Georg-August-Universität
Göttingen

Wirtschaftswissenschaftliche Fakultät
Professur für Informationsmanagement
Prof. Dr. Lutz M. Kolbe

Implementierung eines Prototypen zur Hustenerkennung und Extrahierung von Merkmalen aus Audiosignalen

16-wöchige Seminararbeit im Rahmen des Kurses
„Aktuelle Themen im Informationsmanagement“ an der Universität Göttingen zum Thema
„Cough Detection and Feature Extraction“

Vorgelegt am: 04.11.2022

Von: Evgeni Uschakov

Aus: Karpinsk, Russland

Matrikelnr.: 21970213

Betreuer: Dr. Benjamin Brauer

Inhaltsverzeichnis

Abkürzungsverzeichnis ..... **iii**

Abbildungsverzeichnis ..... **iiv**

1 Einleitung **1**

2 Grundlagen **2**

2.1 Audiosignalverarbeitung 2

2.2 Machine-Learning 3

2.3 Themenbezogene Arbeiten 4

3 Methodik **5**

3.1 Design-Science 5

3.2 Problembeschreibung 6

3.3 Umsetzungsansatz 6

4 Entwicklung des Artefakts **8**

4.1 Erste Schritte 8

4.2 Machine-Learning-Ansatz 8

4.3 Trainings- und Testdaten 10

4.4 Extrahierung der Features 10

4.5 Anwendungsverlauf 11

5 Evaluation des Artefakts **12**

6 Diskussion **13**

6.1 Verbesserungsmöglichkeiten und Limitationen 13

6.2 Potentiale für zukünftige Forschung 13

7 Fazit **14**

8 Literaturverzeichnis **15**

Anhang **19**

Ehrenwörtliche Erklärung **20**

Abkürzungsverzeichnis

DSR	Design-Science-Research
EDCTP	European and Developing Countries Clinical Trials Partnership
Hz	Hertz
MFCC	Mel-Frequency-Cepstral-Coefficients
ML	Machine-Learning
PCA	Principal-Component-Analysis
RBF	Radial-basis-function
SVM	Support-Vector-Machine
ZCR	Zero-crossing-rate

Abbildungsverzeichnis

Abbildung 1. Design-Science-Research nach Hevner [Quelle: Hevner A. 2007, p. 88] 5

Abbildung 2. Funktionsweise einer One-Class-SVM [Quelle: (scikit-learn 2022)] 9

Abbildung 3. Visuelle Darstellung einer Aufnahme mit Markierungen [eigene Darstellung] 11

1 Einleitung

Tuberkulose ist nach Covid-19 die weltweit am häufigsten vorkommende Infektionskrankheit. Erkrankungserreger befallen häufig die Lunge und führen zu Symptomen wie Husten, Fieber und Atemnot. 2019 infizierten sich 10 Mio. Menschen mit der aktiven Variante, worauf etwa 1,4 Mio. Todesfälle folgten. Südafrika ist eines der am schwersten betroffenen Länder. Gründe für die dort vorzufindenden hohen Infektionszahlen sind die Verbreitung von Bakterien bei einer hohen Kontaktrate und langen Wartezeiten in öffentlichen Behandlungseinrichtungen (Karat et al. 2022), ungenaue Tests (Dr. Parker R. 2017), sowie die geringe Bereitschaft der Bevölkerung sich zu testen und das Ergebnis nach einigen Tagen abzuholen (Claassens et al. 2013). (Chakaya et al. 2021)

Einfache Smartphones können die Infektionsbekämpfung unterstützen. Mithilfe des Mikrofons könnte ein Gerät Husten aufnehmen und dem Nutzer sofort die Wahrscheinlichkeit einer Infektion, sowie das empfohlene weitere Vorgehen auf dem Bildschirm mitteilen. Aufgrund der lückenhaften Infrastruktur des Internets in Gebieten hoher Infektionszahlen müsste die Software offline funktionieren (Delaporte A. 2021).

Im Rahmen eines internationalen Forschungsprojekts mit den Universitäten Stellenbosch aus Südafrika, Makerere aus Uganda, dem Global Health and Development Institut aus Amsterdam, sowie der Georg-August-Universität Göttingen soll ein Prototyp zur Triage entwickelt werden. Die vom EDCTP geförderte Anwendung soll auf Android-Geräten funktionieren und den Namen „Cough Audio Triage for TB“ tragen. Aufgrund der freien Zugänglichkeit, einer intuitiven Benutzeroberfläche und entfallender Wartezeit wird eine Erhöhung der Testbereitschaft in der Bevölkerung erhofft. (Bigalke I. 2021)

Die Zielsetzung dieser Arbeit ist die Entwicklung einer Komponente für die Hustenerkennung und Extrahierung markanter Merkmale aus Aufnahmedaten eines Mikrofons. Dies soll eine erste Grundlage zur weiteren Entwicklung des Prototyps bilden, in dem die erkannten Hustenaufnahmen weiterverarbeitet werden.

Zur Erfüllung dieser Zielsetzung gliedert sich die Arbeit folgendermaßen. Zunächst werden die Grundlagen der Audiosignalverarbeitung mit möglichen Algorithmen und Modellen zur Geräuschklassifizierung unter Einbezug erfolgreicher themenbezogener Arbeiten vorgestellt. Später wird die Vorgehensweise bei der Implementierung des Artefakts erläutert und abschließend die Funktionalität mit Limitationen und Ausbaumöglichkeiten evaluiert.

2 Grundlagen

2.1 Audiosignalverarbeitung

In der Natur breiten sich Schallwellen als Druckschwankungen entlang eines Mediums aus (Sigal E. 2005). Ein modernes Mikrofon ist dazu fähig diese Druckunterschiede zu erfassen und zu binären Signalen zu verarbeiten. Diese können dann vom Computer gelesen werden. (Fox A. 2022)

Audiosignale bestehen aus einer Aneinanderreihung von Datenpunkten mit jeweils einer Zeit- und Amplitudenangabe. Die Amplitude ist die Stärke einer Druckwelle in Dezibel (dB), welche das menschliche Ohr als Lautstärke wahrnimmt. Diese Datenpunkte werden auch Samples genannt und stellen die Grundelemente der Audiosignalverarbeitung dar. Moderne Mikrofone nehmen mit 44.100 oder 48.000 Samples pro Sekunde auf. Das ist die Sample-Rate einer Audioquelle und wird mit der Maßeinheit Hertz (Hz) angegeben. (“Sample Rate” 2019)

Die großen Datenmengen von Samples aus Audioaufnahmen bieten die Grundlage der Erkennung von Geräuschen. Dafür werden aus diesen die Ausprägungen bestimmter Merkmale (Features) entzogen, die eine klare Unterscheidung von Audiosignalen ermöglichen. (Agashe R. 2021)

Eine sehr bekannte Eigenschaft sind sogenannte Mel-Frequency-Cepstral-Coefficients (MFCC). Sie bieten eine komprimierte Darstellung des Frequenzspektrums und werden häufig zur Spracherkennung verwendet. Dabei teilt dessen Algorithmus das Audiosignal in Bereiche von 1024 Samples, also etwa 25 Millisekunden bei einer Sample-Rate von 44.100, und rechnet diese mit einer Reihe mathematischer Formeln so um, dass das Ergebnis 39 signaldefinierende Werte pro Bereich ausgibt. (Hui J. 2019)

Weitere Merkmale sind Standardabweichung, Zero-crossing-rate (ZCR) und Wölbung (Kurtosis) (Rahman et al. 2019). Unter Standardabweichung wird im Kontext der Audiosignalverarbeitung die durchschnittliche Entfernung aller Amplituden von deren Durchschnitt verstanden (statista 2022). Wird die Berechnung auf Aufnahmedaten angewendet, errechnet sich eine Grenze. Unterhalb dieser befinden sich 68% der Samples (Yorkinov O. 2018). Die übrigen besitzen auffällig laute Amplituden und lassen sich deshalb deutlichen Geräuschen zuordnen. In der Praxis bietet sich das Verfahren für die Filterung von Hintergrundgeräuschen und die weitere Verarbeitung der hervorgehobenen Bereiche an.

ZCR ist ein Ausdruck des Lärms und ergibt sich aus der Anzahl an Vorzeichenwechsel von Amplituden pro Sekunde (Lee et al. 2018). Die Wölbung, auch genannt Kurtosis, erstellt ein Histogramm, dessen horizontale Achse Amplitudenstufen und vertikale Achse die Anzahl an Vorkommen der jeweiligen Amplitudenstufe sind. Der Algorithmus vergleicht dieses dann mit der Normalverteilung. Der Wert der Wölbung ergibt sich aus dem festgestellten Höhenunterschied. (Schaldenbrand P. 2019)

2.2 Machine-Learning

Nach der Audiosignalverarbeitung liegt eine Reihe von markanten Audioeigenschaften vor. Die Aufgabe von ML-Algorithmen besteht darin, Merkmalsausprägungen als Eingabe zu akzeptieren und intern so zu verarbeiten, dass die Ausgabe eine Einordnung der Ausgangsdaten ergibt. Um das zu ermöglichen, übergibt man dem gewählten Algorithmus große Mengen an Beispieldaten. Daraufhin findet die Optimierung interner Entscheidungsprozesse statt. Nach diesem Trainingsprozess testet man das Modell unter Nutzung von neuer und nicht-klassifizierter Daten. (Awan & Soofi, 2017)

Der interne Prozess läuft je nach Algorithmus unterschiedlich ab. Die Effizienz ist dabei von der Anzahl erfasster Merkmalsausprägungen abhängig. Nach deren Extrahierung lassen sich die Daten zu dem Zweck strukturieren und nichtssagende Ausreißer entfernen. Das Ergebnis ist eine verringerte Anzahl an Eingabeparametern, die einen Großteil der Merkmalsausprägungen beschreiben. Hier kann die Hauptkomponentenanalyse (PCA) verwendet werden. (Chen et al. 2021)

Nach dieser Reduzierung der Dimensionalität wird ein Werkzeug zur Klassifizierung benötigt. Je nach vorgegebenen Beispieldaten nutzt man überwachtes (supervised) oder unüberwachtes (unsupervised) Lernen. Bei der überwachten Variante werden als Beispiel zwei bestimmte Geräusche voneinander unterschieden und in zwei Gruppen von Beispieldaten eingeteilt. Mit dieser Unterteilung lernt das Modell Zusammenhänge der Eingabe zu deren Klassifizierung. Unüberwachte Algorithmen erhalten bei gleichem Beispiel nur Aufnahmen von einem der zu erkennenden Geräusche. Nach dem Lernprozess werden alle anderen als Ausreißer erkannt und somit unterschieden. Zu diesem Zweck kann eine SVM genutzt werden (Chen et al. 2021). (Ghahramani Z. 2003)

2.3 Themenbezogene Arbeiten

Zum Problem der Hustenerkennung existieren vielfältige Ansätze. Barry et al. behandeln in ihrer 2006 veröffentlichten Publikation das Zählen von Husten in Audiodateien. Dafür wurden zuerst Aufnahmen von Husten und häufig vorkommenden Störgeräuschen akquiriert. Aus diesen erkennt eine Software laute Stellen zur weiteren Verarbeitung. Im Anschluss werden Merkmalsausprägungen extrahiert und an die Parameterreduzierung mit PCA weitergeleitet. Ein neuronales Netzwerk lädt die gekürzten Daten als überwachter ML-Algorithmus und optimiert die Entscheidungsfindung anhand der gegebenen Klassifizierung, ob ein Husten enthalten ist oder nicht. Das Ergebnis ist ein robustes Vorgehen, jedoch mit Problemen bei der Unterscheidung von Husten mit lauten Geräuschen. (Barry et al. 2006)

Chen et al. nutzen in deren Publikation aus 2021 MFCC als Audiomerkmal und PCA zur Parameterreduzierung. Trainingsdaten bestanden hier aus Aufnahmen mit und ohne Husten. Eine Variante der supervised SVM wird zur darauffolgenden Klassifizierung der Audioaufnahmen verwendet. Die Umsetzung zeigt eine hohe Genauigkeit der Hustenerkennung von 94,9%. (Chen et al. 2021)

Eine mögliche Anwendung der Hustenerkennung auf Smartphones mit deutlich mehr Features wird in der Publikation aus 2019 von Rahman et al. beschrieben. Hier sollen Patienten mit Lungenerkrankungen mithilfe von effizienter Hustenerkennung automatisch erfasst werden. Dabei werden aus Amplituden Merkmale wie MFCC, Standardabweichung, Kurtosis, Zero-crossing-rate, Schalldruckpegel, spektraler Schwerpunkt und viele mehr in Echtzeit extrahiert. Die Features werden in Java berechnet und über das Internet an einen Server gesendet. Die Klassifizierung findet daraufhin auf einem Cloud-Server unter Benutzung des Random-Forest-Algorithmus statt. Dieser ordnet den Audioausschnitt zu Husten, Sprache oder Stille ein. Die Genauigkeit beträgt bei dieser Vorgehensweise 99,8% und ist damit das Beste der drei vorgestellten Modelle zur Erkennung von Husten. (Rahman et al. 2019)

3 Methodik

3.1 Design-Science

Das Design-Science-Research-Framework aus Abbildung 1 bietet eine grundlegende Vorgehensweise zur Entwicklung eines Artefakts als Lösung für ein aktuelles Praxisproblem. Dabei wird im sogenannten Relevance-Cycle die betroffene organisatorische und technische Umgebung analysiert und im Rigor-Cycle nach passenden Erkenntnissen aus der Wissenschaft gesucht. Im Anschluss dient der Design-Cycle der Implementierung mit Evaluation. Die Cycles werden während der Ausarbeitung eines Artefakts oft angewendet und stellen die iterative Vorgehensweise vom Design-Science dar. (Benner-Wickner et al. 2020; Hevner A. 2007)

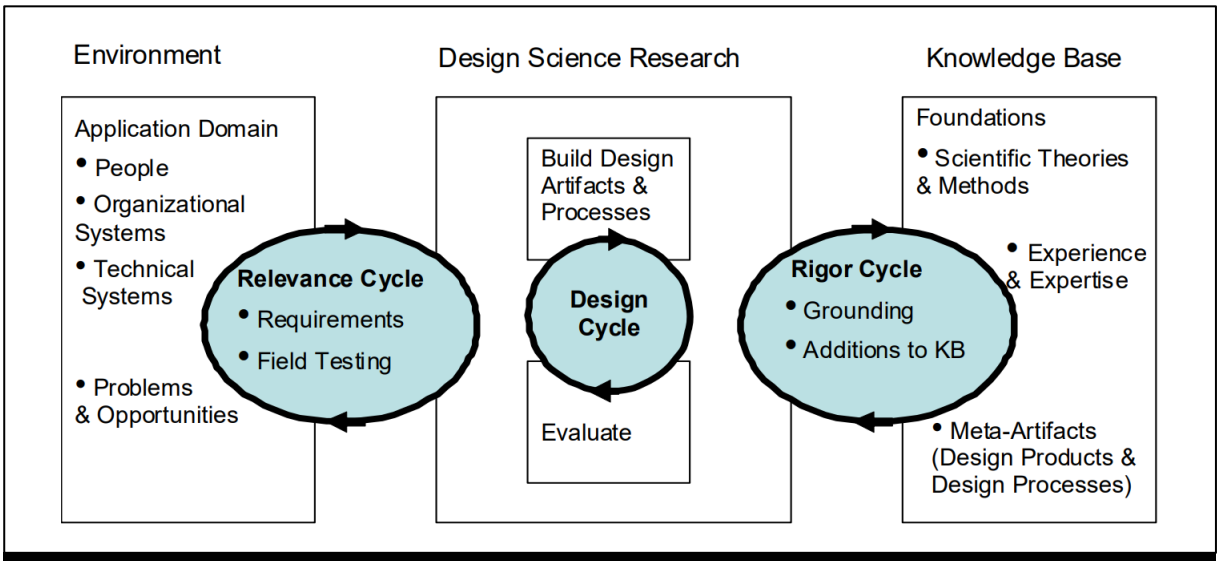

Abbildung 1. Design-Science-Research nach Hevner [Quelle: Hevner A. 2007, p. 88]

Zu Beginn sollte das Praxisproblem im Relevance-Cycle beschrieben werden. Dazu gehören die organisatorischen und technischen Umgebungen mit der Nennung von Nutzergruppen, Stakeholdern, technischen Anforderungen und weiteren Bedingungen an das Artefakt. Im Anschluss sollen die Akzeptanzkriterien potenzieller Lösungen aufgezeigt werden anhand derer das Ergebnis der Arbeit bewertet wird.

Im Rigor-Cycle wird der aktuelle Zustand der mit dem Problem zusammenhängenden Technologie dargestellt. Dafür sollen vorhandene Lösungen ähnlicher Probleme als State of the Art erwähnt werden. Hier muss deutlich gemacht werden, dass das neue Problem eine klare zu bewältigende Unterscheidung enthält.

Der Design-Cycle nutzt die Anforderungen aus dem Relevance-Cycle und die wissenschaftlichen Methoden aus dem Rigor-Cycle zur Entwicklung des Artefakts. Anschließend wird jede Iteration des Design-Cycles mit einer Evaluierung abgeschlossen. (Hevner A. 2007)

3.2 Problembeschreibung

An der Umsetzung des internationalen Forschungsprojekts beteiligen sich die Universitäten aus Stellenbosch, Makerere, Amsterdam und Göttingen. Das Ziel ist die Erhöhung der Testbereitschaft auf Tuberkulose in Risikogebieten. (Bigalke I. 2021)

Im Rahmen dieser Seminararbeit ist die Implementierung einer Lösung zur Hustenerkennung und anschließender Ausgabe von Merkmalsausprägungen gefordert. Dafür soll das Artefakt mit dem Mikrofon aufnehmen und Husten erkennen können. Diese können als Amplitudendarstellung zum Auswählen angezeigt werden. Der Nutzer entscheidet anschließend aus welchem der aufgenommenen und erkannten Husten die Features extrahiert werden sollen.

Das zu entwickelnde Artefakt soll in Java programmiert werden und aufgrund der schwachen Internetabdeckung der späteren Einsatzgebiete ohne Internetverbindung funktionieren. Deshalb werden effiziente und speichersparende Algorithmen benötigt. Zudem sollte die Implementierung ohne Mehraufwand auf das Betriebssystem Android übertragbar sein, da die Software des Forschungsprojekts auf Smartphones eingesetzt wird.

Die Qualität eines Artefakts als Lösung für das Problem kann durch die Genauigkeit der Hustenerkennung bewertet werden. Diese lässt sich aufteilen in die Rate der Husten die nicht erkannt und die Rate der Störgeräusche die als Husten klassifiziert werden. Aufgrund der Entscheidung des Nutzers, von welchem Husten die Features extrahiert werden sollen, sind nicht erkannte Husten ein größeres Problem als nicht erkannte Störgeräusche.

3.3 Umsetzungsansatz

Zur Erstellung des Artefakts wird eine etwas abweichende Variante des DSR-Frameworks genutzt. Vor und während der Implementierung des Artefakts werden vor den meisten Entscheidungen Recherchen durchgeführt. Dabei stehen die aus den Relevance-Cycles analysierten Informationen zu den organisatorischen und technischen Umgebungen, sowie den Evaluationsmöglichkeiten und der Problembeschreibung stets im Fokus. Die Design-Cycles

enthalten nicht immer eine anschließende Evaluation, da viele entwickelte Teilkomponenten alleinstehend nicht hinreichend getestet werden können. Erst nach dem Zusammenfügen mehrerer Iterationsergebnisse werden ausführliche Evaluierungen mit weiteren darauffolgenden Design-Cycle-Iterationen zur Korrektur durchgeführt. Die Ergebnisse werden dabei mit den unter dem Kapitel „Problembeschreibung“ erwähnten Möglichkeiten zur Evaluierung bewertet.

4 Entwicklung des Artefakts

4.1 Erste Schritte

Nach dem Aufsetzen der integrierten Entwicklungsumgebung Eclipse und Java, wird zuerst die Sprachaufzeichnung implementiert. Das Ergebnis ist die automatische Erstellung von Audiodateien aus Mikrofonaufnahmen. Zur Anzeige der Amplitude wird ein Fenster mit Koordinatensystem entwickelt. Diese visuelle Anzeige bietet eine hilfreiche Darstellung der Audiodateien für die späteren Schritte an.

In der nächsten Iteration sollen laute Momente aus der Aufnahme erkannt werden. Zur Lösung des Problems wird eine Schwelle (Threshold) der Amplituden eingeführt, ab der Geräusche als laut klassifiziert werden. Da jede Aufnahme unterschiedlich laut ist lässt sich dafür kein konstanter Wert finden. Eine Möglichkeit der Unterscheidung ist die Implementierung der Standardabweichung als dynamische Schwelle. Diese Berechnung ergibt eine Grenze, die über genau 68% der Samples liegt. Aufgrund der hohen Fluktuation von Amplituden, werden alle Samples in Blöcke der Länge 128 unterteilt, um in diesen jeweils den absoluten Mittelwert zu berechnen. Das Ergebnis ist eine abgeflachte Kurve der Amplituden mit einer Grenzlinie. Aus deren Schnittpunkten lassen sich anschließend laute und leise Zeitfenster ablesen.

Nach der Extrahierung lauter Momente muss eine Vorgehensweise zur Erkennung von Husten implementiert werden. Die ersten Tests mit einfachen Abfragen des Amplitudenverlaufs zeigen sich erfolglos. Das Geräusch eines Hustens hat häufig zwei Höhepunkte in der Lautstärke. Zu Beginn gibt es einen steilen Anstieg der Amplituden, gefolgt von einem kurzen leisen Moment und anschließendem lauten Teil der Stimme. Positionen mit hoher Lautstärke sollen ermittelt und mit denen anderer Husten verglichen werden. Das Problem bei dieser Vorgehensweise ist, dass Geräusche mit ähnlicher Verteilung der Höhepunkte, zum Beispiel Klatschen oder Niesen, als Husten erkannt werden. (Serrurier et al. 2022)

4.2 Machine-Learning-Ansatz

Nach ausführlicher Recherche über automatische Sprach- und Hustenerkennung und der bisher gewonnenen Erfahrung wird die Notwendigkeit eines Machine-Learning-Ansatzes deutlich (Kolossa 2017). Aus Audioaufnahmen lassen sich viele Eigenschaften der Amplituden entnehmen. Es wäre ein großer Arbeitsaufwand diese Ausprägungen manuell zu untersuchen, um daraus festzustellen, welche Merkmale Husten definieren. Dabei ist zu bedenken, dass Husten je nach Person und Krankheit unterschiedlich klingt.

Aufgrund ausreichender Genauigkeit und hoher Effizienz der Vorgehensweise der oben erwähnten Publikation von Chen et. al. wird im nächsten Schritt ein Prototyp einer SVM mit Hilfe der Library „LIBSVM“ implementiert. Eine SVM besitzt die Aufgabe aus Beispieldaten zu lernen und anschließend neue Eingaben zu klassifizieren. Bei der Hustenerkennung reicht es aus Husten von allen anderen Geräuschen zu unterscheiden, weshalb hier die „One-Class“ Variante der SVM implementiert wurde. Dieses Verfahren bietet die Erkennung von Modellfremden Eingaben an. Nachdem die SVM auf Husten trainiert wurde, sollte es alle anderen Geräusche als Ausreißer erkennen können.

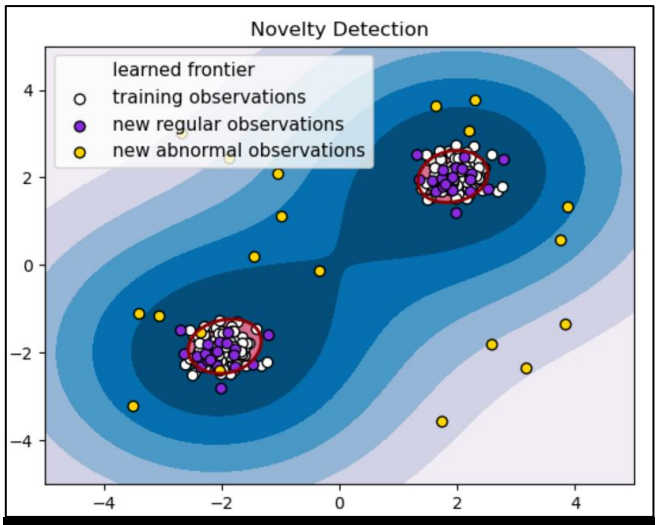

Abbildung 2. Funktionsweise einer One-Class-SVM [Quelle: scikit-learn 2022]

Abbildung 2 zeigt die Funktionsweise einer One-Class-SVM. Vereinfacht lässt sich davon ausgehen, dass die Achsen die Merkmalsausprägungen sind. Die weißen Punkte stellen die Trainingsdaten dar, anhand derer eine passende Form (blau) berechnet wird. Die anderen Punkte sind dem Algorithmus unbekannte Testdaten. Je nach Entfernung der Punkte von den Häufungen werden diese als Ausreißer (gelb) oder als regulär (violett) erkannt. Zur Berechnung der roten Trennlinien, die diese Daten unterscheiden, wird ein sogenannter Radial-basis-function-kernel (RBF) verwendet. Dieser fügt allen Punkten eine weitere Dimension hinzu und bestimmt in dem neuen Raum die Trennung. Dieser Prozess benötigt trotz effizienter Implementierung eine hohe Leistung des Endgeräts, da die Anzahl der Dimensionen beliebig groß werden kann. Aus diesem Grund soll die Menge der Eingabedaten minimiert werden.(Bounsiar and Madden 2014)

Eine Möglichkeit dieser Minimierung von Features mit geringem Informationsverlust ist PCA. Bei diesem Algorithmus werden neue Achsen berechnet, die die meiste Varianz der Trainingsdaten erklären. Die Anzahl, auf die die Dimensionen reduziert werden, ist dabei beliebig wählbar. Hier ist jedoch zu beachten, dass ein Dimensionsverlust ebenfalls ein Informationsverlust darstellt und deswegen die Genauigkeit des Modells beeinträchtigen kann.

Bei der Implementierung dieser Algorithmen finden die Bibliotheken „LIBSVM“ für SVM, und „CoMIRVA“ für PCA als Grundlage Verwendung. Damit die Modelle nicht bei jedem

Start der Software neu trainiert werden müssen, wird die Speicherung der Modelle von PCA und SVM jeweils als Datei implementiert.

4.3 Trainings- und Testdaten

Im nächsten Schritt werden 25.000 Hustenaufnahmen aus der Publikation von Orlandic et. al. aus 2021 heruntergeladen. Diese Audiodateien stehen für Machine-Learning-Anwendungen öffentlich zur Verfügung und wurden von Personen mit Covid-19-Infektion aufgenommen und hochgeladen (Orlandic et al. 2021). Die Erkrankung spielt im Anwendungsfall der Hustenerkennung nur eine geringe Rolle, da die Formen der Husten ähnlich sind und die Einstellungsparameter der späteren ML-Modelle eine weniger strenge Klassifizierung erlauben können.

Die heruntergeladenen Audiodateien müssen im nächsten Schritt nach Nutzbarkeit und Dateityp gefiltert, in eine Trainings- und Testgruppe unterteilt und auf eine Sample-Rate von 44.100 Hz konvertiert werden. Der letzte Schritt stellt sicher, dass die Merkmalsausprägungen aller Aufnahmen vergleichbar sind. Beim Trainieren der One-Class-SVM ist es besonders wichtig auf die Sauberkeit der Eingaben zu achten, da störende Geräusche im späteren Einsatz als Ausreißer erkannt werden müssen. Die Filterung nach der Nutzbarkeit wird manuell durchgeführt, indem 80 Dateien aus der Trainingsgruppe und 18 aus der Testgruppe auf die Beinhaltung von Husten geprüft werden. Es stellt sich heraus, dass viele Aufnahmen nur Stille, Gespräche oder andere Störgeräusche enthalten.

4.4 Extrahierung von Features

In der nächsten Iteration werden Entscheidungen bei der Auswahl von Features getroffen. Aufgrund der vielversprechenden Ergebnisse der unter „Themenbezogene Arbeiten“ vorgestellten Publikation von Rahman et. al., werden einige Merkmale übernommen. Diese sind Wölbung, Zero-crossing-rate und MFCC. Dazu wird das Verhältnis vom Mittel- zum Maximalwert, wodurch eine Darstellung der Verteilung von Amplituden möglich ist, eingeführt. Die Berechnungen der Wölbung und den MFCC übernehmen Bibliotheken. Hier wird „CoMIRVA“ für die Erzeugung der Koeffizienten und „Apache Commons Math“ für die Wölbung genutzt. Die anderen Features werden mit selbsterstellten Funktionen extrahiert. Mit diesen ausgelagerten Methoden lassen sich Merkmalsausprägungen ohne Umwege aus Aufnahmen entziehen.

Im darauffolgenden Schritt soll die Vorgehensweise der Extrahierung von Features aus längeren Aufnahmen feststehen. Dabei ist zu beachten, dass die Werte der MFCC in Zeitfenstern fester Länge von etwa 25 Millisekunden berechnet werden. Ein Ansatz ist die Erfassung der Merkmalsausprägungen aus Zeitfenstern der lautesten Amplitude. Nach der Evaluation der Ergebnisse fällt jedoch auf, dass Aufnahmen von Husten mehrere laute Amplituden an unterschiedlichen Stellen enthalten können. Die lauteste Amplitude kann sich am Anfang oder am Ende eines Hustens befinden. Die nächste implementierte Vorgehensweise ist ein sich bewegendes Zeitfenster. Die Aufnahme wird in überlappende Fenster eingeteilt, von denen jeweils die Features berechnet und an die Modelle zur Auswertung weitergegeben werden. Zur Klassifizierung reicht es aus, wenn eines der Zeitfenster als Teil eines Hustens erkannt wird.

4.5 Anwendungsverlauf

Mit dieser Vorgehensweise werden die Features, aus den vorbereiteten Hustenaufnahmen der Trainingsgruppe, normalisiert und dem Modell zum Trainieren überreicht. Das Testen verläuft mit den übrigen oder eigenen Aufnahmen.

Nach der Aufnahme und der Hustenerkennung öffnet das Programm ein Fenster zur Darstellung der Amplitude mit Markierungen von lauten Momenten mit grauen, und erkannten Husten mit blauen Bereichen. Zwei rote Linien zeigen die Höhe der Standardabweichung der gesamten Aufnahme. Die Abbildung 3 präsentiert die visuelle Darstellung anhand eines Beispiels. Anschließend wird dem Nutzer die Ausgabe der Features eines der Husten angeboten. Dafür soll die Nummer des gewünschten Bereichs in der Konsole eingegeben werden.

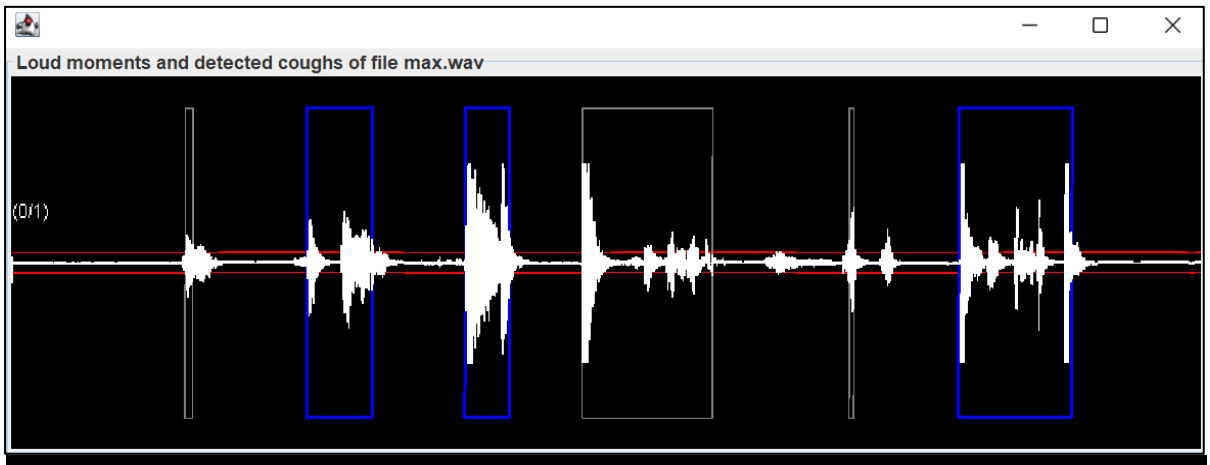

Abbildung 3. Visuelle Darstellung einer Aufnahme mit Markierungen [eigene Darstellung]

5 Evaluation des Artefakts

Die implementierte Testfunktion gibt bei Ausführung eine Wiedererkennungsrate der Husten von 75% aus. Dieses Ergebnis wird berechnet, indem die Anzahl der erkannten lauten Momente aus Hustenaufnahmen durch die Anzahl der erkannten Husten geteilt wird. Dabei werden alle Störgeräusche wie Räuspern, Stimme und nahes Atmen am Mikrofon als laute Momente klassifiziert, die das Ergebnis verfälschen. Manuelles Durchzählen von deutlichen Husten bei gleichen Testdaten ergibt eine Erkennungsrate von 86%. Selbst erstellte Aufnahmen bei Nutzung eines Laptopmikrofons in ruhiger Umgebung zeigen eine deutlich höhere Genauigkeit von 95%. Sehr laute Husten, deren Audiowellen das Maximum der Mikrofonlautstärke übertreffen, klingen verzerrt und werden dadurch häufig falsch klassifiziert.

Bei dem Vorhaben der Triage mit Android-Geräten können mit dieser Grundlage Husten in ruhiger Umgebung verlässlich erkannt und zur weiteren Auswertung der Features ausgewählt werden.

Sollten laute Momente häufiger als Husten erkannt werden, lässt sich die Strenge der Klassifizierungen in den SVM-Parametern verringern. Solche Änderungen können jedoch dazu führen, dass das Programm andere Störgeräusche fälschlicherweise als Husten markiert. Letzteres stellt kein großes Problem dar, da der Nutzer am Ende die Wahl hat, von welchem markierten Bereich die Features extrahiert werden sollen.

6 Diskussion

Die unter „Themenbezogene Arbeiten“ vorgestellten Publikationen zeigen ähnliche Genauigkeiten auf. Laute und unklare Aufnahmen werden dabei ebenso häufig falsch klassifiziert. Die Unterschiede sind die Wahl der Features, die Unabhängigkeit von einer Internetverbindung und die Nutzung von unsupervised One-Class-SVM, wodurch die Klassifizierung durch Hustenaufnahmen ohne Gebrauch von Störgeräuschen erlernt wird.

Mögliche nächsten Schritte können die Erweiterung der Trainingsdaten und die Aufspaltung des Programms in die Komponenten zum Trainieren und zum Anwenden der Hustenerkennung im Forschungsprojekt sein. Die Idee ist, dass die Modelle aufgrund hoher Komplexität auf einem Computer trainiert werden. Die daraus erstellten Modelldateien können anschließend mit geringem Aufwand auf Smartphones gespeichert und zur Hustenerkennung eingelesen werden.

6.1 Verbesserungsmöglichkeiten und Limitationen

Die manuelle Filterung von Audioaufnahmen zum Trainieren der Modelle ist ein Zeitaufwändiger Prozess, weshalb bei dieser Arbeit nur etwa 100 Dateien geprüft und verwendet wurden. Mehr Trainingsdaten würden zu einer noch höheren Genauigkeit der Hustenerkennung führen da die Modelle auf weitere Hustenarten vorbereitet wären. Hier ist besonders auf die Sauberkeit der genutzten Aufnahmen zu achten. Wenn Störgeräusche enthalten sind, werden diese vom Modell aufgefasst und bei der späteren Anwendung als Husten markiert.

Erwägenswert ist die Einführung weiterer Features. Aufgrund der Implementierung von PCA wird die Effizienz nur durch den Aufwand der Extrahierung von Merkmalen beeinflusst. Deshalb sollte die Einführung weiterer Features kein Problem für die Leistung darstellen.

Eine weitere Möglichkeit für eine höhere Genauigkeit ist die zusätzliche Anpassung von Parametern des SVM-Modells. Dadurch können mehr Geräusche mit Ähnlichkeit zu Husten zur Auswahl markiert, oder striktere Bedingungen an Hustenaufnahmen gestellt werden.

Ein potenzielles Problem ist die Normalisierung der Features. Das Programm bestimmt minimale und maximale Werte jeder Spalte aus den Trainingsdaten und wendet diese auf die Features der Testdaten an. Sollten die neuen Eingaben jedoch außerhalb der Grenzen liegen kann eine falsche Klassifikation erfolgen.

Eine weitere Limitation ist die fehlende Filterung von Geräuschen. Husten während der Aufnahme mehrere Personen gleichzeitig, oder treten nebenbei Störgeräusche im Hintergrund auf, besitzen die aus dem Husten extrahierten Features keine Aussagekraft.

6.2 Potentiale für zukünftige Forschung

Das Artefakt ist eine robuste Grundlage für die ML-Verfahren PCA und SVM in Java. Mit den implementierten Funktionen der Speicherung und dem Laden von den zum Modell zugehörigen Dateien ist eine effiziente Weiterentwicklung und die spätere Ausführung des Programms auf Android-Geräten ohne erneutem Trainingsaufwand möglich. Das bietet im Bezug zum Forschungsprojekt eine Grundlage zur Bestimmung von Infektionen aus Hustenaufnahmen. Zudem ist das Artefakt nicht nur auf Husten begrenzt. Es erlaubt die Erkennung von beliebigen Geräuschen, sofern genug Aufnahmen zum Trainieren der Modelle vorhanden sind.

7 Fazit

In dieser Arbeit wurde im Rahmen eines Forschungsprojekts zur Triage für Tuberkulose auf Smartphones eine effiziente Hustenerkennung mit anschließender Ausgabe der Features mit Java implementiert. Die Features MFCC, ZCR, Wölbung und der Anteil des Mittelwerts an dem Maximalwert der Amplituden werden mit geringem Aufwand extrahiert und an die Algorithmen PCA und SVM zur Klassifizierung weitergegeben. Diese arbeiten effizient, benötigen keine Verbindung zum Internet und sind in der hier implementierten Form auch auf Android umsetzbar. Mit einer hohen Rate richtig klassifizierter Husten von 95% sind alle Anforderungen des Forschungsprojekts an diese Komponente erfüllt. Für die weitere Entwicklung ist diese Grundlage demnach gut geeignet.

8 Literaturverzeichnis

Agashe R. 2021. “Building Intelligent Audio Systems.” (https://www.einfochips.com/blog/building-intelligent-audio-systems-audio-feature-extraction-using-machine-learning/, accessed October 21, 2022).

Barry, S. J., Dane, A. D., Morice, A. H., and Walmsley, A. D. 2006. “The Automatic Recognition and Counting of Cough,” *Cough* (2:1), Springer Science and Business Media LLC. (https://doi.org/10.1186/1745-9974-2-8).

Benner-Wickner, M. ;, Kneuper, R. ;, and Schlömer, I. 2020. “Leitfaden Für Die Nutzung von Design Science Research in Abschlussarbeiten.” (http://hdl.handle.net/10419/229136).

Bigalke I. 2021. “News - Huge Grant Enables SU Scientists to Develop...” (http://www.sun.ac.za/english/Lists/news/DispForm.aspx?ID=8557, accessed October 19, 2022).

Bounsiar, A., and Madden, M. G. 2014. “Kernels for One-Class Support Vector Machines,” *ICISA 2014 - 2014 5th International Conference on Information Science and Applications*, IEEE Computer Society. (https://doi.org/10.1109/ICISA.2014.6847419).

Chakaya, J., Khan, M., Ntoumi, F., Aklillu, E., Fatima, R., Mwaba, P., Kapata, N., Mfinanga, S., Hasnain, S. E., Katoto, P. D. M. C., Bulabula, A. N. H., Sam-Agudu, N. A., Nachega, J. B., Tiberi, S., McHugh, T. D., Abubakar, I., and Zumla, A. 2021. “Global Tuberculosis Report 2020 – Reflections on the Global TB Burden, Treatment and Prevention Efforts,” *International Journal of Infectious Diseases* (113), Elsevier B.V., p. 8. (https://doi.org/10.1016/j.ijid.2021.02.107).

Chen, X., Hu, M., and Zhai, G. 2021. *Cough Detection Using Selected Informative Features from Audio Signals*. (http://arxiv.org/abs/2108.03538).

Claassens, M. M., du Toit, E., Dunbar, R., Lombard, C., Enarson, D. A., Beyers, N., and Borgdorff, M. W. 2013. “Tuberculosis Patients in Primary Care Do Not Start Treatment. What Role Do Health System Delays Play?,” *International Journal of Tuberculosis and Lung Disease* (17:5), pp. 603–607. (https://doi.org/10.5588/ijtld.12.0505).

Delaporte A. 2021. “GSMA | The State of Mobile Internet Connectivity in Sub-Saharan Africa: Why Addressing the Barriers to Mobile Internet Use Matters Now More than Ever | Mobile for Development.” (https://www.gsma.com/mobilefordevelopment/blog/the-state-of-mobile-internet-connectivity-in-sub-saharan-africa/, accessed October 19, 2022).

Dr. Parker R. 2017. “Implications of Tuberculosis Sputum CultureTest Sensitivity on Accuracy of OtherDiagnostic Modalities,” *American Journal of Respiratory and Critical Care Medicine* (371:2), pp. 1005–1015. (https://doi.org/10.1164/rccm.201803).

Fox A. 2022. “How Do Microphones Work? (The Ultimate Illustrated Guide) – My New Microphone.” (https://mynewmicrophone.com/how-do-microphones-work-a-helpful-illustrated-guide/, accessed November 1, 2022).

Ghahramani Z. 2003. *Advanced Lectures on Machine Learning*, (Vol. 3176), Lecture Notes in Computer Science, (O. Bousquet, U. von Luxburg, and G. Rätsch, eds.), Berlin, Heidelberg: Springer Berlin Heidelberg. (https://doi.org/10.1007/B100712).

Hevner A. 2007. “A Three Cycle View of Design Science Research.” (https://www.researchgate.net/publication/254804390).

Hui J. 2019. “Speech Recognition MFCC & PLP.” (https://jonathan-hui.medium.com/speech-recognition-feature-extraction-mfcc-plp-5455f5a69dd9, accessed October 21, 2022).

Karat, A. S., McCreesh, N., Baisley, K., Govender, I., Kallon, I. I., Kielmann, K., MacGregor, H., Vassall, A., Yates, T. A., and Grant, A. D. 2022. “Estimating Waiting Times, Patient Flow, and Waiting Room Occupancy Density as Part of Tuberculosis Infection Prevention and Control Research in South African Primary Health Care Clinics,” *PLOS Global Public Health* (2:7), Public Library of Science (PLoS), p. e0000684. (https://doi.org/10.1371/journal.pgph.0000684).

Kolossa, D. 2017. “Grundlagen Der Automatischen Spracherkennung.” (https://www.researchgate.net/publication/270281453).

Lee, S.-C., Wang, J.-F., and Chen, M.-H. 2018. *Threshold-Based Noise Detection and Reduction for Automatic Speech Recognition System in Human-Robot Interactions*, pp. 4–4. (https://doi.org/10.3390/s18072068).

Orlandic, L., Teijeiro, T., and Atienza, D. 2021. “The COUGHVID Crowdsourcing Dataset, a Corpus for the Study of Large-Scale Cough Analysis Algorithms,” *Scientific Data* (8:1), Nature Research. (https://doi.org/10.1038/s41597-021-00937-4).

Rahman, J., Nemati, E., Rahman, M., Vatanparvar, K., Nathan, V., Digital, K., and Lab, H. 2019. *Efficient Online Cough Detection with a Minimal Feature Set Using Smartphones for Automated Assessment of Pulmonary Patients*.

“Sample Rate.” 2019. (https://magroove.com/blog/en-us/sample-rate/, accessed October 21, 2022).

Schaldenbrand P. 2019. “Kurtosis.” (https://community.sw.siemens.com/s/article/kurtosis, accessed November 2, 2022).

scikit-learn. 2022. “Scikit Learn: One-Class-SVM.” (https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html, accessed November 3, 2022).

Serrurier, A., Neuschaefer-Rube, C., and Röhrig, R. 2022. “Past and Trends in Cough Sound Acquisition, Automatic Detection and Automatic Classification: A Comparative Review,” *Sensors (Basel, Switzerland)* (22:8), Sensors (Basel). (https://doi.org/10.3390/S22082896).

Sigal E. 2005. “Akustik: Schall.” (https://www.mu-sig.de/Theorie/Akustik/Akustik04.htm, accessed October 21, 2022).

Soofi, A. A., and Awan, A. 2017. “Classification Techniques in Machine Learning: Applications and Issues,” *Journal of Basic & Applied Sciences* (13), pp. 459–465.

statista. 2022. “Standardabweichung | Statista.” (https://de.statista.com/statistik/lexikon/definition/126/standardabweichung/, accessed October 20, 2022).

Yorkinov O. 2018. “Understanding Standard Deviation and 68-95-99.7 Rule with R.” (https://www.datatechnotes.com/2018/01/undersanding-standard-deviation-and-68.html, accessed October 22, 2022).

Anhang

In der folgenden Tabelle werden genutzte Programmiersprachen, Tools und Bibliotheken mit Versionsnummern und deren Verwendung zur vollständigen Dokumentation angegeben.

Name	Version	Verwendung
Java	JavaSE-1.8	Programmiersprache
Eclipse	4.25.0	Entwicklungsumgebung
CoMIRVA	0.36	MFCC, PCA
Apache Commons Math	3.6.1	Wölbung (Kurtosis)
JAMA	1.0.3	Matrixberechnungen
LIBSVM	3.3	One-Class-SVM

Ehrenwörtliche Erklärung

Ich versichere, dass ich die Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen oder anderen Quellen entnommen sind, sind als solche kenntlich gemacht. Die schriftliche und elektronische Form der Arbeit stimmen überein. Ich stimme der Überprüfung der Arbeit durch eine Plagiatssoftware zu.

Göttingen, 4.11.2022
Ort, Datum

Uschanov
Unterschrift