

Umgebungserkennung aus Rauschen in Audiodaten

Dokumentation zum Praktikum „Implementierung in Forensik und Mediensicherheit“

A. Stefanie Blümer, Tim Pollandt

20. April 2021



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Prof. Martin Steinebach
Dr.-Ing. Sascha Zmudzinski
Fraunhofer SIT

Inhaltsverzeichnis

1 Übersicht	3
1.1 Einführung	3
1.2 Ansatz	3
2 Implementierung	4
2.1 Allgemein	4
2.2 Rauschextraktion	4
2.3 Umgebungsdetektion	5
2.4 Empfehlungen	5
3 Ergebnisse	6
3.1 Umgebungen	6
3.2 Plots	7
3.3 Erkennung	10
3.4 Resümee	11
4 Mögliche Erweiterungen	12
5 Bedienung	13
5.1 Einrichtung	13
5.2 Ausführung	13
5.3 Ordnerstruktur	13
5.4 Parametrisierung	13
5.5 Interpretation	14

1 Übersicht

Hier werden wir kurz in die Thematik einführen und einen Überblick über die Teile des Projekts geben.

1.1 Einführung

In der Forensik ist es wichtig Manipulationen in Audiodaten erkennen zu können, um etwa Beweismittel auf ihre Plausibilität prüfen zu können. Darüber hinaus ist es sehr hilfreich prüfen zu können, ob Aufnahmen an einer bestimmten Stelle aufgenommen sein könnten. Dadurch ist es möglich qualitativ unzureichend gefälschte Aufnahmen zu erkennen.

Neben vielen weiteren Ansätzen wie etwa Hall oder ENF¹-Muster kann hierzu das spezifische Rauschen eines Ortes genutzt werden. Dieses unterscheidet sich je nach Umgebung und enthält hohe Anteile unterschiedlicher Frequenzen, die verglichen werden können.

1.2 Ansatz

Unser Ansatz basiert auf der Arbeit von Ikram und Malik [IM10]. Unser Ziel ist es durch Betrachtung des Rauschens einer Audiodatei zu beurteilen, ob diese an bestimmten Orten aufgenommen wurde. Dies kann einerseits zur Plausibilitätsprüfung behaupteter Aufnahmeorte genutzt werden, ermöglicht andererseits mit Erweiterungen auch verschiedene Ansätze zur Erkennung, ob in einer Audiodatei Abschnitte aus anderen, an einem anderen Ort aufgenommenen, Aufnahmen eingefügt wurden. Dadurch wären viele Manipulationen erkennbar.

Im Folgenden arbeiten wir meist im Frequenzraum, haben zunächst also eine Schnelle Fourier-Transformation (FFT) durchgeführt.

Da die Erkennung von Orten und Manipulationen auf Aufnahmen arbeiten soll, in denen Sprache vorhanden ist, müssen wir diese Sprachelemente zunächst eliminieren bzw. reduzieren, um reines Rauschen vergleichen zu können. Dieses Rauschen kann zwar über den zeitlichen Verlauf weiterhin variieren (etwa durch vorbeifahrende Autos), der Einfluss solcher Effekte kann allerdings durch verschiedene Parameter, wie etwa die Nutzung von Median/Mittelwert in verschiedenen Verarbeitungsschritten später beeinflusst werden. Um vorzubereiten, dass Manipulationen auch dann erkannt werden, wenn ausschließlich ein Sprachsegment, also ein Audienteil ohne reine Rauschsequenzen, inseriert wurden, kommt ein Herausschneiden aller Teile mit Sprache nicht infrage und wir müssen auch an diesen Stellen das reine Rauschen extrahieren.

Dazu nutzen wir einen zweistufigen Ansatz in dem zunächst eine Standard-Rauschreduzierung durchgeführt wird. Durch Subtraktion dieses sprachreduzierten Signals erhalten wir eine erste Rauschapproximation. In einer weiteren Stufe reduzieren wir anschließend weiter sogenannte Voice Leakage, also verbliebene Einflüsse von Sprache. Mehr Details zur Rauschreduzierung folgen in Abschnitt 2.2.

Anschließend folgt eine Zuordnung von Audiodateien zu verschiedenen Umgebungen. An dieser Stelle haben wir ein eigenes Vorgehen entwickelt, das auf Abweichungen zu einem erwarteten Rauschen basiert. Dazu nehmen wir zunächst zu verschiedenen Umgebungen jeweils einige Audiodateien als Grundlage. Diese müssen jeweils ein kurzes Intervall reinen Rauschens (also Sprachfreiheit) enthalten, um optimale Ergebnisse bei der Rauschreduzierung zu erzielen. Zur Erstellung des erwarteten Rauschens der jeweiligen Umgebung wird dann allerdings der gesamte Zeitraum genutzt, da überall die Sprache wie oben beschrieben entfernt wurde. Durch Segmentierung aller in einer Umgebung aufgenommenen Dateien haben wir nun viele Segmente aus denen wir die Rauscherwartung bestimmen. Mit den Rauscherwartungen verschiedener Umgebungen vergleichen wir später die Rauschapproximationen zuzuordnender Aufnahmen und können so Plausibilitäten abschätzen, dass diese in einer bestimmten Umgebung aufgenommen wurden. Details zu diesem Prozess und den Metrikvariationen werden in Abschnitt 2.3 beschrieben. Weitere Ansätze werden in Abschnitt 4 beschrieben.

Derzeit ist die Software trotz der Schaffung vieler Grundlagen noch nicht voll für den Einsatz zur Erkennung von Manipulationen in Dateien geeignet. Nöte Schritte und Ideen dazu werden in Abschnitt 4 diskutiert.

¹Elektrische Netzfrequenz

2 Implementierung

Im Folgenden gehen wir auf die Teile der Implementierung ein und wie diese funktionieren. Details zu den einzelnen Methoden sind im Quelltext dokumentiert und werden hier deshalb ausgelassen. Die Details zur Bedienung unserer Software sind im Abschnitt 5 zu finden. Auf die Ergebnisse und Grafiken aus der Software gehen wir in Abschnitt 3 ein. Details zu den generierbaren Plots folgen in den Abschnitten 3.2 und 5.

2.1 Allgemein

Unsere Implementierung ist in Python geschrieben. Das ermöglicht die Nutzung effizienter Library-Implementierungen von Funktionen wie etwa FFT aus NumPy und einfachen In- und Export sowie Verarbeitung von Audio-Signalen und strukturierten Daten wie CSV. Eine gute Alternative wäre eine Implementierung in der ebenfalls verbreiteten Sprache Matlab mit ähnlichen Eigenschaften.

Zunächst lesen wir Audiospuren aus WAV-Dateien als Signalvektor ein. Unsere Audiodateien sind zunächst mit einem Gerät und ohne Rauschunterdrückung bzw. Lautstärkeadaption aufgenommene WAV-Dateien mit einer Samplerate von 44,1kHz und einer Tonspur. Die Software funktioniert jedoch auch mit anderen Frequenzen und kann auch Stereo-Dateien einlesen, wobei die Analysen jedoch nur auf einer der Spuren ausgeführt werden.

Unsere vollständige Implementierung sowie eine Liste der benötigten pip-Pakete und ein Beispielskript zur Nutzung der Software sind unter [BP20] zu finden.

2.2 Rauschextraktion

Wir analysieren zunächst auf kleinen Fenstern das Vorhandensein von Sprache. Dazu nutzen wir auf den Fenstern eine Implementierung von Googles WebRTCVAD [Wis]. Diese ist grundlegend nur für wenige Samplerates implementiert. Deshalb approximieren wir das Resultat durch Nutzung der logarithmisch nächsten Samplerate, was in unseren Tests zu sehr guten Ergebnissen führte. Aus der booleschen Klassifizierung kleiner Fenster nach Enthalten von Sprach- bzw. Tonanteilen detektieren wir darauf basierend das größte Intervall mit reinem Rauschen in der gegebenen Audiodatei. Dieses wird benötigt, um eine Rauschreduzierung durchführen zu können. Die rauschreduzierte Version ist dann Grundlage für die Extraktion reinen Rauschens.

Mithilfe des zuvor bestimmten größten Rauschintervalls führen wir nun eine Rauschreduzierung auf überlappenden Fenstern durch. Diese ist ähnlich zur Implementierung in Audacity. Dennoch funktionierte sie – vermutlich durch flexiblere Parametrisierung und die automatische Erkennung eines geeigneten Rauschintervalls – besser². Eine erste Rauschapproximation erhalten wir nun durch Subtraktion des rauschreduzierten Signals vom Ursprungssignal.

Um verbleibende Sprachanteile „Voice Leakage“ weiter zu reduzieren, folgen wir dem Ansatz von [KL02]. Dort wird Multiband Spectral Subtraction vorgestellt, durch die frequenzbandspezifisch Sprachanteile im Rauschen und zuvor zu stark reduziertes Rauschen ausgeglichen werden. Die Formeln lassen wir hier aus und verweisen auf das o. g. Paper sowie den Quellcode. Unter Zuhilfenahme einer Matlab-Implementierung [Zav] haben wir eine funktionell an unsere Bedürfnisse angepasste Python-Implementierung erstellt. Diese nutzt in unserer angepassten Version die zuvor erkannten Intervalle reinen Rauschens als Grundlage. Hierbei mussten wir zwar einen deutlichen Abfall der Gesamtenergie und damit einen potenziellen Präzisionsverlust verzeichnen, erhielten aber nach o. g. Kriterien auch sprachfreiere Ergebnisse.

Zu beachten ist, dass auch in der Implementierung der Multiband Spectral Subtraction ein Voice Activity Detector (VAD) zwecks Sprachklassifizierung zum Einsatz kommt. Im Gegensatz zum zuvor erwähnten, basiert dieser jedoch darauf bereits ein kurzes Intervall reinen Rauschens als Grundlage zu haben und bekommt somit bei uns auch das längste Rauschintervall aus der Klassifizierung vor der ersten Rauschreduzierung.

²Auch wenn es hier keine klare Metrik gibt, basiert diese Aussage auf einem akustischen Anhören der resultierenden Audiodateien und der Betrachtung von Zeit-Frequenz-Plots in denen die jeweiligen Energien durch Farben dargestellt werden. Jeweils kann das Verbleiben von Sprachfrequenzen subjektiv beurteilt werden.

2.3 Umgebungsdetektion

Um das Rauschen verschiedener Aufnahmen verschiedenen Umgebungen zuzuordnen bzw. Zuordnungsplausibilität zu prüfen, vergleichen wir die Energieabweichungen aller Frequenzen zwischen Umgebungsrauschen und Rauschen der Aufnahme.

Zunächst generieren wir also pro Umgebung eine erwartete Energie pro Frequenz. Dazu berechnen wir den Median aus allen Fenstern (Länge unterschiedlich wählbar) aller an diesem Ort aufgenommenen Aufnahmen.

Zur Zuordnung bzw. Plausibilitätsprüfung berechnen wir den Median der Frequenzenergien nun also auch über die Fenster einer Aufnahme. Damit können wir nun durch verschiedene Abweichungsberechnungen verschiedene Metriken erhalten. Grundlegend mitteln wir die quadrierten Abweichungen aller Frequenzen und sehen Umgebungen mit kleineren Ergebnissen als plausibler an. Implementierte Metrikänderungen sind die folgenden:

- Lineare (oder andere wie Wurzel) Abweichungsberechnung statt quadrierter Abweichungsberechnung
 - gewichtet einzelne starke Abweichungen weniger extrem
 - funktionierte in Tests schlechter
- Logarithmische y-Skalierung vor Differenzberechnung und anschließende Anwendung der Exponentialfunktion auf die Differenzen
 - betrachtet prozentuale statt absoluten Abweichungen
 - beides ähnlich gut, abhängig von anderen Parametern
- Mean oder Median für Differenzen
 - derzeit: Mean über Differenzen eines Fensters, um starke Abweichungen in ein paar Frequenzen nicht zu vernachlässigen
 - derzeit: Median über die Fenster einer Aufnahme, um einzelne Signalstörungen (bspw. herunterfallende Gegenstände) zu vernachlässigen
- Gewichtungen der Frequenzen (x-Achse)
 - gleichverteilt (Standard) → funktioniert gut
 - Bevorzugung niedriger Frequenzen, da dort nach FFT geringe Frequenzdichte pro Oktave → funktioniert gut
 - geringere Gewichtung des Sprachspektrums (implementiert durch zwei Gauß-Kurven) → funktioniert weniger gut

2.4 Empfehlungen

Aktuell akzeptiert die Implementierung ausschließlich WAV-Dateien. Diese können mit verschiedenen Frequenzen erstellt werden, bei Stereo-Dateien geschieht die Analyse derzeit ausschließlich auf der linken Audio-Spur. Aufgrund der Implementierungen genutzter Frameworks gibt es zwar feste Frequenzen, auf denen die Klassierung von Segmenten als reines Rauschen oder Sprache funktioniert, unsere Software approximiert an dieser Stelle jedoch durch die ähnlichste unterstützte Frequenz, wodurch wir auch beispielsweise auf 44,1 kHz keine signifikanten Fehlklassifizierungen feststellen konnten. Wir empfehlen grundlegend Dateien von mindestens 10 Sekunden zu nutzen, die mindestens 3 Sekunden reinen Rauschens erhalten. Sollte kein Segment aus mindestens 512 Segmenten als reines Rauschen klassifiziert werden können, nehmen wir an, dass der Anfang der Datei reines Rauschen ist was zu schlechteren Ergebnissen führen kann. Die Klassifizierungsentscheidung ist aus den Plots erkennbar (s. Abschnitt 3.2), im Fall von zu wenig erkanntem reinem Rauschen wird außerdem eine entsprechende Warnung ausgegeben. Um eine realistische Schätzung des Umgebungsrauschen zu erhalten, empfehlen wir außerdem zu jedem Ort mit etwas Abstand Aufnahmen zu machen, um zeitlich schwankende Rauschkomponenten zu mitteln. Des Weiteren empfehlen wir die Nutzung eines einzigen Aufnahmegeräts ohne Lautstärkeadaption, da aktuell kein relativer Energievergleich zwischen den Frequenzen, sondern ausschließlich ein Absolutvergleich zum Einsatz kommt.

3 Ergebnisse

Hier beschreiben und zeigen wir die Resultate unserer Experimente. Zum Erhalten der Rohdaten, Audiodateien und Bilder, siehe Abschnitt 5.

3.1 Umgebungen

Um die Software zu testen, haben wir zunächst mehrere Aufnahmen an verschiedenen Orten erstellt. Dabei haben wir darauf geachtet sowohl diverse als auch sehr ähnliche Umgebungen vertreten zu haben. Die Umgebungen sind die folgenden:

Raum 1 Ein möblierter Raum in einem Wohnhaus. Die Fenster sind geschlossen und es läuft keine menschlich wahrnehmbare Technik.

Raum 2 Der Nachbarraum von **Raum 1** mit sehr ähnlichen Eigenschaften.

Treppe Das Treppenhaus des gleichen Gebäudes. Hier sind die Fenster geöffnet und es ist eine leichte Geräuschkulisse hörbar.

Platz Ein Platz in der Nähe des Gebäudes. Es ist sehr windig und im Hintergrund sind immer wieder Menschen oder Autos zu hören.

Straße Ein Wanderweg neben einer Bundesstraße. In dichtem Abstand sind deutlich Autos verschiedener Arten hörbar.

Wald Ein Wanderweg mitten im Wald. Es ist leichtes Rascheln von Blättern und ein paar Vögel zu hören.

An jeder Umgebung haben wir mit zwei Mobiltelefonen zunächst 5 Aufnahmen mit verschiedenen Sprachsequenzen verschiedener Menschen erstellt, die jeweils auch einige Sekunden Stille beinhalten. Später haben wir mit einem der Geräte nochmal weitere Aufnahmen an diesen Orten erstellt. Da unsere Software im aktuellen Stadium kaum für verschiedene Aufnahmegeräte geeignet ist, konzentrieren wir uns in den hier beschriebenen Experimenten meist auf die Aufnahmen eines Gerätes.

Zu Raum 2 existieren außerdem auch eine Aufnahme des Mobiltelefons unter einer Bettdecke und eine Aufnahme bei geöffnetem Fenster.

3.2 Plots

Zunächst wenden wir uns der Rauschextraktion zu. In Abbildung 1³ ist dieser Prozess dargestellt. Dabei ist zu beachten, dass das linkeste Diagramm im dargestellten Verlauf nicht genau der Differenz der ersten beiden entspricht, da als Zwischenschritt noch die Multiband Spectral Subtraction angewendet wurde, die wie bereits erwähnt zwar die Voice Leakage, damit aber auch das Energilevel reduziert. Im oberen Beispiel von Abbildung 1 ist eine zu starke Entfernung von Sprachsignalen erkennbar, die zu „Löchern“ im Rauschen führt. Dieser Effekt war in unseren Tests jedoch nur selten sichtbar. Während im Originalton und im rauschreduzierten Signal jeweils klar Sprache sichtbar ist (ca. Sekunden 9-25), ist dies im reinen Rauschen rechts stark reduziert. Akustische Tests bestätigen, dass zwar keine vollständig präzise Eliminierung aller Sprache erreicht wurde, diese aber stark reduziert wurde und auch mit Lautstärke-Erhöhung kaum noch verständlich ist.

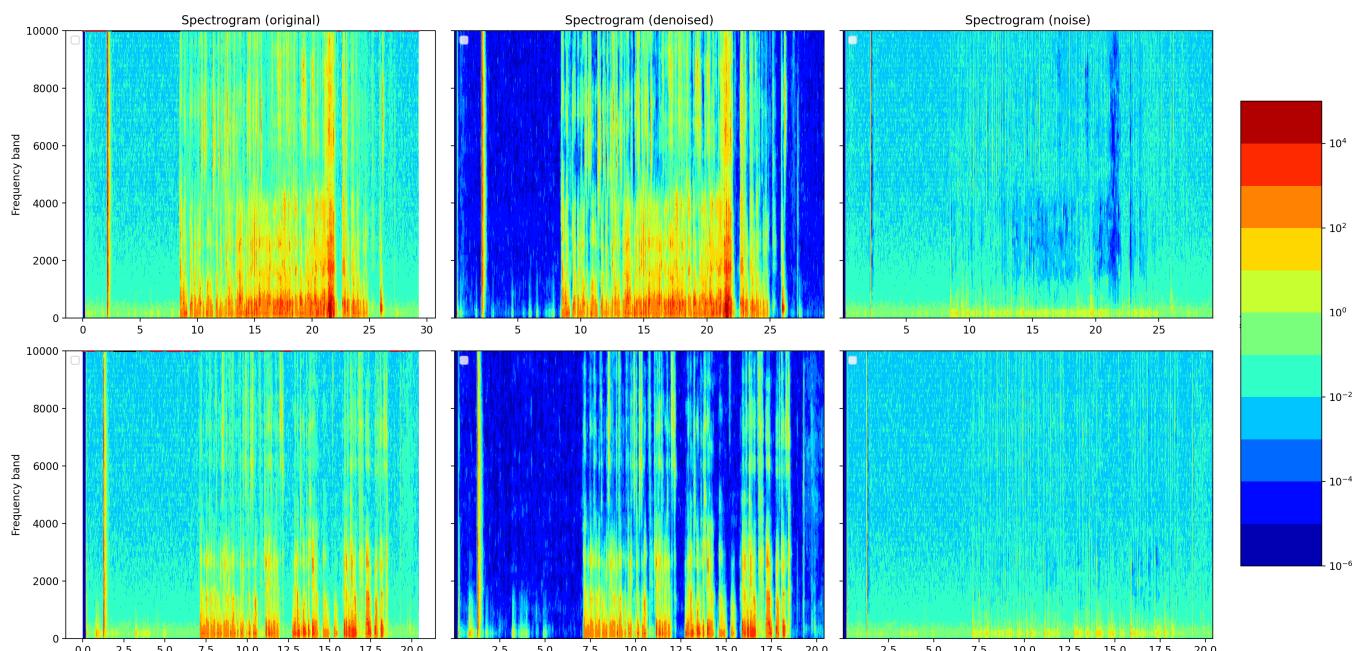


Abbildung 1: Darstellung der Rauschextraktion von zwei Aufnahmen (obere bzw. untere Zeile) aus der Umgebung „Raum 2“. Es handelt sich jeweils um Zeit-Frequenzplots bei denen die Farben das jeweilige Energilevel der entsprechenden Frequenz zum entsprechenden Zeitpunkt darstellen. Von links nach rechts sind jeweils erst Originaldaten, dann das rauschreduzierte Signal und schließlich das reine Rauschen nach Multiband Spectral Subtraction dargestellt. Auf der x-Achse sind jeweils Sekunden, auf der y-Achse Hertz aufgetragen.

Um mehr Informationen über die erste Stufe der Rauschdetektion zu erhalten, ist in den linken Plots von Abbildung 1 und allen Plots von Abbildung 2 die Klassifizierung in Sprache und reines Rauschen oben mit dünnen Strichen farblich dargestellt. Dabei stellt grün Sprachsequenzen, rot reines Rauschen und schwarz das längste, also zur weiteren Verarbeitung genutzte, Intervall reinen Rauschens dar. Mit kleinen Unterbrechungen entspricht dies weitgehend der intuitiven Klassifizierung anhand der akustischen Beurteilung und kann ansonsten auch in der Sensibilität angepasst werden. In der Umgebung „Platz“ sind die erkannten Rauschintervalle teils recht kurz, da starker Wind teilweise als Sprache klassifiziert wurde. In Tests mit einem anderen Aufnahmegerät funktionierte die Zuordnung teils schlechter und klassifizierte zu oft als Sprache, was zu wenig reinen Rauschintervallen führte. Erklärt werden kann dies unter Umständen mit einem dort aktivierten Software-Rauschfilter, der zu unreinen Audiosignalen führte.

³Für die Kombinierung mehrerer Plots in einer Grafik und die Bereitstellung der TU-Farben nutzen wir eine Bibliothek aus [Dam20].

Als nächstes betrachten wir nun also die Rauschsignale verschiedener Aufnahmen an verschiedenen Umgebungen in Abbildung 2. Teilweise sind in diesen Plots in den Spalten stärkere Gemeinsamkeiten erkennbar, die für die spätere Klassifizierung vielversprechend sind. Teilweise wirken die Spektrogramme verschiedener Orte allerdings auch sehr ähnlich und weisen auch innerhalb der Spalten Abweichungen auf. Wie zu erwarten, sind sich beispielsweise die Umgebungen „Raum 1“ und „Raum 2“ recht ähnlich. Akustisch sind diese ebenfalls nicht zu unterscheiden, die Umgebungen „Platz“ (durch starken Wind) und „Straße“ (durch lokalen Geräuschanstieg durch Autos) heben sich akustisch noch meist ab.

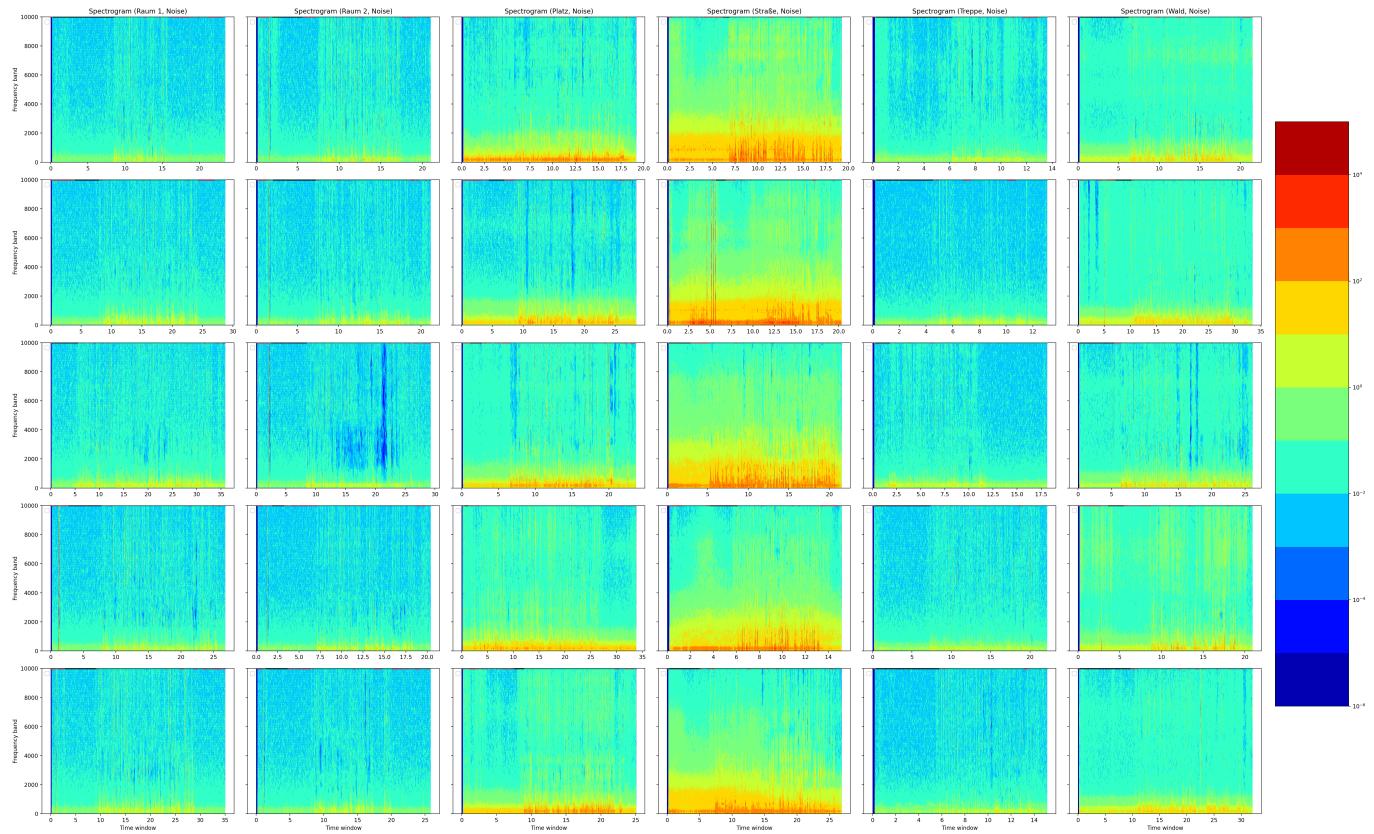


Abbildung 2: Darstellung der Rauschapproximationen verschiedener Aufnahmen (Spalten sind Umgebungen, Zeilen verschiedene dort aufgenommene Aufnahmen). Es handelt sich jeweils um Zeit-Frequenzplots bei denen die Farben das jeweilige Energielevel der entsprechenden Frequenz zum entsprechenden Zeitpunkt darstellen. Die Multiband Spectral Subtraction war hier deaktiviert, da die Unterschiede dadurch optisch besser erkennbar sind. Auf der x-Achse sind jeweils Sekunden, auf der y-Achse Hertz aufgetragen.

Schließlich wenden wir uns noch dem Vergleich der Umgebungs-Rauschapproximationen zu. Um die Umgebungen zu vergleichen, haben wir für jede dieser Umgebungen einen Vektor von Energie in verschiedenen Frequenzbändern (vgl. Abschnitt 2.3). Dieser ist in Abbildung 3 für unsere Umgebungen dargestellt. Die Umgebung „Straße“ hebt sich hier schon durch hohe Intensität deutlich ab. „Platz“ und „Wald“ haben jeweils ein Maximum um $7 \cdot 10^3$ Hz, unterschieden sich aber auch im Verlauf in Richtung niedriger Frequenzen. Die drei verbleibenden und im gleichen Haus aufgenommenen Aufnahmen unterscheiden sich hingegen weniger deutlich. Hier zeichnen sich Schwierigkeiten bei der Klassifizierung der Aufnahmen ab.

In allen Graphen ist ein starker Abfall der Intensitäten in hohen Frequenzen zu verzeichnen, der sowohl durch wenig akustischen Signalen in diesem Bereich als auch durch Schwächen des Aufnahmegerätes erklärbar ist. Außerdem haben alle Umgebungen einen starken punktuellen Anstieg bei etwa $1,5 \cdot 10^4$ Hz. Da dieser bei Referenzaufnahmen mit anderen Geräten nicht auftritt, ist dies sicherlich auf eine Schwäche des Aufnahmegeräts,

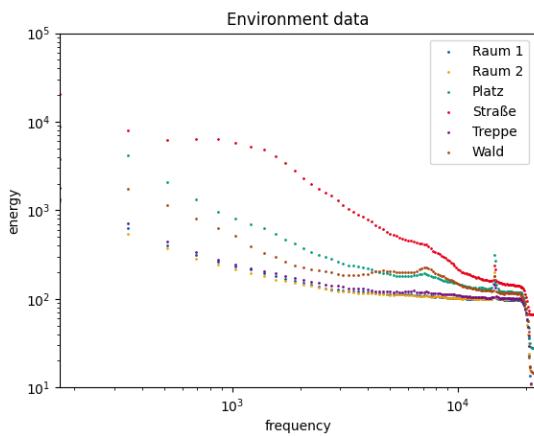


Abbildung 3: Darstellung der Rauschapproximationen verschiedener Umgebungen (eine Farbe pro Umgebung). Es handelt sich um einen Frequenz-Intensitätsplot. Die Multiband Spectral Subtraction war hier deaktiviert, die Unterschiede zu aktiver Multiband Spectral Subtraction sind nicht groß. Auf der x-Achse sind Hertz aufgetragen.

beispielsweise durch eine Eigenfrequenz zurückzuführen.

Bei Betrachtung der Unterschiede zwischen den einzelnen Aufnahmen einer Umgebung (Abbildung 4) stellen wir fest, dass auch hier Unterschiede erkennbar sind, diese jedoch in den meisten Fällen geringer ausfallen, als die zwischen den Umgebungen. Eine Unterscheidung der Räume und des Treppenhauses scheint weiterhin schwierig, da die Abweichungen zwischen diesen Abweichungen die Differenzen aus Abbildung 3 zu übersteigen scheinen.

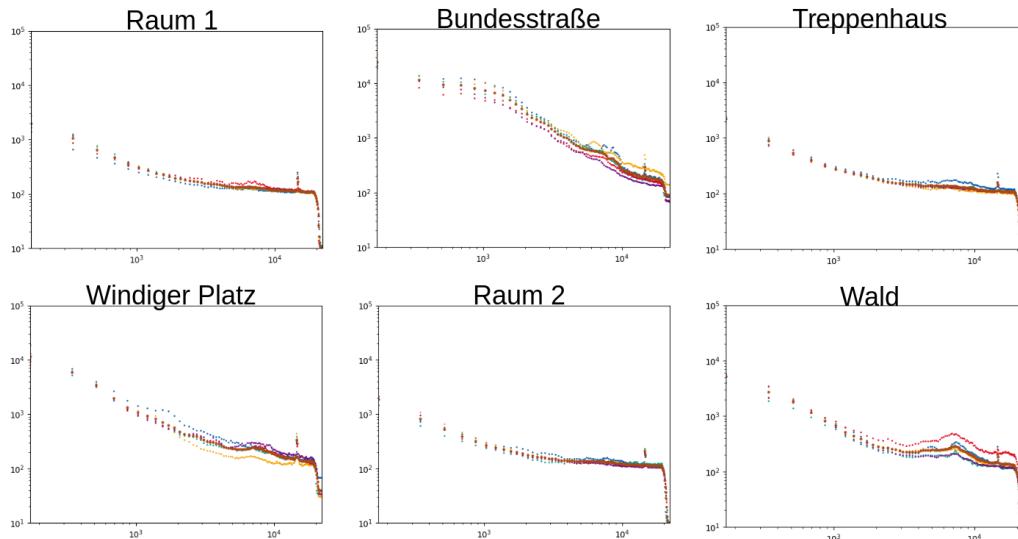


Abbildung 4: Darstellung der Rauschapproximationen der einzelnen Aufnahmen der Umgebungen (eine Farbe pro Aufnahme). Es handelt sich um Frequenz-Intensitätsplots. Auf den x-Achsen sind Hertz aufgetragen.

3.3 Erkennung

Zum Test der Umgebungserkennung haben wir die aus den Umgebungen aus Abschnitt 3.1 generierten Rauschapproximationen erstellt. Zunächst versuchen wir nun die einzelnen dieser Aufnahmen den Umgebungen wieder zuzuordnen⁴. Die Ergebnisse unterscheiden sich mit den verschiedenen in Abschnitt 2.3 beschriebenen Parametern, zwei gute Ergebnisse sind in Abbildung 5 dargestellt. Im linken Beispiel wurden einige Umgebungen richtig erkannt. Vor allem wurden jedoch einige in „Raum 1“ aufgenommene Aufnahmen zu „Raum 2“ zugeordnet was aufgrund der ähnlichen Eigenschaften der Räume wenig überraschend ist. Im rechten Beispiel funktioniert die Klassifizierung der beiden Räume etwas besser, dafür gibt es im Treppenhaus einige Fehlklassifizierungen. Die verschiedenen Parameter erzielen an verschiedenen Stellen also unterschiedlich gute Ergebnisse. Bei Fehlklassifizierungen sind die richtigen Umgebungen jedoch meist noch grün was eine immer noch geringe Abweichung (also kleine Zahl) darstellt.

Room 1							Room 1						
1.35E+04	1.10E+06	7.27E+06	1.32E+04	3.12E+04	1.57E+05		1.92E+00	2.99E+00	7.80E+00	1.92E+00	1.97E+00	2.46E+00	
1.85E+04	1.03E+06	7.09E+06	1.84E+04	3.97E+04	1.49E+05		1.96E+00	2.91E+00	7.47E+00	1.97E+00	1.99E+00	2.42E+00	
4.41E+04	8.66E+05	6.65E+06	4.46E+04	4.50E+04	1.25E+05		1.95E+00	2.76E+00	7.05E+00	1.96E+00	1.98E+00	2.34E+00	
1.82E+04	1.03E+06	7.13E+06	1.78E+04	3.67E+04	1.46E+05		2.00E+00	3.09E+00	8.08E+00	2.01E+00	2.05E+00	2.53E+00	
1.59E+04	1.08E+06	7.22E+06	1.59E+04	4.02E+04	1.61E+05		1.97E+00	3.03E+00	7.88E+00	1.98E+00	2.03E+00	2.50E+00	
Windy place							Windy place						
7.49E+06	3.58E+06	3.90E+06	7.51E+06	6.99E+06	5.87E+06		3.14E+00	2.07E+00	2.78E+00	3.20E+00	2.87E+00	2.32E+00	
1.03E+06	5.53E+05	4.05E+06	1.04E+06	8.55E+05	5.19E+05		2.31E+00	2.13E+00	4.33E+00	2.34E+00	2.25E+00	2.21E+00	
1.24E+06	5.86E+05	3.96E+06	1.25E+06	1.05E+06	6.79E+05		2.46E+00	2.07E+00	3.81E+00	2.49E+00	2.37E+00	2.14E+00	
1.80E+06	6.71E+05	3.63E+06	1.80E+06	1.59E+06	1.10E+06		2.53E+00	2.05E+00	3.58E+00	2.57E+00	2.40E+00	2.12E+00	
2.18E+06	7.29E+05	3.64E+06	2.18E+06	1.91E+06	1.35E+06		2.67E+00	2.08E+00	3.45E+00	2.71E+00	2.51E+00	2.14E+00	
Street							Street						
1.03E+07	6.92E+06	3.92E+06	1.04E+07	9.95E+06	8.82E+06		6.91E+00	3.43E+00	2.11E+00	7.17E+00	6.42E+00	4.29E+00	
2.92E+07	2.08E+07	1.10E+07	2.93E+07	2.83E+07	2.59E+07		7.33E+00	3.43E+00	2.04E+00	7.57E+00	6.69E+00	4.42E+00	
1.55E+07	1.01E+07	5.19E+06	1.56E+07	1.50E+07	1.34E+07		6.33E+00	3.22E+00	2.07E+00	6.54E+00	5.84E+00	3.98E+00	
8.38E+06	4.45E+06	2.87E+06	8.42E+06	8.00E+06	6.80E+06		4.77E+00	2.62E+00	2.13E+00	4.91E+00	4.40E+00	3.14E+00	
8.98E+06	5.00E+06	2.95E+06	9.01E+06	8.54E+06	7.31E+06		5.01E+00	2.69E+00	2.12E+00	5.20E+00	4.65E+00	3.27E+00	
Room 2							Room 2						
1.95E+04	1.03E+06	7.11E+06	1.91E+04	3.78E+04	1.50E+05		1.94E+00	2.84E+00	7.29E+00	1.94E+00	1.96E+00	2.37E+00	
2.59E+04	9.71E+05	6.95E+06	2.61E+04	4.14E+04	1.36E+05		1.92E+00	2.79E+00	7.23E+00	1.92E+00	1.95E+00	2.34E+00	
1.92E+04	1.01E+06	7.09E+06	1.87E+04	3.40E+04	1.38E+05		2.08E+00	3.28E+00	8.69E+00	2.06E+00	2.14E+00	2.68E+00	
1.92E+04	1.01E+06	7.07E+06	1.90E+04	3.72E+04	1.44E+05		1.95E+00	2.96E+00	7.76E+00	1.94E+00	1.98E+00	2.45E+00	
1.45E+04	1.09E+06	7.26E+06	1.43E+04	3.59E+04	1.59E+05		1.93E+00	3.03E+00	7.96E+00	1.93E+00	1.98E+00	2.48E+00	
Stairs							Stairs						
4.96E+04	8.27E+05	6.60E+06	4.90E+04	4.16E+04	1.05E+05		1.95E+00	2.52E+00	6.30E+00	1.95E+00	1.93E+00	2.15E+00	
4.01E+04	8.62E+05	6.70E+06	3.96E+04	3.59E+04	1.04E+05		1.91E+00	2.80E+00	7.16E+00	1.91E+00	1.92E+00	2.35E+00	
4.95E+04	8.09E+05	6.56E+06	4.95E+04	4.00E+04	1.02E+05		1.91E+00	2.68E+00	6.90E+00	1.92E+00	1.92E+00	2.28E+00	
3.47E+04	8.70E+05	6.72E+06	3.44E+04	3.41E+04	1.05E+05		1.91E+00	2.75E+00	7.07E+00	1.91E+00	1.92E+00	2.31E+00	
4.40E+04	8.36E+05	6.61E+06	4.36E+04	3.84E+04	1.04E+05		1.92E+00	2.72E+00	6.92E+00	1.92E+00	1.93E+00	2.30E+00	
Forest							Forest						
2.97E+05	5.78E+05	5.32E+06	2.97E+05	2.10E+05	1.47E+05		2.27E+00	2.15E+00	4.38E+00	2.29E+00	2.19E+00	1.99E+00	
3.47E+05	5.81E+05	5.19E+06	3.50E+05	2.59E+05	1.59E+05		2.21E+00	2.17E+00	4.57E+00	2.24E+00	2.14E+00	2.01E+00	
1.33E+05	7.03E+05	5.98E+06	1.34E+05	8.90E+04	1.23E+05		2.08E+00	2.40E+00	5.55E+00	2.09E+00	2.05E+00	2.12E+00	
3.60E+05	6.54E+05	5.38E+06	3.60E+05	2.68E+05	1.98E+05		2.71E+00	2.39E+00	4.00E+00	2.74E+00	2.58E+00	2.20E+00	
4.30E+05	4.99E+05	4.92E+06	4.33E+05	3.29E+05	1.77E+05		2.10E+00	2.12E+00	4.70E+00	2.12E+00	2.04E+00	2.01E+00	

(a) y linear, squared error

(b) y logarithmic, linear error

Abbildung 5: Ergebnis der Umgebungserkennung auf den verschiedenen Audiodateien. Links wurden die Standardparameter aus Abschnitt 2.3 genutzt, rechts wurde eine lineare Abweichungsberechnung und eine logarithmische y-Skalierung genutzt. Pro Zeile sind jeweils die Werte einer Aufnahme angegeben, die Umgebung der folgenden 5 Aufnahmen ist jeweils über den Blöcken angegeben. In den Spalten sind jeweils die Abweichungswerte zu den einzelnen Umgebungen dargestellt, die Umgebungen sind hier in der gleichen Reihenfolge wie der Block dargestellt. Kleine Zahlen entsprechen einer höheren Wahrscheinlichkeit, dass diese Umgebung die der Aufnahme war, die wahrscheinlichste Umgebung ist jeweils eingeklammert. Optimal wären die eingerahmten Einträge im n-ten Block in der n-ten Spalte.

⁴Hierbei sind die zuzuordnenden Testdaten auch in der Umgebungsgenerierung genutzt worden. Durch die vorherige Median-Bildung über die Werte aller Fenster in allen dort aufgenommenen Aufnahmen, gehen wir jedoch davon aus, dass die Ergebnisverfälschung dadurch gering ist. Tests bei denen nur auf den anderen Aufnahmen des Ortes die Umgebung generiert wurde oder neue Aufnahmen zugeordnet wurden, führen zu ähnlichen Ergebnissen.

Um die Störabweichung zu untersuchen haben wir in „Raum 2“ jeweils zwei Aufnahmen mit unter der Bettdecke liegendem Mobiltelefon, bei geöffneten Fenster und mit den alten Bedingungen gemacht. Die Rauschapproximation der Umgebung basiert weiterhin auf den alten Aufnahmen. Die Ergebnisse sind in Abbildung 6 dargestellt. Dabei wurde die Aufnahmen unter der Bettdecke und die unter den alten Bedingungen korrekt zugeordnet. Bei geöffnetem Fenster wurden die Aufnahmen der Umgebung „Treppenhaus“ zugeordnet. Da diese jedoch bei geöffnetem Fenster im gleichen Gebäude aufgenommen wurde, ist dieses Resultat jedoch überraschend.

			Correct env		
<i>Blanket</i>					
	1.40E+04	1.10E+06	7.30E+06	1.36E+04	2.81E+04
	2.13E+04	1.06E+06	7.24E+06	2.10E+04	3.42E+04
<i>Normal</i>					
	1.33E+04	1.15E+06	7.39E+06	1.32E+04	3.52E+04
	1.30E+04	1.16E+06	7.42E+06	1.28E+04	3.56E+04
<i>Window opened</i>					
	4.10E+04	8.78E+05	6.61E+06	4.21E+04	4.08E+04
	5.37E+04	8.23E+05	6.44E+06	5.56E+04	4.75E+04
			Correct env		

Abbildung 6: Ergebnis der Umgebungserkennung auf den verschiedenen Audiodateien. Pro Zeile sind jeweils die Werte einer Aufnahme angegeben, die Eigenschaften der folgenden 2 Aufnahmen ist jeweils über den Blöcken angegeben. In den Spalten sind jeweils die Abweichungswerte zu den einzelnen Umgebungen dargestellt, die Umgebungen sind hier wie in Abbildung 5. Kleine Zahlen entsprechen einer höheren Wahrscheinlichkeit, dass diese Umgebung die der Aufnahme war, die wahrscheinlichste Umgebung ist jeweils eingerahmt. Optimal wären eingerahmte Einträge in der 4-ten Spalte („Raum 2“).

3.4 Resümee

Unsere Rauschextraktion macht es für menschliche Hörer*innen meist schwer noch Teile der Sprache zu verstehen bzw. erfordert dazu eine deutliche Verstärkung des Signals. Subjektiv wirkt die Rauschentfernung ohne aufwändige Konfiguration besser als die beispielsweise in Audacity [Aud] implementierte was an flexiblerer Implementierung der genutzten Funktionen und der automatischen Detektion eines geeigneten Rauschintervalls liegen könnte. Die Multiband Spectral Subtraction führte zu weniger Voice Leakage, reduzierte aber auch die Gesamtenergie – insgesamt funktionierten die weiteren Funktionen unserer Software nach Anwendung von Multiband Spectral Subtraction nicht wesentlich besser.

Die Erkennung der Umgebungen funktioniert mit den meisten Parametern recht zuverlässig auf sehr unterschiedlichen Umgebungen, bei ähnlichen Umgebungen wie etwa den beiden benachbarten Räumen funktioniert sie teils nicht mehr oder nur mit wenigen Parametern.

Der Vergleich mit [IM10] fällt schwer, da nach Rauschunterdrückung zwar ähnliche Plots zu Abbildung 1 angegeben werden, uns jedoch keine Audiodaten vorliegen. Aus den Plots gehen wir davon aus, dass die Ergebnisse unserer ähnlichen Rauschentfernungsansätze ähnlich gut sind. Die Umgebungserkennung ist schon bei leicht unterschiedlichen Aufnahmen kaum noch vergleichbar was eine Beurteilung der Unterschiede schwierig macht. Die Detektion von Manipulationen innerhalb einzelner Dateien ist bei uns derzeit nicht implementiert, siehe Abschnitt 4.

4 Mögliche Erweiterungen

Hier werden ein paar Ideen zur Erweiterung und Verbesserung unserer Software kurz vorgestellt, die über den Rahmen dieses Praktikums hinausgehen.

Zeitliche Änderungen Viele Geräusche treten nicht dauerhaft sondern immer wieder kurz auf, beispielsweise vorbeifahrende Autos auf einer Bundesstraße. Diese werden durch die aktuelle Median-Bildung über den zeitlichen Verlauf nicht optimal abgebildet und könnten durch neue Ansätze besser in die Erkennung einbezogen werden.

Verschiedene Aufnahmegeräte Um Auswirkungen verschiedener Aufnahmegeräte besser analysieren zu können, müsste zunächst ein insgesamt niedrigeres Energielevel einer Aufnahme ausgeglichen werden, um nur relative Unterschiede zwischen den Frequenzbändern zu betrachten. Anschließend könnten durch weitere Maßnahmen mehr Verbesserungen zum Ausgleich aufnahmegerätabhängiger Effekte ergriffen werden, etwa ein Ausgleich der in Abbildung 3 festgestellten lokalen Ausreißer bei $1,5 \cdot 10^4$ Hz.

Grenzwert Die aktuelle Implementierung sucht nach der bestpassendsten aus mehreren bekannten Umgebungen. Wünschenswert wäre ein Plausibilitätswert für eine Umgebung, der von anderen Umgebungen unabhängig ist. Aktuell wäre ein solcher Wert stark von den Parametern abhängig, funktioniert aber auch nur begrenzt. Insgesamt ist hierzu vermutlich eine weitere Verfeinerung der Metrik mit neuen Ansätzen nötig.

Wahrscheinlichkeiten Beispielsweise um Grenzwerte zu realisieren, hätten wir gerne eine Wahrscheinlichkeitsverteilung, um Konfidenzintervalle anzugeben, in denen wir eine Aufnahme einer Umgebung zuordnen. Da wir durch die einzelnen Fenster viele Werte haben wie stark die einzelnen Frequenzen in einer Aufnahme enthalten sind, könnten wir diese als eine Gauß-Glocke pro Frequenz approximieren und anschließend Aufnahmen durch die Positionen in diesen Gauß-Glocken zuordnen. Da dieser Ansatz jedoch zunächst alle Frequenz-Intensitäten als unabhängig ansehen würde, müsste dieser mathematisch noch weiter entwickelt werden, um ihn einsatzbereit zu machen.

Manipulationserkennung Neben der Abschätzung, ob eine Umgebung zu einer Aufnahme plausibel ist oder der Zuordnung von Aufnahmen zu Umgebungen ist auch die Erkennung von Schnitten bzw. Manipulationen in Aufnahmen auf Basis von Rauschen forensisch interessant. Für diese sind mehrere Ansätze denkbar.

[IM10] untersucht die Ähnlichkeit von Gesamtaufnahmen zu verschiedenen Umgebungen und vermutet bei mehreren ähnlichen Umgebungen, dass Anteile beider Umgebungen enthalten sind. Dazu ist es in unserer Implementierung vermutlich nötig den Median über die Frequenzen der Fenster der Aufnahme durch einen Mittelwert zu ersetzen, um nicht kurze starke Abweichungen direkt zu eliminieren. Der Ansatz hat jedoch die Nachteile, dass die potenzielle Umgebung einer Manipulation bekannt sein muss und die Tests funktionierten nachdem dort recht große Anteile hineinkopiert wurden. Bei kurzen manipulierten Stellen, könnte das deutlich schlechter funktionieren.

Ein weiterer Ansatz wäre starke Abweichungen zwischen verschiedenen, kurzen Fenstern zu untersuchen und Manipulationen zu vermuten, wenn ein oder mehrere benachbarte Fenster stark von den übrigen Fenstern abweichen. Dieser Ansatz könnte jedoch auch bei verbleibender Voice Leakage, also übrigen Sprachanteilen oder zu stark reduzierter Sprache eine Manipulation vermuten.

5 Bedienung

5.1 Einrichtung

1. Erstellen Sie eine virtuelle Umgebung mit dem Befehl `virtualenv $name`
2. Aktivieren Sie diese mit `source $name/bin/activate`
3. Installieren Sie alle erforderlichen Pakete mit `pip install -r requirements.txt`.

Nach Änderung des Codes müssen die erforderlichen Pakete ggf. durch `pip freeze > requirements.txt` angepasst werden.

5.2 Ausführung

Durch Ausführen von `python main.py` im Ordner `src` wird der Code ausgeführt. Dazu müssen vorher die Parameter angepasst werden und ggf. die benötigten Audiodateien entsprechend der Parameter und Ordnerstruktur hinterlegt werden.

Ein Beispielskript zur Ausführung unter Linux wird bereitgestellt.

5.3 Ordnerstruktur

Es gibt die folgenden Ordner:

`src`: Hier liegt der Quellcode inklusive der `main.py`.

`Audio`: Hier liegen die einzugebenden Audiodateien.

`results`: Hier werden nach Ausführung der Software alle erstellten Dateien abgelegt. Dazu wird für jede Ausführung ein neuer Ordner mit der Run-ID (aufsteigende Zahlen) angelegt. In diesen liegen dann die Plots/Spektrogramme und Ausgabewerte (CSV), sowie Log-Dateien und Konfigurationsinformationen. Für die Audiodateien wird lediglich ein Pfad auf den Ordner `_resources` in der Datei `run.js` angegeben, damit identische Dateien aus mehreren Ausführungen nicht mehrfach abgelegt werden.

`results`: Hier liegt die Projektdokumentation sowie eine Visualisierungsvorlage.

5.4 Parametrisierung

In der Datei `src/main.py`, können Funktionalitäten (de-)aktiviert werden und Parameter, wie bspw. die zu verarbeitenden Audiodateien) gesetzt werden. Im Code sind kurze Beschreibungen der einzelnen Einstellungen gegeben.

Audio (Code-Kommentar „`Comparision metric parameters`“): Hier werden die Audiodateien festgelegt. Dazu gibt es ein Mapping von bekannten Umgebungen zu den dort erstellten Aufnahmen. Anschließend können weitere Aufnahmen festgelegt werden, die nun (so wie auch die Dateien zur Umgebungsgenerierung) den Umgebungen zugeordnet werden. Außerdem kann die Generierung von Audiodateien (ohne Rauschen / reines Rauschen) (de-)aktiviert werden.

Plots (Code-Kommentar „`Graph properties`“): Hier können Einstellungen zu den Grafiken festgelegt werden, also bspw. Dateinamen, Aktivierung des Multipass-Ansatzes usw.

Metriken (Code-Kommentar „`Comparision metric parameters`“): Die in Abschnitt 2.3 genannten Metrikanpassungen können eingestellt werden.

5.5 Interpretation

Da auf die meisten Ausgabedaten schon in der Dokumentation näher eingegangen wurde, konzentrieren wir uns hier auf die numerischen Ergebnisse in der Datei `environment_errors.csv` in den Ergebnissen. Hier stellen die Zeilen die zuzuordnenden Dateien dar und die jeweils kleinste Zahl entspricht der wahrscheinlichsten Umgebung für die Datei (s. Abschnitt 3.3 für mehr Details). Mehrere kleine Zahlen zeigen weitere wahrscheinliche Umgebungen auf oder deuten auf Audioanteile beider Umgebungen in der Datei hin. Zur Visualisierung entsprechend Abbildung 5 können die Werte in die Vorlage in `docs/data_visualization_template.ots` kopiert werden.

Literatur

- [Aud] Audacity. *Audacity audio software*. en-US. URL: <https://www.audacityteam.org> (besucht am 21.02.2021).
- [BP20] A. Stefanie Blümer und Tim Pollandt. *pollti/P4FM*. en. 2020. URL: <https://github.com/pollti/P4FM> (besucht am 20.02.2021).
- [Dam20] Fabian Damken. *fdamken/bachelors-thesis_code*. en. 2020. URL: https://github.com/fdamken/bachelors-thesis_code (besucht am 20.02.2021).
- [IM10] S. Ikram und H. Malik. „Digital audio forensics using background noise“. In: *2010 IEEE International Conference on Multimedia and Expo*. ISSN: 1945-788X. Juli 2010, S. 106–110. doi: [10.1109/ICME.2010.5582981](https://doi.org/10.1109/ICME.2010.5582981).
- [KL02] Sunil D Kamath und Philipos C Loizou. „A multi-band spectral subtraction method for enhancing speech corrupted by colored noise“. en. In: *ICASSP*. Bd. 4. Dallas, Texas, USA: Citeseer, 2002, S. 44164–44164.
- [Wis] John Wiseman. *webrtcvad: Python interface to the Google WebRTC Voice Activity Detector (VAD)*. URL: <https://github.com/wiseman/py-webrtcvad> (besucht am 20.02.2021).
- [Zav] Esfandiar Zavarehei. *Multi-band Spectral Subtraction*. en. URL: <https://de.mathworks.com/matlabcentral/fileexchange/7674-multi-band-spectral-subtraction> (besucht am 20.02.2021).