_{50+3\,dB,Ceps}$, $D_{50+3\,dB,Zeit+Ceps}$ sowie $D_{50+3\,dB,Freq+Ceps}$ wurde der selbe Klassifikator erzeugt, der in Gleichung 3.15 zu definiert wird. Wie zu sehen ist, handelt es sich um einen einfachen Grenzwert des $v.Diff_t(Ceps_{mag})$-Features, da trotz der höchst möglichen Baumtiefe von 2 nur eine Tiefe von 1 genutzt wurde.

$$C(v_i) = \begin{cases} 1, & \text{if } v.Diff_t(Ceps_{mag}) > 0.02, \\ 0 & \text{else} \end{cases} \tag{3.15}$$

Auf Basis der Datensätze $D_{50\,dB,Ceps}$ und $D_{50\,dB,Zeit+Ceps}$ wurde der Klassifikator nach Gleichung 3.16 erzeugt. Er unterscheidet sich von dem Klassifikator aus Gleichung 3.15 nur durch den Grenzwert.

$$C(v_i) = \begin{cases} 1, & \text{if } v.Diff_t(Ceps_{mag}) > 0.03, \\ 0 & \text{else} \end{cases} \tag{3.16}$$

[4]Der Stern verdeutlicht die Sonderrolle des Datensatzes mit einem SNR von 7 dB, da er nur zu Evaluation verwendet wurde

Da der Klassifikator aus Gleichung 3.15 eine durchschnittliche Accuracy von 92,22% und der Klassifikator aus Gleichung 3.16 eine unwesentlich geringere Accuracy von 91,45% erzielt, wurden für beide Modelle die Specificity und Sensitivity berechnet, um eine Entscheidung für eines der beiden Modelle fällen zu können. Dazu wurden die Signalmengen $A_{3\,\mathrm{dB}}$, $A_{50\,\mathrm{dB}}$ und $A_{7\,\mathrm{dB}*}$ in Frames à 100 Windows zerlegt und für jedes Zeitfenster die Senstivity, Specificity und Accuracy bezüglich der beiden Klassifikatoren berechnet. Die Ergebnisse werden als Boxplots in Abbildung .1 dargestellt. Die Modelle unterscheiden sich am stärksten bezüglich der Datensätze mit 3 dB und 7 dB. Der Klassifikator mit dem Grenzwert von 0.03 erzielt in beiden Fällen eine höhere Specificity, aber geringere Senstivitiy als das Modell mit dem Grenzwert bei 0.02. Es wurde sich für das Modell für mit einem Grenzwert von 0.02 entschieden, da durch die höhere Senstivity mehr Cry-Units erkannt werden, die in späteren Verarbeitungsschritten immernoch als False-Positives erkannt und verworfen werden können. Einmal im Prozess der VAD als Stimmlos markierte Fenster werden jedoch nicht weiter verarbeitet und gehen somit „verloren“.

Der Finale Klassifikations-Funktion eines Signalfensters $C(x_i[\,])$ in $0 \longmapsto \textit{Stille}$ oder $1 \longmapsto \textit{Stimme}$ ist somit durch Gleichung 3.17 gegeben, wobei $c_i[\,]$ das Cepstrum des Signalfensters ist.

$$C(x_i[\,]) = \begin{cases} 1, & \text{if } v.Diff_t(Ceps_{mag}(c_i[\,])) > 0.02, \\ 0 & \text{else} \end{cases} \tag{3.17}$$

3.4.4 Markierung der Cry-Units

Wird die Voice-Activity-Detection für das Signal $x[\,]$ nach Gleichung 3.17 durchgeführt, ist das Ergebnis eine Zuordnung der Signalfenster $x_1[\,] \ldots x_n[\,]$ zu den Klassen $C(x_i[\,]) = 1$ *Stimme* oder $C(x_i[\,]) = 0$ *Stille*. Varallyay [32, S. 16 - 17] stellt die Idee vor, auf Grundlage der Informationen der Voice-Activity-Detection die Cry-Units zu extrahieren (welche er in seiner Publikation als Cry-Segmente beschreibt). Das genaue vorgehen konnte jedoch nicht eingesehen werden, da der Autor dieser Arbeit keine Zugriffsrechte auf die Publikation erhielt.

Waheed et al [17] stellen die Idee vor, zusammenhängende und ununterbrochene Ketten als *stimmhaft* klassifizierter Signalfenster zu *Stimm-Segmenten* zusammenzufassen. Dieser Ansatz wird übernommen, wobei ein Stimmsegment in dem Kontext dieser Arbeit einer *Cry-Units* entspricht. Möglicherweise ist dies der Ansatz, den auch Varallyay [32, S. 16 - 17] gewählt hat. Abbildung 3.8 veranschaulicht diese Gruppierung.

Formel 3.18 gibt die Definition des Datentypes *Cry-Unit* $[CU]$. Eine Cry-Unit wird definiert durch den Anfangszeitpunkt *start*, einen Endzeitpunkt *end* und der Liste seiner Signalfenster $windows = [x_1 ... x_m]$.

$$CU = (windows = [x_1 ... x_m], start \in Zeit, end \in Zeit) \tag{3.18}$$

Die Dauer eine Cry-Unit $cu \in CU$ wird nach Formel 3.19 berechnet und mit λ bezeichnet. Der (Stille)-Zeitraum zwischen zwei Cry-Units $\mathrm{d}(cu_i, cu_j)$, wird nach Formel 3.20 berechnet. Diese Zusammenhänge werden in Abbildung 3.9 visualisiert.[17, S. 2]

$$\lambda(cu) = cu.end - cu.start \tag{3.19}$$

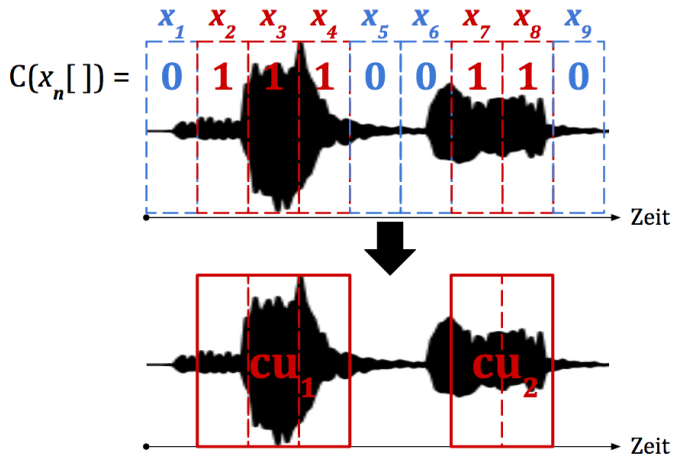

Abbildung 3.8: Zusammenfassung klassifizierter Signalfenster zu Cry-Units

$$\mathrm{d}(cu_i, cu_j) = cu_j.start - cu_i.end \tag{3.20}$$

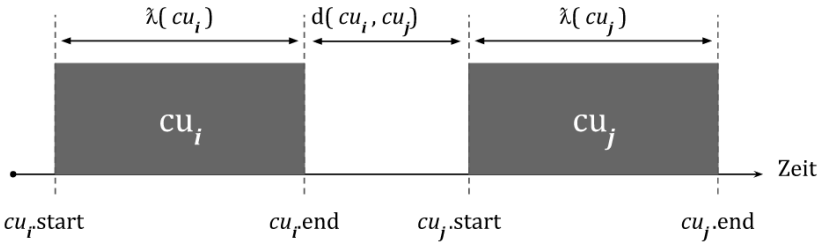

Abbildung 3.9: Beziehung zwischen agrenzenden Cry-Units, nach [17, S. 2]

Algorithmus 1 zeigt in Pseudo-Code, wie auf Basis der Liste aller Signalfenster eines Signals $X_{all} = [x_1[\,], \ldots, x_n[\,]]$ die Liste der Cry-Units $CU_{all} = [cu_1 ... cu_m]$ generiert wird. Die Funktion $C(x)$ ist die Klassifikations-Funktion der Signalfenster in Stille/Stimme nach Gleichung 3.17. Die Funktion getTimeOf($x_i[\,]$) liefert die Anfangszeitpunkt des Signalfensters $x_i[\,]$.

3.4.5 Decision Smoothing

Abbildung 3.10 zeigt ein Audiosignal mit einem Signal-Rausch-Abstand von 3 dB, bei dem die Klassifikation nach Gleichung 3.17 durchgeführt wurde. Die rote Linie zeigt die tatsächliche Klassifizierung und die grüne Linie die prognostizierte Klassifizierung. Es ist zu sehen, dass False-Negatives und False-Positives in der Klassifizierung enthalten sind. Im folgenden werden drei charakteristische Arten falscher Klassifikationen näher erläutert:

False Negatives nach (a) : Eine korrekt erkannte, längere Cry-Unit wird zu früh beendet. Oft werden kurz nach dem Ende einer längeren Cry-Unit sehr kurze Cry-Units erkannt, die eigentlich noch zu der längeren, vorhergehenden Cry-Unit gehören.

False Positives nach (b): Kurze Cry-Units werden in eigentlichen Stille-Bereichen erkannt.

Algorithm 1 Gruppierung von Signalfenstern zu Cry-Units

```
1: function turnWindowsIntoCryUnits(X_all)
2:     CU_all ← [ ]
3:     cu ← ([ ], 0, 0)
4:     for i = 1, ..., length(X_all) do
5:         c_i ← C(x_i[ ])
6:                                                  ▷ Start of Cry-Unit
7:         if c_i == 1 ∧ isEmpty(cu_j.windows) then
8:             cu ← ([ ], 0, 0)
9:             cu.start ← getTimeOf(x_i[ ])
10:            cu.windows ← [cu.windows, x_i[ ]]
11:        end if
12:                                                 ▷ Inside Cry-Unit
13:        if c_i == 1 ∧ ! isEmpty(cu.windows) then
14:            cu.windows ← [cu.windows, x_i[ ]]
15:        end if
16:                                                 ▷ End of Cry-Unit
17:        if c_i == 0 ∧ ! isEmpty(cu.windows) then
18:            cu.end ← getTimeOf(x_i[ ])
19:            CU ← [CU, cu]
20:            cu.windows ← [ ]
21:        end if
22:    end for
23:                                  ▷ End last Cry-Unit by force if is still open.
24:    if ! isEmpty(cu.windows) == 0 then
25:        cu.end ← getTimeOf(X_windows[end])
26:        CU_all ← [CU_all, cu]
27:    end if
       return CU_all
28: end function
```

False Negatives nach (c): Eine Cry-Unit zerfällt in zwei Cry-Units, da Signalfenster in der Mitte als Stille erkannt wurden.

Im Process des **Decision Smoothing** werden kontextuelle Informationen genutzt, um nachträglich False-Positives und False-Negatives zu entfernen. Es werden dazu die von Waheed et al [17] präsentierten Ideen verwendet. Es werden zwei Parameter eingeführt: λ_{min}, die Mindestlänge einer akzeptierten Cry-Unit, und d_{min}, die Mindestlänge eines akzeptierten Stille-Segmentes. Das Decision Smoothing wird nach den folgenden Entscheidungsregeln durchgeführt:

- ist $\lambda(cu_i) \leq \lambda_{min}$?
  - wenn $\lambda(cu_{i-1}) > \lambda_{min}$ und $d(cu_{i-1}, cu_i) \leq d_{min}$, dann vereinige CU_i mit CU_{i-1} . $\Rightarrow$ behebt False-Negatives des Types (a)
  - ansonsten entferne cu_i $\Rightarrow$ behebt False-Negatives des Types (b)

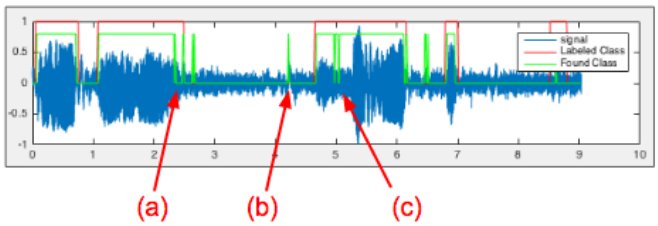

Abbildung 3.10: Klassifizierung vor dem Decision Smoothing

- wenn $\lambda(cu_i) > \lambda_{min}$ und $d(cu_{i-1}, cu_i) \leq d_{min}$, dann vereinige cu_i mit cu_{i-1} . $\Rightarrow$ behebt False-Negatives des Types (c)

Die Entscheidungsregeln greifen nur auf die letzten beiden erkannten Cry-Units zu, um eine kontinuierliche Analyse zu gewährleisten. Bei einer kontinuierlichen Analyse wird die Auswertung um die Zeitdauer einer Cry-Unit verzögert, da die Entscheidungsregeln erst nach Beendigung einer Cry-Unit abgefragt werden können. Bei einer offline-Analyse können die Entscheidungsregeln vereinfacht werden, da die False-Negatives nach Typ (a) und (c) mit der selben Regel abgefragt werden können. Algorithmus 2 zeigt in Pseudo-Code, wie das Decision-Smoothing durchgeführt wird. Input der Funktion ist die Liste aller Cry-Units CU_{all}, die durch Algorithmus 1 entstanden ist, sowie die Grenzwerte λ_{min}, d_{min}. Ausgang der Funktion ist die Liste aller Cry-Units nach dem Decision-Smoothing $CU_{smoothed}$.

Abbildung 3.11 zeigt das Beispielsignal vor und nach dem Decision-Smoothing. In verschiedenen Veröffentlichungen wurden unterschiedliche Mindestlängen von Cry-Units festgestellt. Varallyay [32, S. 8] hat eine Mindestlänge von 250 ms gemessen. Der geringste Wert, der nach dem Wissen des Autors dieser Arbeit in einer Veröffentlichung genannt wurde, stammt von Zeskind et al [24, S. 325] und beträgt 60 ms, welcher für $\lambda - min$ übernommen wurde. Es konnten hingegen keine Werte über die geringste festgestellte Pause zwischen zwei Cry-Units gefunden werden. Der Wert wurde daher auf Basis des verwendeten Trainings-Datensatzes experimentell ebenfalls mit $d_{min} = 60$ ms bestimmt.

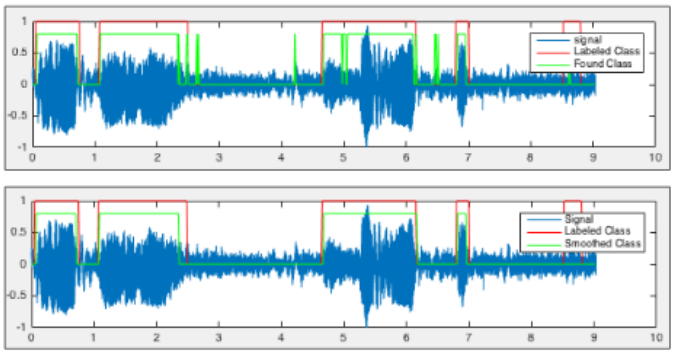

Abbildung 3.11: Klassifizierung vor und nach dem Decision Smoothing

3.4.6 Diskussion der Voice-Activity-Detection

In diesem Kapitel wurden verschiedene Varianten der Voice-Activity-Detection vorgestellt, verglichen und evaluiert, wobei eine Voice-Activity-Detection auf Basis des Cepstrums

Algorithm 2 Decision-Smoothing for VAD

function DECISIONSMOOTHING($CU_{all}, \lambda_{min}, d_{min}$)
  $CU_{smoothed} \leftarrow [CU_{all}[1]]$
  for $i = 2, \ldots, length(CU_{all})$ **do**
    $cu_i \leftarrow CU_{all}[i]$
    $cu_{i-1} \leftarrow CU_{smoothed}[end]$
    if $\lambda(cu_i) > \lambda_{min}$ **then**
                                                ▷ Accept Cry-Unit
      if $\mathrm{d}(cu_{i-1}, cu_i) > d_{min}$ **then**
        $CU_{smoothed} \leftarrow [CU_{smoothed}, cu_i]$
      else
                                                ▷ Erase False-Negative Type (c)
        $cu_i \leftarrow \text{vereinige}(cu_i, cu_{i-1})$
        $CU_{smoothed} \leftarrow [CU_{smoothed}[1 : end - 1], cu_i]$
      end if
    else
                                                ▷ Erase False-Negative Type (a)
      if $d(cu_{i-1}, cu_i) \leq d_{min}$ **then**
        $cu_i \leftarrow \text{vereinige}(cu_i, cu_{i-1})$
        $CU_{smoothed} \leftarrow [CU_{smoothed}[1 : end - 1], cu_i]$
      else
                                                ▷ Don't accept cu_i. Erases False-Positives (b)
      end if
    end if
  end for
  return $CU_{smoothed}$
end function

die besten Ergebnisse erzielt hat. Die Voice-Activity-Detection betrachtet kontextuelle Informationen in Bezug auf den zeitlichen Verlauf jedoch nur in einem geringen Maße beim Decision-Smoothing. Schlussendlich markiert der VAD-Algorithmus eine Reihe von kurzen Signalfenstern genau dann als zusammenhängende Cry-Unit, wenn jedes Signalfenster für sich betrachtet als Lautäußerung eines Babies klassifiziert wurde. Ob jedoch die Reihenfolge der in den Signalfenstern enthaltenen Lautäußerungen Sinn macht, wird nicht betrachtet. Schneidet man beispielsweise wenige Sekunden aus der Mitte einer längeren Cry-Unit aus und konkateniert dieses Sample viele Male, um eine synthetische, längere Cry-Unit zu erzeugen, klingt das Ergebnis für den Menschen stark unnatürlich, wird von dem hier vorgestellten VAD-Algorithmus jedoch trotzdem als valide Cry-Unit markiert. Das Cepstrum als Feature mit der höchsten Accuracy ist somit so zu bewerten, dass es vor allem im geringen Maße kontextuell Informationen benötigt, um eine Entscheidung über das vorahndensein von Stimme zu fällen. Zukünftige Forschungen könnnen an diesem Punkt ansetzen, um die Accuracy der VAD zu erhöhen.

3.5 Segmentierung

Das Ergebnis der Voice-Activiy-Detection ist eine Liste an Cry-Units $cu_1 ... cu_n$. Pain-Scores werden nicht aus einzelnen Cry-Units abgeleitet, sondern aus dem Verbund mehrerer Cry-Units. Daher ist es notwendig, die Cry-Units zu Cry-Segmenten zusammenzufassen. Dieser Prozess des Zusammenfassens von Cry-Units zu Segmenten wird in dieser Arbeit kurz als *Segmentierung* bezeichnet. Die Frage ist, nach welchen Kriterien Cry-Units zu Segmenten zusammengefasst werden. Abbildung 3.12 verdeutlicht das Problem, in dem drei mögliche Segmentierungen für eine Signal beispielhaft gezeigt werden.

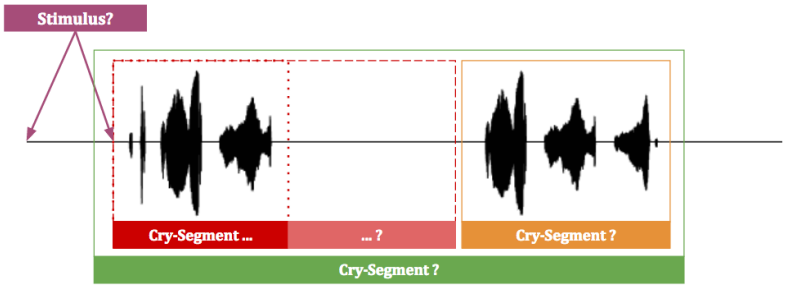

Abbildung 3.12: Mögliche Segmentierungen eines Signals

Ein Cry-Segment wird von Golub et al definiert als „die komplette klangliche Antwort auf einen spezifischen Stimulus. Sie kann mehrere Cry-Units enthalten “. [12, S. 61, übersetzt aus dem Englischen]. Die Defintion lässt folgende Fragen offen:

- Beginnt das Segment bereits bei Zuführung des Stimulus, oder erst ab der ersten Cry-Unit?
- Wodurch definiert sich der Beginn, wenn der Stimulus unbekannt ist?
- Endet ein Cry-Segment mit Ende der letzten „Cry-Unit“, oder erstreckt es sich bis zu Beginn des nächsten Cry-Segmentes?

Keines der in Kapitel 3.1 vorgestellten Veröffentlichungen schlägt Methoden zur Segmentierung vor. Bei den nicht-kontinuierlichen Systemen werden manuell geschnittene Cry-Segmente verwendet. Entweder werden keine objektiv messbaren Krtierien gegeben zur Festlegun der länge dieser Cry-Segmente gegeben, oder feste Längen wie zum Beispiel 90 s[24, S. 324] gegeben. Bei den kontinuierlichen Systemen wird die Segmentierung nicht als Verarbeitungsschritt erwähnt.

Es wird daher das folgende Vorgehen zur kontinuierlichen Segmentierung vorgestellt: Wenn das Baby keine Äußerungen von sich gibt, weil es beispielsweise schläft, wird keine Cry-Unit festgestellt, und somit existiert auch momentan kein offenes Segment. Fängt das Baby an, Laute von sich zu geben, also eine Cry-Unit zu produzieren, wird ein neues Segment eröffnet und die Cry-Unit diesem Segment hinzugefügt. Weitere Cry-Units werden so lange diesem Segment hinzugefügt, wie die Dauer der Stille nach einer Cry-Unit einen festgelegten Grenzwert t_s nicht überschreitet. Ein Cry-Segment wird folglich dann geschlossen, wenn das Baby „aufhört, zu weinen“, also keine Laute mehr für einen festgelegten Zeitraum von sich gibt. Das Endzeitpunkt des Segmentes wird als der Endzeitpunkt der letzten Cry-Unit des Segmentes festgelegt.

Formel 3.21 definiert ein *Cry-Segment* $[CS]$ als Datentyp. Ein Cry-Segment ist eine Liste

von Cry-Units. Alle Cry-Units erfüllen die Nebenbedingung 3.22, das heißt, dass die Distanz aller benachbarter Cry-Units eines Cry-Segments unterhalb des Grenzwertes t_s liegen.

$$CS = [cu_1, \ldots, cu_n] \tag{3.21}$$

$$\forall cs \in CS : \forall i = 1 \ldots length(cs) - 1 : d(cs[i], cs[i+1]) < t_s \tag{3.22}$$

Der Start-Zeitpunkt eines Cry-Segmentes wird nach Formel 3.23 als der Startzeitpunkt der ersten Cry-Unit des Segmentes definiert. Das Ende eines Segmentes wird definiert als das Ende der letzten Cry-Unit nach Gleichung 3.24.

$$start(CS) = CS[1].start \tag{3.23}$$

$$end(CS) = CS[end].end \tag{3.24}$$

Algorithmus 3 zeigt einen Pseudocode, wie die Segmentierung nach dem beschriebenen Prinzipien offline durchgeführt wird. Input des Algorithmus ist die Liste aller Cry-Units $CU_{all} = [cu_1 ... cu_n]$, die nach dem Decision-Smoothing nach Algorithmus 2 entstanden ist. Das Ergebnis des Algorithmus ist die Liste, die alle gefundene Cry-Segmente $[cs_1 ... cs_m]$ enthält. Der Algorithmus eignet sich nicht für eine Online-Segmentierung, da das Ende eines Segmentes erst nach dem Abschluss einer Cry-Unit festgestellt wird, wobei beliebig viel Zeit zwischen zwei Cry-Units liegen kann. Bei einer online durchgeführten Segmentierung empfiehlt es sich, ein Segment sofort zu beenden, wenn der Zeitraum der Stille nach einem Segment den Grenzwert t_s überschreitet. Abbildung **??** die Segmentierung anhand eines Beispiels.

Algorithm 3 Gruppierung von Cry-Units zu Cry-Segments

```
1:  function segmentCryUnits(CU_all, t_s)
2:      CS_all ← []
3:      cs_i ← [CU_all[1]]
4:      for i = 2...length(CU_all) do
5:          cu_i ← CU_all[i]
6:          cu_{i-1} ← CU_all[i − 1]
7:          if d(cu_{i-1}, cu_i) < t_{seg-max} then
8:              cs_i ← [cs_i, cu_i]
9:          else
10:             CS_all ← [CS_all, cs_i]
11:             cs_i ← [cu_i]
12:         end if
13:     end forreturn CS_all
14: end function
```

Das hier vorgestellte Vorgehen ist absichtlich trivial gehalten, damit der Sinn des Parameters t_s leicht ersichtlich ist und somit von der medizinischen Fachkracht selbständig festgelegt werden kann. Schlussendlich ist diese Segmentierung ist eines der Hauptziele dieser Segmentierung, die unnötige Berechnung von Schmerzscores in den nachfolgenden Schritten

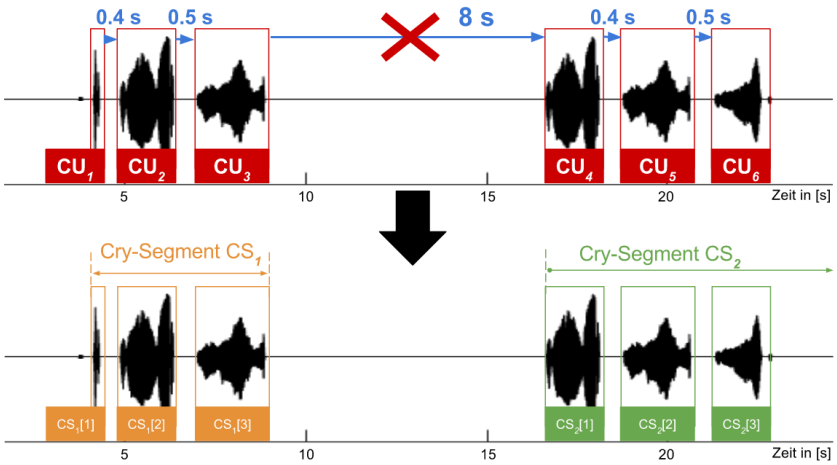

Abbildung 3.13: Ergebnis der Segmentierung

zu vermeiden, so lange keine Cry-Units vorliegen. Trotz der Trivialität dieser laufenden Segmentierung liegt hier ein wichtiger Unterschied im Gegensatz zu vergleichbaren Systemen, wie zum Beispiel das von Cohen et al [5], bei dem die Ableitung von Weinen/nicht-Weinen für Segmente mit einer festen Fenstergröße von 10 Sekunden vorgenommen wird.

3.6 Feature-Extraktion und Ableitung der Schmerzscore

Das Erebnis der Segmentierung ist eine Litse an Cry-Segmenten $cs_1, \ldots, c_n$. Diese Cry-Segmente bilden nun die Basis für die Ableitung der Pain-Score. Die medizinische Fachkraft muss dabei zuerst die Wahl treffen, welche Pain-Scale verwendet werden soll. Das einfachste denkbare Vorgehen ist die Ableitung genau einer Punktzahl aus den insgesamten Eigenschaften eines Segmentes, wobei diese Ableitung erst vollzogen werden kann, sobald ein Segment abgeschlossen wurde und alle Informationen für dieses Segment vorliegen. Es wird also jedem Segment genau eine Punktzahl zugewiesen. Das Vorgehen wird am Beispiel der NIPS aus Tabelle 2.1 verdeutlicht: Dabei steht die Abwesenheit von Weinen für null Punkte, „mumbling“ (murmeln) für einen Punkt und „vigorous“ (energisch) für zwei Punkte. Die Abwesenheit von Lautäußerungen, also der Zeitraum zwischen den Segmenten, bekommt somit null Punkte. Ein Segment, dessen Qualität insgesamt als „murmelnd“ bewertet wird, bekommt somit einen Punkt, und ein Segment, welches als insgesamt als „energisch“ bewertet wird, zwei Punkte. Das Problem ist offensichtlich: „murmelnd“ und „energisch“ sind subjektiv behaftete Begriffe und lassen sich nicht ohne weiteres feststellen aus den Eigenschaften eines Segmentes feststellen.

Um Unklarheiten zu vermeiden, wird an dieser Stelle noch einmal darauf hingewiesen, dass mit „Pain-Scale“ eine Scale, wie zum Beispiel die NIPS gemeint ist, und mit „Pain-Score„ oder einfach nur „Score“ die vergebene Punktzahl.

Es werden zwei verschiedene Lösungs-Strategien für dieses Problem geschildert. Strategie 1 löst das Problem mit Hilfe von *Regression* (Siehe Kapitel **??**):

1. Man erstellt eine Datenbank mit Aufnahmen von kindlichen Lautäußerungen, die man Segmentiert.

2. Man errechnet so viele *objektiv* messabare Eigenschaften wie möglich für jedes Segment, wie zum Beispiel die insgesamte Länge, die durchschnittliche Länge der enthaltenen Cry-Units, durchschnittliche Tonhöhe usw.,
3. Man bittet medizinische Fachkräfte, für jedes Segment der Datenbank eine Score bezüglich einer Pain-Scale zu vergeben. Dadurch erhält man eine gelabelte Test-Datenbank.
4. Man verwendet einen *Regressionsalgorithmus*, um den Zusammenhang zwischen den in Schritt 2 objektiv gemessenen Eigenschaften der Segmente und den in Schritt 3 vergebenen *Scores* herzustellen. An dieser Stelle kann zum Beispiel die in Kapitel **??** beschriebene multiple lineare Regression verwendet werden. Man erhält somit einen Regressor für jede Pain-Scale.
5. Möchte man für neue, unbekannte Segmente die Pain-Score ableiten, nutzt man den entsprechenden Regressor.

Das Vorteil dieses Vorgehens ist, dass das Problem der Übersetzung der objektiv messbaren Parameter in die subjektiv behafteten Begriffe überbrückt wird, indem die Regression direkt von den objektiv messbaren Parametern auf eine Punktzahl durchgeführt wird. Der Nachteil ist, dass eine Testdatenbank für jede Pain-Scale aufgebaut werden muss. Wird ein neue Pain-Scale eingeführt, muss der Regressor für diese Scale durch erneutes Labeln festgestellt werden. Ein weiterer Effekt der Abbildung des Problems als Regression ist, dass ein Regressor in einen kontinuierlichen Zahlenraum abbildet. Es sind also Regressionsergebnisse wie zum Beispiel 2.8 denkbar. Diese „bessere Auflösung“ kann als Vorteil gesehen werden. Ist jedoch eine direkte Übersetzung der Pain-Scale inklusive der ganzzahligen Punktzahlen gewünscht, so stellt sich die Frage, ob eine 2.8 auf- oder abgerundet wird.

Strategie 2 löst das Problem mit Hilfe von Klassifizierung (Siehe Kapitel **??**):

1. und 2. entsprechen Strategie 1
3. Man sammelt alle subektiven Begriffe, die in Pain-Scales verwendet werden, wie zum Beispiel „murmeln“, „energisch“, usw.
4. Man bittet medizinische Fachkräfte, jedes Segment der Datenbann mit denjenigen Begriffen zu labeln, die die jeweilige Person für zutreffend hält.
5. Man Verwendet einen *Klassifizierungsgorithmus*, um einen Zusammenhang zwischen den in Schritt 2 festgestellten objektiv messbaren Eigenschaften der Segmente und den *subjektiv behafteten Begriffen* zu finden. Man erhält somit einen Klassifikator für jedenBegriff, der binär in *positive = zutreffend* und *negative = nicht zutreffend* klassifiziert.
6. Möchte man für neue, unbekannte Segmente die Pain-Score ableiten, so wird für jede Puntkzahl der Pain-Scale überprüft, ob für alle subjektiv beschreibenden Begriffe der entsprechende Klassifikator ein positive prognostiziert. Die Ableitung Score ist somit ein weiters Klassifikationsproblem, wobei eine Score einer Klasse entspricht und genau dann abgeleitet werden kann, wenn alle Vorraussetzungen für die Klasse erfüllt sind.

Der Vorteil dieser Methode ist, dass auch zum Zeitpunkt der Erstellung der Testdatenbank unbekannte Pain-Scores zu einem späteren Zeitpunkt eingebunden werden können, insofern alle in dieser neuen Pain-Scale verwendeten subjektiv behafteten Begriffe bereits gelabelt vorliegen, weil sie auch in anderen Pain-Scales verwendet werden. Das Vorgehen erlaubt somit eine gewissen Flexibilität bezüglich zukünftig entwickelter Pain-Scales. Der Nachteil

dieser Methode ist, dass durch die Umwandlung der eigentlich quantitativ geordenten Punktzahl einer Pain-Scale in qualitative Klassen aus einem implizit als Regression zu betrachtenden Problem ein Klassifizierungsproblem macht. Dies wirft neue Fragen auf, wie zum Beispiel die folgende: Angenommen, in einer Pain-Scale wird jede Score mit jeweils drei subjektiven Begriffen beschrieben, und bei der Klassifizierung eines Segmentes wird festgestellt, dass für jede Punktzahl genau zwei der drei Begriffe erfüllt werden. Welche Score wird dann abeleitet? Ein anderes Beispiel wird am Beispiel der der NIPS-Score aus Tabelle 2.1 verdeutlicht: Angenommen, ein Cry-Segment enthält hörbar „starkes“ Schreien, es kann jedoch weder „mumbling = murmelnd “ noch „vigorous = energisch“ abgeleitet werden. Demzufolgen müsste dieses Segment eine Score von 0 Punkten erhalten, wobei ein Mensch in dieser Situation eventuell „stark“ zu „heftig“ uminterpretieren und 2 Punkte vergeben würde. Strategie 1 ist weniger anfällig für dieses Problem.

In jedem Fall werden medizinische Fachkräfte benötigt, um das Labeling der Cry-Segmente durchzuführen, was aus Zeitgründen im Rahmen dieser Arbeit nicht möglich ist. Die Aquise von Audioaufnahmen von Babie, und das Labeling der Aufanhmen erfodern nicht nur Zeit, sondern das Fachwissen über das Führen und die Auswerten von Interviews.

3.6.1 Feature-Extraction

Im vergangenen Kapitel wurde besprochen, wie die Basis für die Ableitung einer Pain-Score für ein Segment die

3.6.2 Ableitung der Pain-Score

Zu Beginn von Kapitel 3.6 wurde gesagt, dass genau eine Score für ein Segment abgeleitet wird. Diese Aussage wurde getroffen, da dies der einfachste denkbare Fall ist. Dieses Vorgehen hat zwei Nachteile: 1. Kann die Score erst nach der Beendigung eines Segmentes abgeleitet werden, was für einige Anwendungsfälle eventuell zu spät ist. So ist es eventuell notwendig, bereits eine Score abzuleiten, bevor das Segment beendet wurde, um zum Beispiel das schnelle Reagieren auf akuten und starken Schmerz zu ermöglichen. 2. Ist es denkbar, dass sich der Schmerz innerhalb eines Segmentes verändert und eventuell stärker oder schwächer wird.

Eine Lösung ist, bei einem momentan offenen Segment in regelmäßigen Abständen die Eigenschaften abzufragen und direkt die Pain-Score abzuleiten, um Zwischenergebnisse zu erhalten. Der este Parameter, der dafür eingeführt werden muss, ist die Häufigkeit, mit der diese „Aktualisierung“ durchgeführt werden soll Der am häufigsten umsetzbare Fall ist, ein Aktualisierung nach jeder neu dem Segment hinzgefügten Cry-Unit vorzunehmen. Der am wenigsten häufige Fall ist der bereits genannte einfachste, die Aktualisierung erst bei Beedingung eines Segmentes durchzuführen. Wird die Entscheidung der medizinischen Fachkraft überlassen, empfiehlt es sich, den Parameter möglichst einfach verstehbar zu machen, in dem man einen festen Intervall t_{act} festlegen lässt. Ein t_{act} von beispielsweise 10 s bedeutet, dass alle 10 Sekunden ein neuer Pain-Score für ein Segment berechnet wird. Die Beendigung eines Segmentes würde in jedem Fall eine Ableitung der Pain-Score auslösen und einen „erzwungenen Aktualisierungszeitpunkt“ darstellen. Es ist denkbar, den Aktualisierungsintervall fest an eine Pain-Scale zu binden. Die CRIES-Scale ist beispielsweise für das post-operative Monitoring gedacht und benötigt somit möglicherweise weniger

häufige Aktualisierungen als der DAN, welcher ebenfalls zur Schmerzdiagnostik während einer Operation eingesetzt werden kann. [20, S. 98]

Der zweite Parameter, der eingeführt werden muss, ist der Zeitraum, für den die Pain-Score berechnet wird, also der *Beobachtungszeitraum*. Es gibt Eigenschaften, die sich implizit auf den Zeitraum *Beginn des Segmentes* bis *Aktualisierungs-Zeitpunkt* beziehen, wie beispielsweise die *Länge des Segmentes*. Dieser Zeitraum gleichzeitig der längst mögliche Zeitraum innerhalb eines Segmentes, insofern keine Pain-Scores über mehrere Segmente hinweg abgeleitet werden sollen. Es ist jedoch auch möglich, kürzere Beobachtugnszeiträume zu wählen. Die in Kapitel 2.1 geben wenig Aufschluss über „typische

3.7 Visualisierung

4 Zusammenfassung

Literaturverzeichnis

[1] K J S Anand. *Pain in Neonates and Infants*. Elsevier, 2007.

[2] Zachariah Boukydis Barry Lester. *Infant Crying: Theoretical and Research Perspectives*. Springer, 1985.

[3] Judy Bildner. *CRIES Instrument Assessment Tool of Pain in Neonates*. City of Hope Pain, 1997. Online unter `http://prc.coh.org/pdf/CRIES.pdf`.

[4] R Sisto & Giuseppe Buonocore Carlo Bellieni, Franco Bagnoli. Cry features reflect pain intensity in term newborns: An alarm threshold. *Pediatric Research*, 5:142–146, 1. Online unter `https://www.researchgate.net/publication/297827342_Cry_features_reflect_pain_intensity_in_term_newborns_An_alarm_threshold`.

[5] Rami Cohen and Yizhar Lavner. Infant Cry Analysis and Detection. In *27th Convention of Electrical and Electronics Engineers in Israel*. IEEE, 2012. Online unter `https://www.researchgate.net/publication/261116332_Infant_cry_analysis_and_detection`.

[6] H. Hollien & T Murry E Müller. Perceptual responses to infant crying: identification of cry types. *Journal of Child Language*, 1(1):89–95, 1974. Online unter `https://www.cambridge.org/core/journals/journal-of-child-language/article/perceptual-responses-to-infant-crying-identification-of-cry-types/4F0F8088116FCE381851D8D560697A5F`.

[7] B. Simak E. Verteletskaya. Performance Evaluation of Pitch Detection Algorithms, 2009. Online unter `http://access.feld.cvut.cz/view.php?cisloclanku=2009060001`.

[8] Jan Hamers & Peter Gessler Eva Cignac, Romano Mueller. Pain assessment in the neonate using the Bernese Pain Scale for Neonates. *Early Human Development*, 78(2):125–131, 2004. Online unter `http://www.sciencedirect.com/science/article/pii/S0378378204000337`.

[9] Dmitry Goldgof Rangachar Kasturi Terri Ashmeade Ghada Zamzmi, Chih-Yun Pai and Yu Sun. An Approach for Automated Multimodal Analysis of Infants' Pain. In *23rd International Conference on Pattern Recognition*, Cancun, Mexico, 2016.

[10] Health Facts For You. *Using Pediatric Pain Scales Neonatal Infant Pain Scale (NIPS)*, 2014. Online unter `https://www.uwhealth.org/healthfacts/parenting/7711.pdf`.

[11] Hodgkinson. Neonatal Pain Assessment Tool , 2012. Online unter `http://www.rch.org.au/uploadedFiles/Main/Content/rchcpg/hospital_clinical_guideline_index/PAT%20score%20update.pdf`.

[12] Michael J Corwin Howard L Golub. A Physioacoustic Model of the Infant Cry. In *Infant Crying - Theoretical and Research Perspectives*, chapter 3, pages 59 – 82. Plenung, 1985.

[13] Donna Geiss Laura Wozniak & Charles Hall Ivan Hand, Lawrence Noble. COVERS Neo-

natal Pain Scale: Development and Validation. *International Journal of Pediatrics*, 2010, 2010. Online unter https://www.hindawi.com/journals/ijpedi/2010/496719/.

[14] J Gorriz & J Segura J Ramorez. Voice Activity Detection. Fundamentals and Speech Recognition System Robustness. *Robust Speech Recongnition and Understanding*, page 460, 2007. Online unter http://cdn.intechopen.com/pdfs/104/InTech-Voice_activity_detection_fundamentals_and_speech_recognition_system_robustness.pdf.

[15] Jeih-weih Hung & Lin-shan Lee Jia-lin Shen. Robust Entropy-based Endpoint Detection for Speech Recognition in Noisy Environments. 1998. Online unter https://www.researchgate.net/publication/221489354_Robust_entropy-based_endpoint_detection_for_speech_recognition_in_noisy_environments.

[16] Carol Espy-Wilson & Tarun Pruthi Jonathan Kola. Voice Activity Detection. *MERIT BIEN*, 2011. Online unter http://www.ece.umd.edu/merit/archives/merit2011/merit_fair11_reports/report_Kola.pdf.

[17] Kim Weaver & Fathi M. Salam Khurram Waheed. A robust Algorithm for detecting speech segments using an entropic contrast. *IEEE*, 2003. Online unter http://ieeexplore.ieee.org/document/1187039/.

[18] M M Homayounpour M H Moattar. A simple but efficient real-time Voice Activity Detection Algorithm. Signal Processing Conference, IEEE, August 2009. Online unter http://ieeexplore.ieee.org/document/7077834/?arnumber=7077834&tag=1.

[19] Hans M Koot Dick Tibboel Jan Passchier & Hugo Duivenvoorden Monique van Dijk, Josien de Boer. The reliability and validity of the COMFORT scale as a postoperative pain instrument in 0 to 3-year-old infants. *Pain*, 84(2):367—-377, 2000. Online unter http://www.sciencedirect.com/science/article/pii/S0304395999002390.

[20] Sinno Simons Monique van Dijk and Dick Tibboel. Pain assessment in neonates. *Paediatric and Perinatal Drug Therapy*, 6(2):97–103, 2004. Online unter http://www.sciencedirect.com/science/article/pii/S0304395999002390.

[21] Taddio Nulman. A revised measure of acute pain in infants. *J Pain Symptom Manage*, 10:456–463, 1995. Online unter http://geriatricphysio.yolasite.com/resources/Modified%20Behavioral%20Pain%20Scale%20(MBPS)%20in%20infants.pdf.

[22] J L Mathew P J Mathew. Assessment and management of pain in infants. *Postgrad Med J*, 79:438–443, 2003. Online unter http://pmj.bmj.com/content/79/934/438.full.

[23] Steven Creech & Marc Weiss. Patricia Hummel, Mary Puchalski. N-PASS: Neonatal Pain, Agitation and Sedation Scale – Reliability and Validity. *Pediatrics/Neonatology*, 2(6), 2004. Online unter http://www.anestesiarianimazione.com/2004/06c.asp.

[24] Susan Parker-Price & Ronald Barr Philip Zeskind. Rythmic organization of the Sound of Infant Cry. *Dev Psychobiol*, 26(6):321–333, 1993. Online unter https://www.ncbi.nlm.nih.gov/pubmed/8119482.

[25] Ananth N. Iyer Pritam Pal and Robert E. Yantorno. Emotion detection from infant facial experessions and cries. In *Acoustics, Speech and Signal Processing*. IEEE, 2006.

[26] R Ward & C Laszlo Qiaobing Xie. Automatic Assessment of Infants' Levels-of-Distress from the Cry Signals. *IEEE Transanctions on Speech and Audio Processing*, 4(4):253–265, 1996. Online unter http://ieeexplore.ieee.org/document/506929/.

[27] Brian Hopkins & James Green Ronald Barr. *Crying as a Sign, a Symptom, and a Signal*. Mac Keith Press, 2000.

[28] J R Shayevitz & Shobha Malviya Sandra Merkel, Terri Voepel-Lewis. The FLACC: A Behavioral Scale for Scoring Postoperative Pain in Young Children. *Pediatric Nursing*, 23(3):293–7, 1996. Online unter `https://www.researchgate.net/publication/13998379_The_FLACC_A_Behavioral_Scale_for_Scoring_Postoperative_Pain_in_Young_Children`.

[29] Andreas Spanias Sassan Ahmadi. Cepstrum-Based Pitch Detection Using a New Statistical V/UV Classification Algorithm. *IEEE Transactions on Speech and Audio Detection*, 7(3):333–338, 1999. Online unter `http://ieeexplore.ieee.org/document/759042/`.

[30] Henning Reetz & Carla Wegener Tanja Fuhr. Comparison of Supervised-learning Models for Infant Cry Classification. *InternatIonAl Journal of HealtH Professions*, 2015. Online unter `https://www.degruyter.com/view/j/ijhp.2015.2.issue-1/ijhp-2015-0005/ijhp-2015-0005.xml`.

[31] Sabine Deligne & Peder Olsen Trausti Kristjansson. Voicing Features for Robust Speech Detection. In *Interspeech Lisboa*, September 2005. Online unter `http://papers.traustikristjansson.info/wp-content/uploads/2011/07/KristjanssonRobustVoicingEurospeech2005.pdf`.

[32] Gyorgy Ivan Varallyay. *Analysis of the Infant Cry with Objective Methods*. PhD thesis, Budapest University of Technology and Economics, 2009. Online erhältlich unter: `https://pdfs.semanticscholar.org/5c38/b368dc71d67cbfabc3077a50536b086d8eec.pdf`.

[33] P H Wolff. The role of biological rhythms in early psychological development. *Bulletin of the Menninger Clinic*, 31:197–218, 1967.

[34] Syed Ahmad Yousra Abdulaziz, Sharrifah Mumtazah. Infant Cry Recognition System: A Comparison of System Performance based on Mel Frequency and Linear Prediction Coefficients. In *Information Retrieval & Knowledge Management*, 2010. Online unter `http://ieeexplore.ieee.org/document/5466907/`.

Appendices

Tabelle .1: Accuracy-Werte der Grenzwertfindung mit REPTree

$SNR_{Training}$	3 dB				50 dB				50+3 dB			
SNR_{Test}	3 dB	50 dB	7 dB*	Mean	3 dB	50 dB	7 dB*	Mean	3 dB	50 dB	7 dB*	Mean
Zeit	77.81%	79.02%	86.04%	80,96%	49.33%	94.70%	48.66%	64,23%	77.54%	92.47%	84.38%	84,80%
Freq	82.05%	89.28%	82.71%	84,68%	70.52%	94.37%	55.06%	73,31%	81.75%	91.22%	74.90%	82,62%
Ceps	88.98%	94.72%	92.96%	**92,22%**	86.83%	94.68%	92.83%	**91,45%**	88.98%	94.72%	92.96%	**92,22%**
Corr	80.45%	73.47%	84.89%	79,60%	73.07%	87.14%	77.98%	79,39%	77.90%	84.88%	82.84%	81,87%
Zeit+Freq	82.05%	89.28%	82.71%	84,68%	70.52%	94.37%	55.06%	73,31%	81.75%	91.22%	74.90%	82,62%
Zeit+Ceps	88.98%	94.72%	92.96%	**92,22%**	86.83%	94.68%	92.83%	**91,45%**	88.98%	94.72%	92.96%	**92,22%**
Zeit+Corr	80.45%	73.47%	84.89%	79,60%	49.33%	94.70%	48.66%	64,23%	80.32%	92.35%	88.22%	86,96%
Freq+Ceps	88.98%	94.72%	92.96%	**92,22%**	70.65%	94.75%	55.06%	73,49%	88.98%	94.72%	92.96%	**92,22%**
Freq+Corr	82.05%	89.28%	82.71%	84,68%	70.52%	95.60%	95.60%	87,24%	81.75%	94.42%	74.90%	83,69%

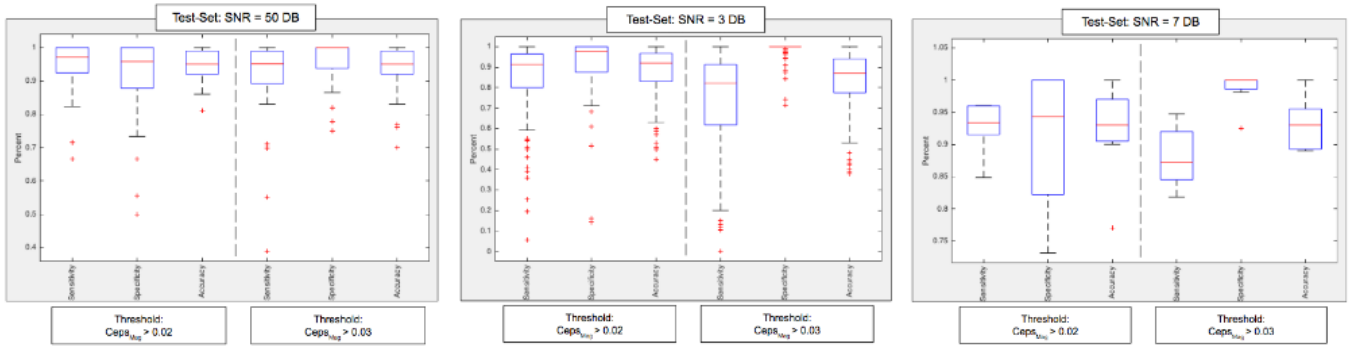

Abbildung .1: Boxplot-Auswertung über Sensitivity, Specificity und Accuracy der beiden VAD-Modelle