Entwicklung einer Methodik zur automatisierten Detektion von Signaltönen im Straßenverkehr

VKA - RWTH Aachen

Michael Kandzia, Thomas Bakaj

6. April 2020

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Häufig verwendete Abkürzungen

AED	Acoustic Event Detection
CNN	Convolutional Neuronal Network
CRNN	Convolutional Recurrent Network
DFT	Diskrete Fourier Transformation
FFT	Schnelle Fourier Transformation, engl. Fast-Fourier-Transformation
LSTM	Long Short-Term Memory
MFCC	Mel-Frequency Coefficients
MLP	Multi Layer Perceptron
NN	Neuronales Netz
ReLU	Rectifier Linear Unit
RNN	Recurrent Neural Network
STFT	Kurzzeit-Fourier-Transformation, engl. Short-Time-Fourier-Transformation

Verwendete Formelzeichen

E	Schallenergiedichte
N	Blockgröße
T	Periode
$\vec{F}$	Vektor einer Schnellen-Fourier-Transformation
$\vec{I}$	Schallintensität
$\vec{v}$	Schallschnelle
c	Schallgeschwindigkeit
f	Frequenz
f_S	Abtastrate, Abtastfrequenz
n	Bitzahl
p	Schalldruck

1 Einleitung

Autonomes Fahren gewinnt zunehmend an Popularität. Bis zur Vollautomatisierung der Fahrzeuge (Autonomiestufe 5), in denen der Passagier lediglich das Ziel festlegt und das System startet, gilt es noch Herausforderungen zu bewältigen. Eine dieser Herausforderungen ist die automatisierte Erkennung von sicherheitsrelevanten Sonder- und Warnsignalen, beispielsweise dem sogenannten Martinshorn in Deutschland oder dem Wail-Signal in den Vereinigten Staaten. Im Rahmen dieser zweiteiligen Projektarbeit soll eine autarke Methodik entwickelt werden, welche die Passagiere vor Gefahrensituationen warnen kann.

Die Projektarbeit beschränkt sich bei der Erkennung auf deutsche und US-Signaltöne. Für beide Länder existiert eine gute Datenlage hinsichtlich Aufnahmen der Signaltöne und die Arten der verwendeten Signaltöne sind repräsentativ für weitere Länder. Zur Entwicklung der Methodik gilt es im ersten Schritt die akustischen Merkmale der Signaltöne zu erfassen. Die Methoden zur akustischen Ereignisdetektion werden anhand einer Literaturrecherche evaluiert. Basierend auf der Literraturrecherche werden Neuronales Netze zur Erkennung der Signaltöne entwickelt. Teil der Neuronalen Netze ist eine Datenvorverarbeitung. Parallel hierzu wird eine Datenbank mit annotierten Trainingsdaten aufgebaut.

Anhand der identifizierten Merkmale und der Datenbank werden Ereignisdetektoren für ausgewählte Sondersignale trainiert. Anschließend wird der Algorithmus anhand von Testdaten validiert. In der Auswertung der Ergebnisse der Arbeit werden die Datenvorverarbeitung und unterschiedlich konfigurierte Neuronale Netze bewertet. Abschließend wird für jedes Land ein finaler Ereignisdetektor vorgestellt.

2 Stand der Technik

Im diesem Kapitel werden Grundlagen vorgestellt, die zum Verständnis des Weges akustischer Sondersignale von Einsatzfahrzeugen bis zum Ausgang eines Neuronalen Netzes relevant sind. Im Einzelnen werden der Luftschall, die Verarbeitung akustischer Signale, Neuronale Netze, die Eigenschaften akustischer Sondersignale im Straßenverkehr und vorangegangene Forschung zum Thema dieser Arbeit erläutert.

2.1 Physikalische Akustik

Die Übertragung von akustischen Signaltönen erfolgt mithilfe von Schallwellen. In diesem Abschnitt werden die physikalischen Grundlagen des Schalls kurz erläutert.

2.1.1 Luftschall

Mechanische Schwingungen eines elastischen Mediums werden als Schall bezeichnet. Diese Schwingungen werden durch die Bewegung von Teilchen des Mediums um eine Ruhelage verursacht. Schwingungen, die sich mittels elastischer Kopplung auf benachbarte Teilchen fortpflanzen, werden **Schallwellen** genannt.

Ferner wird Schall, der sich in Luft oder anderen Gasen ausbreitet als **Luftschall** bezeichnet. Dabei äußern sich die Pendelbewegungen der Moleküle in periodischen Änderungen der Dichte und folglich des Gasdrucks [13].

2.1.2 Physikalische Größen

Ein Maß für die Intensität des Luftschalls ist der **Schalldruck** p. Dieser beschreibt die Differenz des momentanen Drucks zum vorliegenden atmosphärischen Druck.

Als **Schallschnelle** $\vec{v}$ wird die Geschwindigkeit bezeichnet, mit der die einzelnen Moleküle um ihre Ruhelage schwingen. Die Schallschnelle ist eine vektorielle Größe und entspricht der Richtung der Schallwelle.

Die Geschwindigkeit, mit der sich eine Schallwelle ausbreitet, ist die **Schallgeschwindigkeit** c. Sie ist von der Umgebungsdichte und dem Kompressionsmodul des Mediums abhängig, in dem sie sich ausbreitet [13].

Die maximale Auslenkung einer Schwingung aus der Ruhelage wird als **Amplitude** bezeichnet [14]. Eine zur Charakterisierung von Schall essenzielle Größe ist die

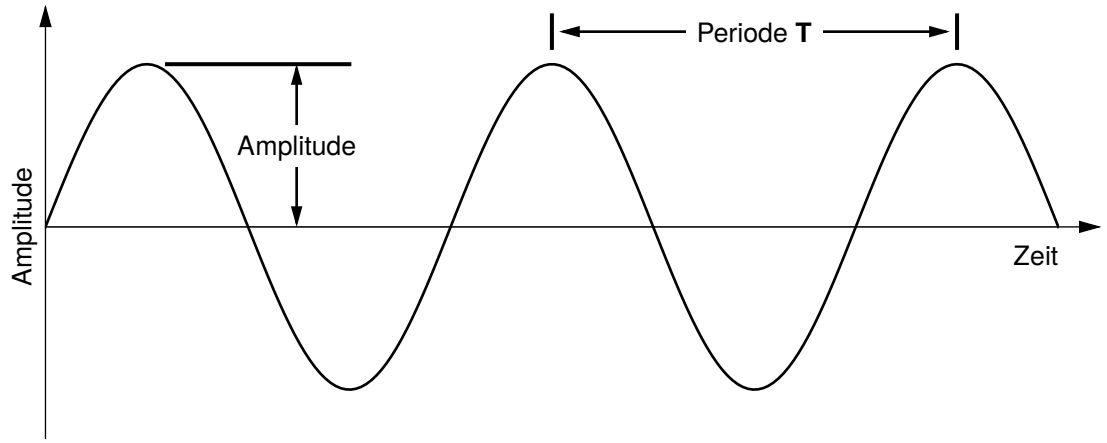


Bild 2.1: Amplitude und Periode einer Sinus-Funktion

Frequenz *f*. Mit ihr wird die Anzahl kompletter Schwingungsvorgänge pro Sekunde bezeichnet. Der Kehrwert der Frequenz ist die **Periode** *T*. Sie gibt die Dauer einer Schwingung an. Der Zusammenhang zwischen Periode und Amplitude ist in Bild 2.1 schematisch dargestellt.

Ein Maß für die Energie eines Schallereignisses ist die **Schallenergiedichte** *E*. Diese errechnet sich mit Gleichung 2.1 aus dem Schalldruck *p*, der Umgebungsdichte ρ_0, der Schallgeschwindigkeit *c* und der Schallschnelle $\vec{v}$ [15].

$$E = \frac{1}{2}\left(\frac{p^2}{\rho_0 c^2} + \rho_0 \vec{v}^2\right) \tag{2.1}$$

Die Leistung des Schalls pro Fläche bezeichnet man als die **Schallintensität** $\vec{I}$ (Gl. 2.2). Diese ergibt sich aus dem Produkt von Schallschnelle und Schalldruck [15].

$$\vec{I} = p \cdot \vec{v} \tag{2.2}$$

Das Verhältnis oder die Differenz zweier Amplituden wird meist in der Dezibel-Skala ausgedrückt. Durch ihre logarithmische Eigenschaft erlaubt die Dezibel-Skala besonders Werte eines sehr großen Zahlenbereiches anschaulich darzustellen. Für Energien und Leistungen gibt sie die Pegeldifferenz als zehnfachen Wert des dekadischen Logarithmus an. Für Größen, deren Quadrate den Energien proportional sind, wie dem Schalldruck, gibt die Skala den 20-fachen Wert an. Als Hilfsmaßeinheit für die berechneten Pegeldifferenzen wird Dezibel (dB) verwendet [14]. Somit ergibt sich:

$$10\, log\frac{P_1}{P_2} = 10\, log\frac{A_1^2}{A_2^2} = 20\, log\frac{A_1}{A_2} = \Delta\mathrm{L\ dB} \tag{2.3}$$

mit P_i – Leistung
A_i – Amplitude
ΔL – Pegeldifferenz

2.1.3 Dopplereffekt

Der Dopplereffekt wurde erstmals 1842 von Christian Doppler beschrieben. Dabei wird von einem Sender und einem Empfänger einer Schallwelle ausgegangen. Bewegen sich Sender oder Empfänger, verändert sich die gemessene Frequenz des Empfängers. Der für diese Arbeit relevante Dopplereffekt betrifft die Ausbreitung von Luftschall. Für diesen werden drei Fälle unterschieden, abhängig davon, ob sich nur der Sender bewegt, sich nur der Empfänger bewegt oder ob sich beide bewegen.

Im ersten Fall bewegt sich der Sender mit der konstanten Geschwindigkeit v_S in Richtung des Empfängers oder von diesem weg und sendet die Frequenz f_S aus. Die vom Empfänger gemessene Frequenz f_E errechnet sich aus Gleichung 2.4, wenn die Geschwindigkeit des Senders v_S positiv in Richtung des Empfängers angenommen wird. Bewegt sich der Sender auf den Empfänger zu, ist f_E größer als f_S. Entfernt sich der Sender, ist f_E kleiner als f_S.

$$f_E = f_S \cdot \frac{1}{1 - \frac{v_S}{c}} \tag{2.4}$$

Im zweiten Fall bewegt sich der Empfänger mit der Geschwindigkeit v_E auf den ruhenden Sender zu oder von ihm weg. Wird v_E positiv in Richtung des Senders angenommen, lässt sich f_E mit Gleichung 2.5 berechnen.

$$f_E = f_S \cdot \left(1 + \frac{v_E}{c}\right) \tag{2.5}$$

Aus den Gleichungen 2.4 und 2.5 lässt sich der Zusammenhang für den dritten Fall herleiten. Wird v_S positiv in Richtung des Empfängers und v_E positiv in Richtung des Senders angenommen, ergibt sich für die gemessene Frequenz f_E Gleichung 2.6 [16].

$$f_E = f_S \cdot \frac{c + v_E}{c - v_S} \tag{2.6}$$

In Tabelle 2.1 sind die empfangenen Frequenzen für verschiedene Geschwindigkeiten des Senders bei ruhendem Empfänger aufgetragen. Folglich erhöht sich im Straßenverkehr bei einer Geschwindigkeit von 120 km/h die Frequenz um 10,76%.

$v_S[km/h]$	$f_E[Hz]$
0	500
7	502,9
30	512,5
40	516,7
50	521,1
60	525,5
90	539,3
120	553,8

Tabelle 2.1: Einfluss der Dopplereffekts auf f_E bei f_S = 500 Hz

2.2 Musiktheorie

Zur Beschreibung akustischer Sondersignale ist die Musiktheorie hilfreich. Sonder- und Warnsignale bestehen nicht nur aus reinen Sinuswellen, sondern Klängen, die meist in bestimmten Frequenzverhältnissen modulieren. Folgend wird auf die Grundlagen von Ton, Klang und Intervallen eingegangen.

2.2.1 Grundlegende Begriffe

Der **Ton** bildet das Ausgangselement der Musik als Tonkunst. Dabei lässt sich dieser sowohl musikalisch als auch physikalisch betrachten. Folgend wird auf die physikalischen Eigenschaften eingegangen. Ein Ton ist reine Sinusschwingung, die durch Frequenz, Dauer und Amplitude bestimmt ist. Für den Menschen wahrnehmbare Töne liegen etwa im Bereich von 20 bis 20000 Hz.

Unter einem **Klang** versteht man das gleichzeitige erklingen mehrerer Töne. Jeder durch ein melodisches Instrument oder Gesang erzeugte umgangssprachlich genannte „Ton“ ist im physikalischen Sinne ein Klang. Er bildet ein Komplex aus vielen erklingenden Tönen. Dabei besteht ein Klang aus einem Grundton und mehreren Obertönen. Als Grundton versteht man den Ton der tiefsten Frequenz [17]. Das spezifische Gemisch an Obertönen und dessen zeitlicher Verlauf ist für die Klangfarbe (auch Timbre) entscheidend [18]. Die Tonhöhe beschreibt die Frequenz eines Tons bzw. die Frequenz eines Grundtons eines Klangs [17].

Ganzzahlige Vielfache der Grundtonfrequenz werden in der musikalischen Betrachtung als **Obertöne** bezeichnet. In der Physik werden Obertöne Harmonische einer Grundfrequenz genannt. Dabei ist die erste Harmonische die Grundfrequenz selbst. Besitzt ein Klang für den Menschen eine klar feststellbare Tonhöhe, so setzt sich dessen Klangspektrum (siehe 2.5.1) vor allem aus Harmonischen zusammen [19]. Gleichung 2.7 verdeutlicht den Zusammenhang zwischen Obertönen und Harmonischen.

$$\overbrace{f + \underbrace{2f + 3f + 4f + 5f + 6f + ...}_{\text{Obertöne}}}^{\text{Harmonische}} \tag{2.7}$$

Als **Intervall** wird in der Musik der relative Tonhöhenunterschied zweier Töne zueinander bezeichnet. Somit ist das Intervall ein Frequenzverhältnis. Das numerische Verhältnis ist dabei vom Tonsystem abhängig [17].

2.2.2 Oktave und Oktavidentität

Die Oktave ist das grundlegende Frequenzverhältnis. So werden im Allgemeinen Tonsysteme zunächst nach Oktaven unterteilt. Der Mensch schreibt Tönen im Frequenzverhältnis $2:1$ den gleichen Charakter zu, sie sind für ihn gleichwertig. Dieses Phänomen wird als Oktavidentität bezeichnet [17]. Die Wahrnehmung ist nicht erlernt, sondern bereits im Hirn des Menschen natürlich verankert. Die Gehirnforschung der letzten Jahre konnte sogar bei Affen und anderen Säugetieren die gleiche Wahrnehmung nachweisen [20].

2.2.3 Gleichstufige Stimmung

Als Stimmung wird in der Musiktheorie die Festlegung der Intervalle durch Frequenzverhältnisse bezeichnet. Die Basis der Gleichstufigen Stimmung ist die zuvor erläuterte Oktave. Somit werden zwei Töne im Frequenzverhältnis $2^n:1$ als nominell gleicher Ton betrachtet, wobei n ganzzahlig ist. Weiter wird der Bereich zwischen zwei Oktaven in zwölf sogenannte Halbtöne unterteilt. Deren Frequenzen sind immer ein $2^{\frac{n}{12}}$-faches der Frequenz des Basistons der Oktave, wobei n die Zahl des jeweiligen Intervalls ist. Folglich hat der Ton mit $n = 12$ die doppelte Frequenz des Basistons und ist der nächste Oktavwert. Das Frequenzverhältnis zweier Halbtöne beträgt immer $2^{\frac{1}{12}}$, daher der Name Gleichstufige Stimmung. [18]. Für die Gleichstufige Stimmung ergibt sich folgender Zusammenhang:

$$f_i = f_R \cdot 2^{i/12} \tag{2.8}$$

mit f_i – Frequenz des Intervalls
f_R – Referenzfrequenz
i – Gewünschtes Intervall

2.3 Mikrofon

Mikrofone ermöglichen die Umsetzung von Schall in elektrische Signale. Das Ergebnis der Umwandlung kann dargestellt, gespeichert und verwertet werden. Im engeren Sinne werden vor allem Luftschallempfänger als Mikrofone bezeichnet [21]. In Digitalkameras und Mobiltelefonen sind meist entweder sogenannte Elektret-Kondensatormikrofone oder in neueren Modellen MEMS-Mikrofone (Micro Electro Mechanical Systems) verbaut. Beide Mikrofontypen basieren auf dem grundsätzlichen Prinzip eines Kondensatormikrofons.

Das Elektret-Mikrofon besteht aus einer kleinen Kondensatorplatte und einer leitfähigen Membran. Durch Schall wird die Membran in Schwingung versetzt. Durch die Schwingung ändert sich die Kondensatorkapazität und damit auch die angelegte Spannung [22]. In Kapitel 2.4 wird erläutert wie diese Spannungsänderungen weiterverarbeitet werden.

Dieses Prinzip wird bei MEMS-Mikrofonen auf ein Mikrosystem mit Abmessungen von höchstens einigen 10 nm angewand. Diese wandeln Schallschwingungen mit Hilfe einer sehr kleinen Membran in elektrische Signale um. Ein hochintegrierter Schaltkreis wandelt das elektrische Signal in ein digitales [23].

2.3.1 Frequenzgang

Der Frequenzgang beschreibt die Empfindlichkeitsschwankungen eines Mikrofons über den gesamten Audiofrequenzbereich. Für jede Frequenz der Ausgangssignalaufnahme wird die Abweichung vom Referenzwert 0 dB angegeben [24]. Wie zuvor erläutert, stammen die verwendeten Aufnahmen zum Großteil von Telefonen und anderen Kleinelektronikgeräten. Durch unterscheidliche Frequenzgänge entstehen Verzerrungen, die vorerst nicht unbeachtet bleiben können.

Bei *Krump* [1] wurden 2012 die Frequenzgänge damals gängiger Smartphones untersucht. Außerhalb der Telefonbandbreite von 300 Hz bis 3,4 kHz weisen diese einen sehr nichtlinearen und resonanzbehafteten Frequenzgang auf. Die Messungen ergeben im Bereich unter 300 Hz eine wesentliche Abnahme im Frequenzgang und im Bereich um 6 bis 10 kHz hohe Resonanzen (siehe Bild 2.2 und Bild 2.3).

2.4 Analog-Digital-Wandlung von Audiosignalen

Um Analysen und Berechnungen eines Audiosignals durchführen zu können, müssen diese zuvor digitalisiert und gespeichert werden. In kontinuierlicher Darstellung besitzt ein Audiosignal unendlich viele Spannungswerte und ist folglich digital nicht darstell-

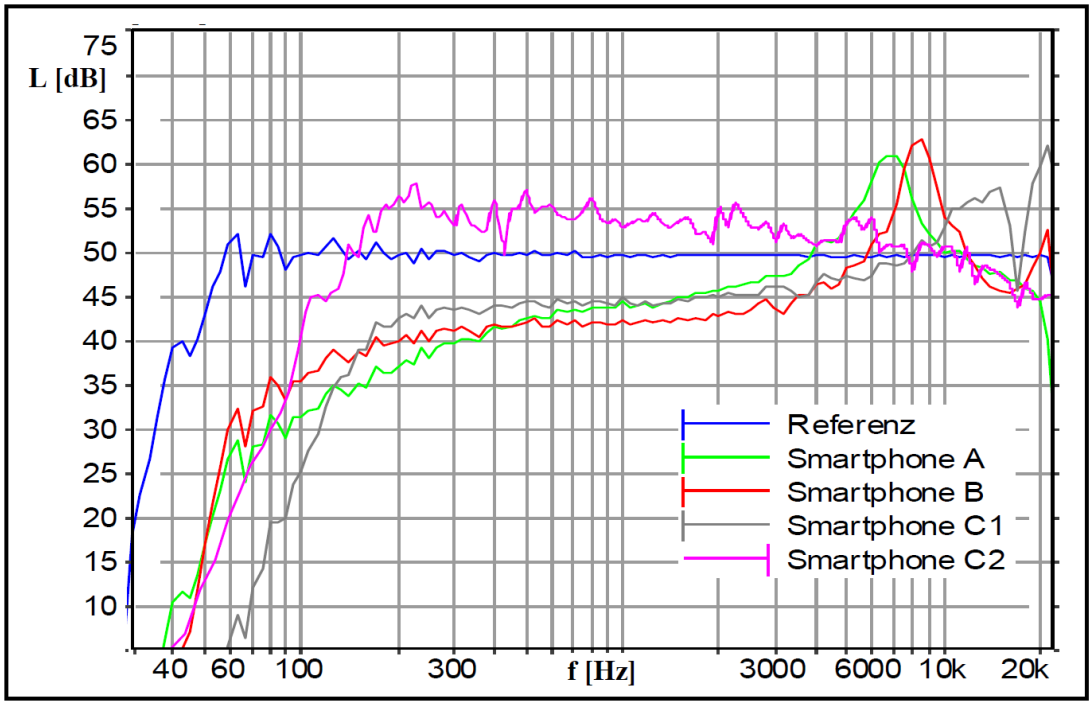


Bild 2.2: Frequenzgänge der Referenzbox und der Smartphones A, B, C1 (eingebautes Mikrophon) und C2 (externes Mikrofon) bei einem Gesamtpegel von 70 dB, nach [1]

bar. Für die digitale Darstellung werden deshalb nur einzelne elektrische Spannungen abgetastet und für eine gewisse Zeit festgehalten. Dies ist der Ausgangspunkt der nachfolgenden Quantisierung. Erst danach kann das entstandene digitale Datenpaket gespeichert werden. Dieser Prozess wird als Analog-Digital-Wandlung bezeichnet. Im Folgenden wird auf die Möglichkeiten eingegangen diese Werte zu speichern und zu verarbeiten [25].

2.4.1 Abtastrate und Quantisierung

Die Anzahl der Messungen in einem bestimmten Zeitschritt eines Signals wird als **Abtastrate** bezeichnet. Da es sich hierbei um eine Frequenz handelt, wird diese in Hertz (Hz) angegeben und alternativ auch Abtastfrequenz f_S genannt. Gängige alternative Bezeichnungen sind in der Audioverarbeitung zudem die Samplingrate oder die Samplerate. Als ein Sample oder eine Probe wird der Wert einer Abtastung bezeichnet [25].

Die bei der Abtastung ermittelten Werte sind durch den endlichen Zahlenvorrat der digitalen Speicherung begrenzt. Diese Werte werden einem definierten Raster zugeordnet und nehmen den nächstliegenden gültigen Wert an. Man bezeichnet diesen Schritt als **Quantisierung**. Je höher die Auflösung des gewählten Rasters, desto geringer ist die Abweichung des tatsächlichen Wertes. Allerdings steigt mit der Auflösung auch der Speicherbedarf [25]. Die Größe der Quantisierung wird in der digitalen Audiobearbeitung oft als Bitrate oder Bittiefe bezeichnet. Die Bitrate bezieht sich auf die Bitzahl n mit

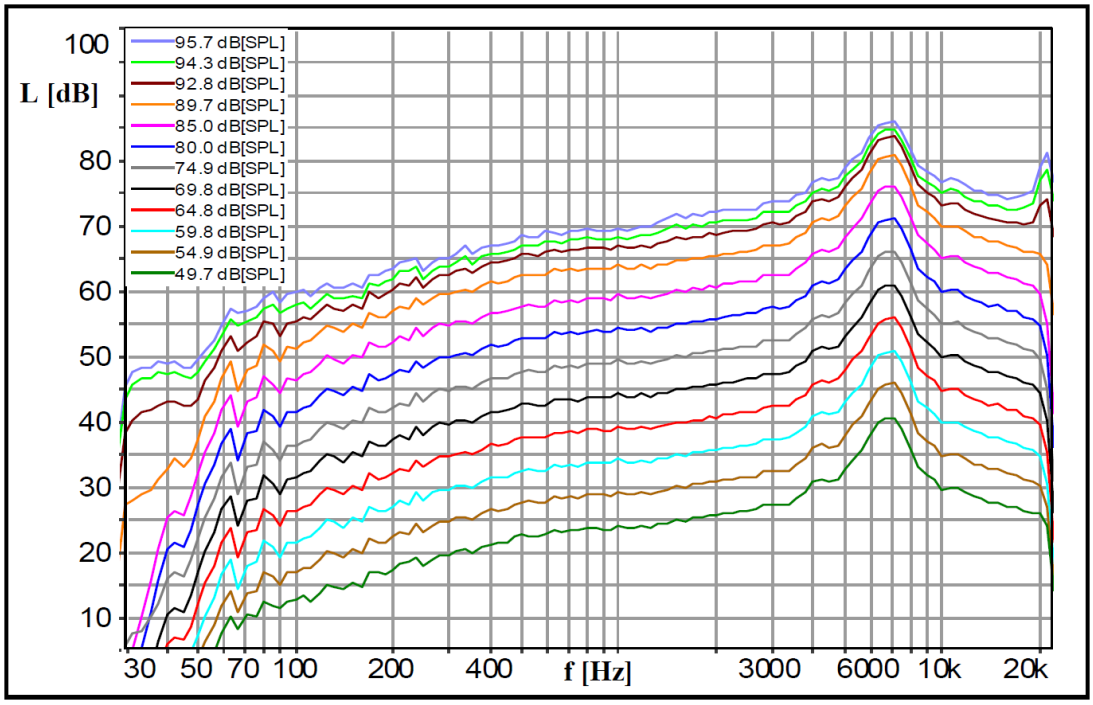


Bild 2.3: Frequenzgang des Smartphone A bei Pegeln von 50 dB bis 105 dB in 5-dB-Schritten, gemessen mit Rosa Rauschen. Kalibrierung bei 80 dB. Legende: Pegelangabe des Smartphone, nach [1]

der die Amplitude gespeichert wird. In der digitalen Welt ist es üblich Zahlen binär darzustellen. Bei einer Bitrate von 8 Bit werden beispielsweise 2^8 = 256 Werte in ganzen Zahlen abgespeichert. Somit definiert die Bitrate die Auflösung der Amplitude [26].

In Bild 2.4 ist der Zusammenhang von Abtastrate und Quantisierung veranschaulicht. Hier wird eine vollständige Sinus-Schwingung mit 60 Abtastungen auf ein Raster der Auflösung von 20 Stufen quantisiert.

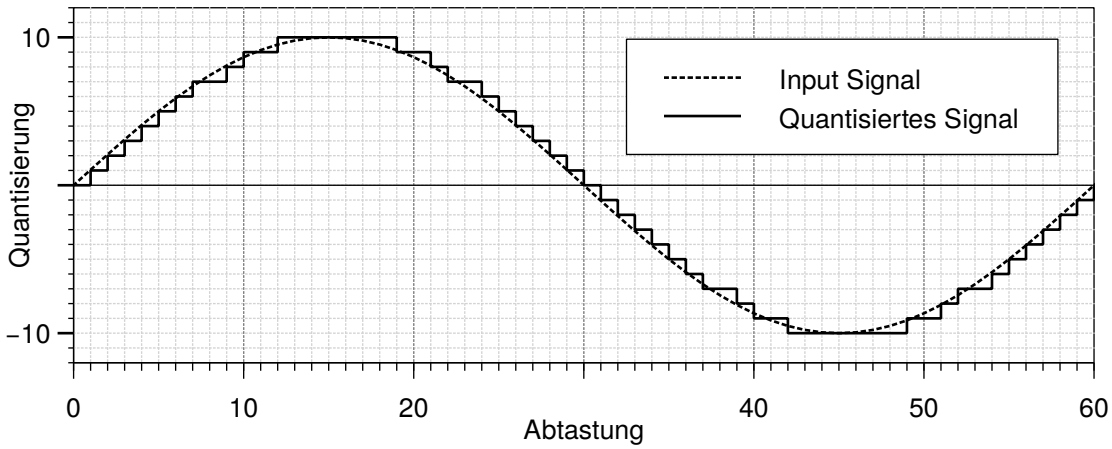


Bild 2.4: Abtastung und Quantisierung einer Sinus-Welle, nach [2]

2.4.2 Nyquist-Shannon-Abtasttheorem

Das Nyquist-Shannon-Abtasttheorem besagt, dass zur Übertragung einer Frequenz genau zwei Samples ausreichen. Beispielsweise ergibt sich zur Darstellung einer Frequenz von 22 kHz, eine Abtastrate von 44 kHz. Allerdings muss im Falle ungefilterter Audiosignale zusätzlich der Aliaseffekt beachtet werden [25].

2.4.3 Alias-Effekt

Ohne eine vorherige Tiefpassfilterung eines eingehenden Signals führt eine niedrige Abtastrate zu Störungen des übertragenen Signals. Da maximal eine Schwingung mit der halben Abtastfrequenz registriert werden kann, werden Frequenzen oberhalb der halben Abtastrate an ihr gespiegelt. Wird zum Beispiel mit 40 kHz abgetastet, so wird eine Schwingung von 30 kHz im Digitalsignal mit 10 kHz übertragen. Um dies zu verhindern, muss das Signal bereits vor der Abtastung bis zur halben Abtastrate tiefpassgefiltert werden [25].

2.4.4 Rauschen

Rauschen beschreibt ein Schallsignal, bei dem Amplitude und Frequenz stochastisch verteilt sind. Zum einen wird es in der Akustik häufig als Testsignal verwendet, zum anderen stellt es aber auch eine Störgröße akustischer Messungen dar. Dabei wird zwischen verschiedenen Arten von Rauschen unterschieden [14].

Ein Rauschen dessen spektrale Intensitätsdichte über den verwendeten Frequenzbereich konstant ist, wird **weißes Rauschen** (White Noise) genannt. Die spektrale Intensitätsdichte ist der Quotient aus Schallintensität und der Frequenzbandbreite. In der Akustik ist die verwendete Frequenzbandbreite meist der Bereich zwischen 20 und 20000 Hz. Das Rauschen setzt sich aus sehr vielen, sehr dicht nebeneinanderliegenden Sinusschwingungen zusammen, deren Amplituden gleich groß sind und deren Phasen statistisch gleichmäßig unabhängig von einander verteilt sind [19].

Daneben gibt es unterschiedliche Arten gefilterten Rauschens, bei denen sich die spektrale Intensitätsdichte abhängig von der Frequenz ändert. Beim $1/f$-Rauschen oder auch **rosa Rauschen** (Pink Noise), fällt die spektrale Intensitätsdichte pro Frequenzverdopplung um den Faktor 0,7 ab. Dies entspricht 3 dB/Oktave [19]. Rosa Rauschen findet häufig in akustischen und elektroakustischen Messungen als Testsignal Anwendung [14]. In Bild 2.5 ist die Energiedichte von Weißem und Rosa Rauschen im Frequenzspektrum dargestellt.

Vom Rauschen zu unterscheiden sind Geräusche, wie Schallereignisse genannt werden, die meist nicht zweckbestimmt sind und Anteile von Rauschen, Ton- und Klang-

gemischen enthalten [19].

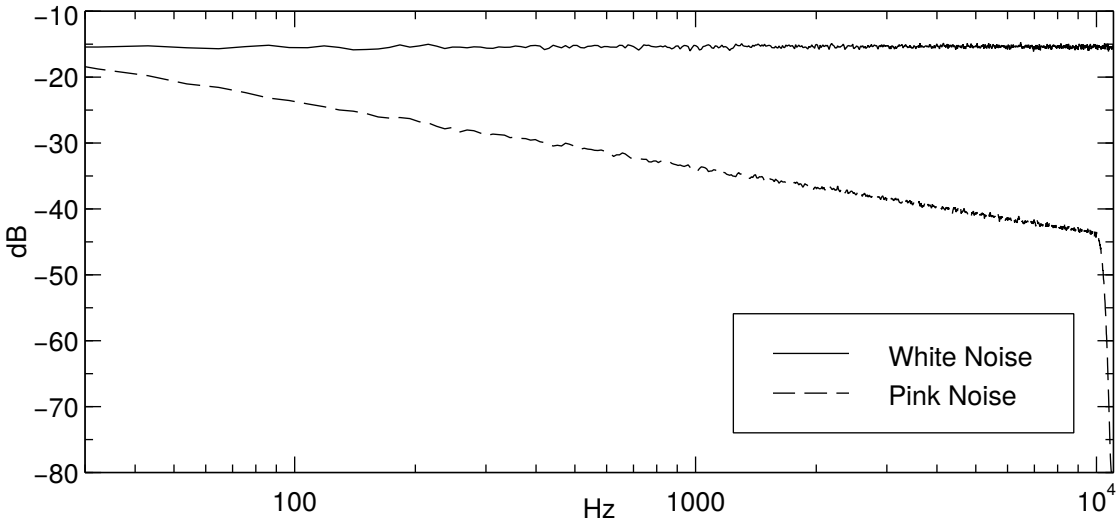


Bild 2.5: Energetisches Frequenzspektrum von Weißem und Rosa Rauschen

2.4.5 WAV-Dateiformat

Das Waveform Audio File Format (WAV) ist ein weit verbreitetes Format zur digitalen Speicherung von Audiosignalen. Es besteht aus einem vorangestellten *Header*, welcher Informationen über die Abtastrate, Bittiefe und die Anzahl der Audiokanäle enthält. Darauf folgen die eigentlichen unkomprimierten Audiodaten, die meist nach dem Pulse-Code-Modulation-Verfahren (PCM) gespeichert werden. Die in einer WAV-Datei gespeicherten Daten sind nicht komprimiert, sodass in ihr alle Samples und Bits der ursprünglichen Aufnahme enthalten sind. Sie kann ohne Dekodierung direkt ausgelesen werden. Folglich kann ohne Rechen- und Speicherverlust auf eine bestimmte Stelle der Daten zugegriffen werden. Nachteil dieses Formats ist der große Speicherbedarf [27].

Audiokomprimierung

Um diesem hohen Speicherbedarf zu begegnen, exisitieren diverse Verfahren zur Audiokomprimierung. Durch verlustfreie Redundanzreduktion und verlustbehaftete Irrelevanzreduktion kann die Datenrate des ursprünglichen Audiosignals stark minimiert werden. Die Irrelevanzreduktion basiert auf der Entfernung von für das menschliche Ohr nicht wahrnehmbaren Bestandteilen des Audiosignals [28].

2.5 Audioverarbeitung

Im Folgenden sollen die verwendeten Methoden zur Verarbeitung und Analyse digitalisierter Audiodaten erläutert werden.

2.5.1 Spektrum

In der Physik ist das Spektrum die Zerlegung eines Vorgangs in eine Summe harmonischer Schwingungen [29]. In der Akustik bezeichnet das Spektrum das Schallspektrum und ist die Darstellung eines Klanges in einem Frequenz-Amplituden-Diagramm. Man spricht auch von der spektralen Darstellung [30].

2.5.2 Fourier-Transformation

Ein bedeutendes Werkzeug zur Signalanalyse ist die (kontinuierliche) Fourier-Transformation [14]. Sie basiert auf der Fourier-Reihe. Die Reihenentwicklung nach Fourier stellt eine beliebige periodische Funktion als eine Linearkombination harmonischer Schwingungen (Sinus- und Cosinusfunktionen) dar. Die Fourier-Reihe kann ausschließlich auf periodische Signale angewandt werden [31].

Für den Fall aperiodischer Signale bedient man sich der kontinuierlichen Fourier-Transformation. Für ein zeitabhängiges Signal $s(t)$, welches über den $\mathbb{R}^n$ integrierbar ist und $T \rightarrow \infty$ lässt sich mit Gleichung 2.9 die Fourier-Transformierte S der Frequenz f berechnen [32].

$$S(f) = \int_{-\infty}^{\infty} s(t) \cdot e^{-j2\pi ft} dt \tag{2.9}$$

Das Ergebnis der Fourier-Transformation ist ein komplexes Spektrum, wobei der Realteil das Amplitudenspektrum und der Imaginärteil das Phasenspektrum ergibt. Folglich ist das Betragsspektrum einer Fourier-Transformation das Amplitudenspektrum [14].

Diskrete Fourier-Transformation

Das Ergebnis einer kontinuierlichen Fourier-Transformation diskreter Signale ist ein kontinuierliches, unendliches Frequenzspektrum $S(f)$. Zur Berechnung eines diskreten, endlichen Frequenzspektrums wurde die diskrete Fourier-Transformation (DFT) eingeführt [32]. Um die kontinuierliche Fourier-Transformation aus 2.9 zu diskretisieren, muss zum einen die Eingangsfunktion $s(t)$ diskretisiert und zum anderen die unendliche Folge zu beiden Seiten begrenzt werden.

Üblicherweise wird die Funktion zu äquidistanten Zeitpunkten abgetastet. Statt der Grenzen $[-\infty, \infty]$ werden die Grenzen $[0, T]$ eingesetzt. Das Intervall $[0, T]$ wird als Block bezeichnet. Die Abtastrate beträgt $f_S = 1/\Delta t$.

Die sich daraus ergebende Funktion wird N-mal im Intervall zu den Zeitpunkten $t = n\Delta t$ abgetastet. N ergibt sich aus dem Produkt von Periode und Abtatstrate $N = T/\Delta t$ und n nimmt die Werte $n = 0, 1, ...N - 1$ an. N wird auch als die Blocklänge oder **Blockgröße**

bezeichnet. Aus dem Integral in Gleichung 2.9 wird so das Produkt von $\frac{1}{N}$ und der Summe über $n = 0$ bis $N - 1$.

Die Transformierte $S(m)$ der Frequenz m berechnet sich dann mit dem Signal $s(n)$ zu den Zeitpunkten $n\Delta t$ mit Gleichung 2.10 [14].

$$S(m) = \frac{1}{N} \sum_{n=0}^{N-1} s(n) \cdot e^{-j2\pi mn/N} \qquad \text{für} \qquad 0 \leq m \leq N-1 \tag{2.10}$$

Wenn weder Informationen geopfert noch Redundanzen hinzugefügt werden sollen, lassen sich für N abgetaste Zeitpunkte genau N Spektralwerte berechnen [14].

Fast Fourier Transformation

Die DFT ist für die Signalverarbeitung bedeutungsvoll, da für ihre Berechnung unter anderem in Form der schnellen Fourier Transformation (FFT, engl. Fast Fourier Transformation) effiziente Algorithmen existieren [32].

Die Schnelle Fourier Transformation, engl. Fast-Fourier-Transformation (FFT) ist eine exakte Lösung der DFT, lässt sich aber nur auf Blöckgrößen von $N = 2^k, k \in \mathbb{N}$ anwenden. Durch vorsortieren der Summanden und Vermeidung überflüssiger Rechenschritte wird die Berechnung optimiert. So kann beispielsweise für eine Blockgröße von $N = 4096$ die Anzahl der Rechenoperationen von 16777216 auf 45056 gesenkt werden [26].

2.5.3 Fensterfunktionen

Falls die Blockgröße der DFT eines periodischen Signals nicht einem ganzzahligen Vielfachen der Periode entspricht, endet das Signal nicht nach einer ganzen Wellenlänge. Folglich bezieht sich das Spektrum auf ein Signal mit einer Unstetigkeit. Um die Unstetigkeit mittels harmonischer Schwingungen darstellen zu können, muss eine Vielzahl zusätzlicher Schwingungen addiert werden. Dies führt zu einer verfälschten Darstellung des Frequenzspektrums und wird als Leck-Effekt bezeichnet [26].

Für viele Anwendungen kann die Blockgröße nicht exakt an die Periodenlänge der Eingangssignale angepasst werden. Um den Leck-Effekt zu mindern, wurde die Fenstertechnik entwickelt. Ein Block bei der Diskretisierung kann auch als Fenster bezeichnet werden, da er nur einen Ausschnitt des kontinuierlichen Signals darstellt. Mathematisch kann ein Fenster durch die Multiplikation des Signals mit einer Fensterfunktion erzeugt werden. Die einfachste Form der Fensterfunktion ist das Rechteckfenster. Alle Eingangswerte werden für den betrachteten Zeitabschnitt mit 1 multipliziert und entsprechen dem unveränderten Eingangssignal. Die Werte außerhalb des Fensters wer-

den mit 0 multipliziert [33] .

Zur Milderung des Leck-Effekts wird, statt des Rechteck-Fensters, eine andere Funktionen verwendet. Zur Veranschaulichung wird in Bild 2.6 jeweils eine Sinuswelle mit dem Hann- und dem Hamming-Fenster multipliziert. Den verwendeten Fensterfunktionen ist gemein, dass sie einen bis zum Mittelpunkt des betrachteten Zeitraums ansteigenden und vom Mittelpunkt bis zum Ende des Zeitraums symmetrisch absteigenden Verlauf haben. Dem Signal wird dadurch eine Periodizität aufgeprägt. Anfang und Ende nähern sich dem Wert 0 an, wodurch eventuelle Unstetigkeiten der betrachteten Welle abgeschwächt werden. Abhängig von der verwendeten Fensterfunktion, bilden

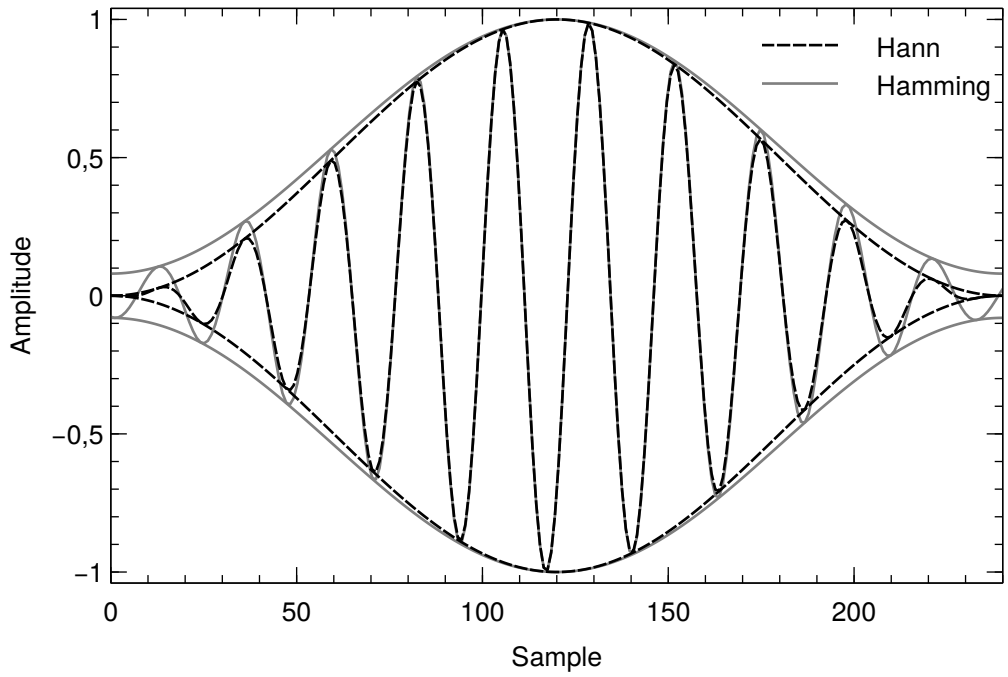


Bild 2.6: Vergleich Hann- und Hamming-Fenster auf einer Sinus-Schwingung, nach [3]

sich im Ergebnis der FFT unterschiedliche Leckagen aus. Für jede im ursprünglichen Signal enthaltene Frequenz ergibt sich im Frequenzgang der FFT ein sogenannter Hauptzipfel sowie mehrere Nebenzipfel. Der Hauptzipfel gibt den Frequenzbereich direkt um die untersuchte Frequenz wieder, während die Nebenzipfel weitere Bereiche zu beiden Seiten des Frequenzspektrums darstellen. Je nach Fensterfunktion variieren die Amplituden und die Breiten der Zipfel. Die Wahl einer Fensterfunktion ist deshalb stark vom Frequenzinhalt des betrachteten Signals abhängig. Die Einflussfaktoren sind der frequenzielle Abstand von Frequenzkomponenten zueinander und die benötigte Amplitudengenauigkeit einzelner Frequenzanteile. Niedrige Nebenzipfel reduzieren die spektrale Streuung. Ein schmaler Hauptzipfel bedeutet eine hohe Frequenzauflösung auf Kosten der Amplitudengenauigkeit. Ein breiterer Hauptzipfel bringt eine weniger

exakte Frequenzauflösung, dafür eine höhere Amplitudengenauigkeit mit sich [3].

Hann-Fenster

Eines der am häufigsten verwendeten und für ein breites Spektrum an Anwendungen geeignetes Fenster ist das Von-Hann-, Hann- oder auch Hanning-Fenster [3]. Dabei handelt es sich um eine Cosinusfunktion, die durch ihre Minima begrenzt ist. Die Minima sind auf den Wert 0 angehoben. Somit ergibt sich eine cosinusförmige Glocke mit den Werten 0 an den Rändern und dem Wert 1 in der Mitte.

Die DFT des Hann-Fensters hat einen relativ breiten Hauptzipfel mit deutlich abgesenkten Nebenzipfeln. Zusätzlich besitzen die Nebenzipfel eine relativ große Nebenzipfel-Roll-Off-Rate. Diese ist ein Maß für die Abnahme der Amplituden der Nebenzipfel mit zunehmendem Abstand zum Hauptzipfel [4]. In Bild 2.7 werden die Hann-Fensterfunktion und ihre DFT dargestellt. In dieser Darstellung werden der breite Hauptzipfel und die schmalen, abnehmenden Nebenzipfel deutlich. Die Funktion ist wie folgt definiert:

$$w(n) = 0.5 - 0.5\cos\left(\frac{2\pi n}{M-1}\right) \qquad \text{für} \qquad 0 \leq n \leq M-1 \tag{2.11}$$

mit w – Fensterfunktion
n – Sample
M – Anzahl der Sample pro Fenster

Kurzzeit-Fourier-Transformation

Die Fourier-Transformation transformiert das eingehende Signal in eine spektrale Darstellung ohne die Spektralanteile Zeitpunkten oder Zeitintervallen zuzuordnen. In der Analyse instationärer Prozesse ist aber besonders die Änderung der Spektralwerte über die Zeit von Interesse.

Die Kurzzeit-Fourier-Transformation (STFT, engl. Short-Time Fourier Transformation) berechnet die Fourier-Transformation für ein gefenstertes Signal. Das Eingangssignal wird mit einer Fensterfunktion multipliziert und von diesem die Fourier-Transformation berechnet. Durch die Zeitverschiebung des Signals werden sukzessive die Fourier-Transformationen der einzelnen Zeitschritte berechnet und es ergibt sich für das untersuchte Intervall eine Zeit-Frequenz-Darstellung des Signals. Für zeitdiskrete Signale erfolgt die STFT durch Berechnung der DFT, zum Beispiel mittels der FFT [34]. Die Anzahl der Samples, um die das diskrete Signal verschoben wird, wird Hop-Länge (Hop Length) bezeichnet [35].

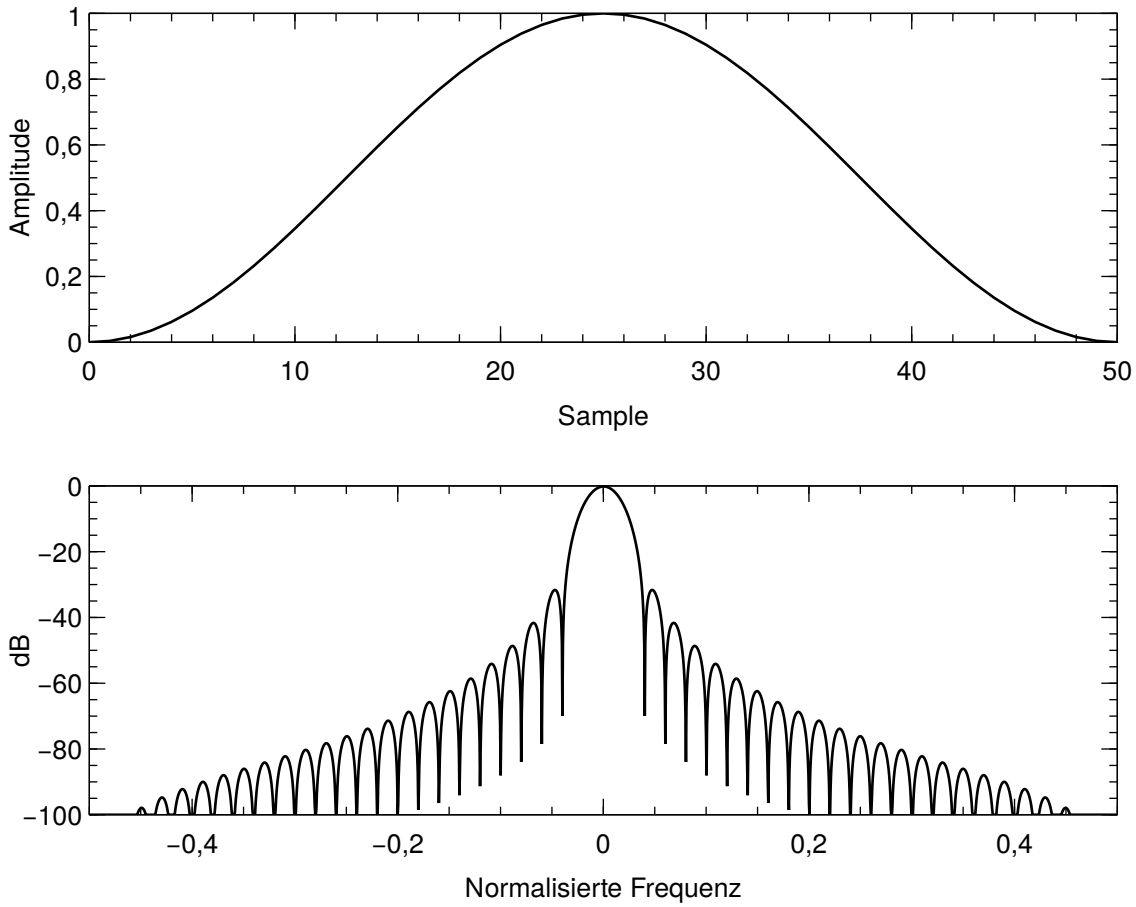


Bild 2.7: Hann-Fenster und die zugehörige DFT, nach [4]

2.5.4 Spektrogramm

Die Darstellung von Kurzzeit-Fourier-Transformationen aufgetragen über die Zeit wird als Spektrogramm bezeichnet. Dabei stellt die x-Achse die Zeit und die y-Achse die Frequenzen dar. Die aufgetragenen Werte berechnen sich aus dem Betragsquadrat der STFT und werden anhand einer Farbskala oder in Graustufen dargestellt [34]. In Bild 2.8 ist das Spektrogramm einer Signaltonaufnahme dargestellt, transformiert mit einer Blockgröße von $N = 1024$ und einer Abtastrate von $f_S = 48000$ Hz.

2.6 Neuronale Netze

Der Begriff Machine Learning wurde bereits 1959 von dem Forscher Arthur Samuel für einen Forschungsbereich verwendet, der Rechnern die Fähigkeit gibt zu „lernen“, ohne dies explizit zu programmieren [36] . Als (künstliches) Neuronale Netz (Artificial Neural Network) wird eine spezielle Art von Machine Learning-Algorithmen bezeichnet, die der Funktionsweise und dem Aufbau des menschlichen Hirns nachempfunden sind.

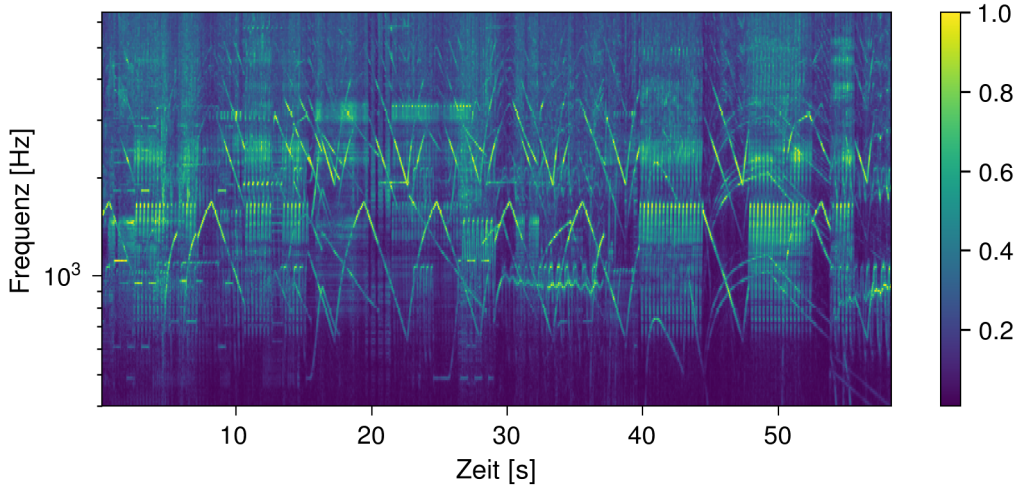


Bild 2.8: Beispiel Spektrogramm eines Signals

2.6.1 Architektur Klassischer Neuronaler Netze

Grundlegend bestehen künstliche Neuronale Netze (Artificial Neural Networks) aus mehreren, untereinander verbundenen Schichten von Knoten. Die Knoten entsprechen den Neuronen des menschlichen Hirns. Die Verbindungen zwischen den Knoten werden Kanten genannt und entsprechen den Synapsen.

Knoten

Ein Knoten oder Neuron (englisch: Node oder Neuron) eines Neuronalen Netzes (NN) hat einen oder mehrere gewichtete Eingangswerte, sowie einen sogenannten Bias b. Diese Werte werden im Knoten verarbeitet und ergeben einen Ausgangswert. Während eines Einlernvorgangs (siehe 2.6.2) eines NN werden Gewichte und Bias angepasst. Die Verarbeitung im Knoten wird als Übertragungsfunktion bezeichnet [36]. Mathematisch lässt sich die Übertragungsfunktion im Knoten klassischer Neuronaler Netze als Summe aller Produkte der Eingangswerte mit ihren jeweiligen Gewichtungen und des Bias betrachten. Im Anschluss durchläuft das Ergebnis der Übertragungsfunktion die Aktivierungsfunktion. Die gesamte Verarbeitung ist in Gleichung 2.12 definiert. In Bild 2.9 ist der Aufbau eines Knotens veranschaulicht.

$$y = \theta(\sum_{k}(w_k \cdot x_k) + b) \tag{2.12}$$

mit y – Ausgangswert
θ – Aktivierungsfunktion
x_k – k-ter Eingangswert
w_k – Gewicht der k-ten Kante
b – Bias des Knotens

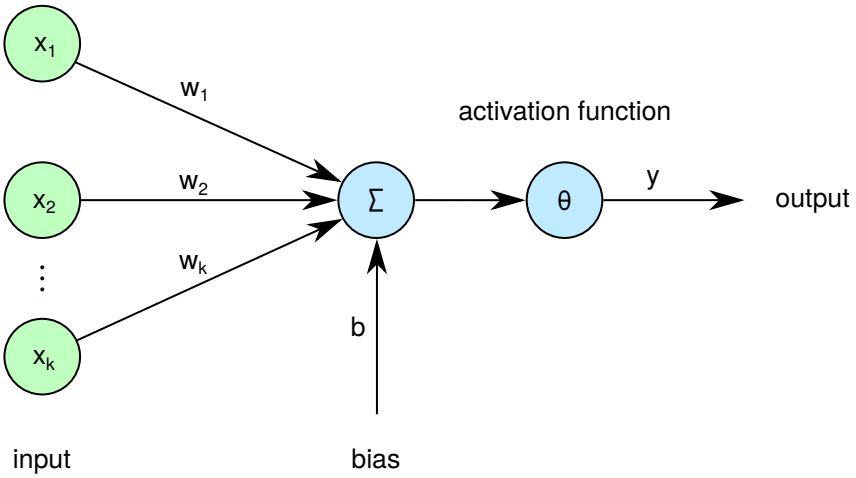


Bild 2.9: Prinzipskizze eines Neurons, nach [5]

Aktivierungsfunktionen

Die Aktivierungsfunktion θ (Activation Function) bestimmt basierend auf dem Ergebnis der Übertragungsfunktion, ob Informationen durch den Knoten weitergeleitet werden oder nicht. Nimmt der Ausgang des Knotens den Wert 0 an, wird der Knoten nicht „aktiviert“ und es werden keine Informationen an die nächste Schicht weitergegeben. Die einfachste Form einer Aktivierungsfunktion ist die Heaviside-Funktion (Gleichung 2.13).

$$y = \begin{cases} 1, & \text{if } \sum_k (w_k \cdot x_k) + b > 0 \\ 0, & \text{if } \sum_k (w_k \cdot x_k) + b < 0 \end{cases} \tag{2.13}$$

Der Schwellenwert der Aktivierungsfunktion wird durch den Bias verschoben. Diese Art der Aktivierung ist einfach, führt allerdings zu einer Diskontinuität in ihren Ausgangswerten, da diese genau 0 oder 1 betragen, ohne Werte dazwischen anzunehmen. Daher werden bevorzugt andere Aktivierungsfunktionen gewählt [5]. Die Sigmoidfunktion, die tanh-Funktion und die ReLU-Aktivierungsfunktion gehören zu den gängigen Aktivierungsfunktionen im Bereich Neuronaler Netze [37].

Die **Sigmoidfunktion** wird bereits seit den 1990er Jahren im Bereich künstlicher Neu-

ronaler Netze genutzt [38]. Eine der Standardformen der Sigmoidfunktion ist die logistische Funktion (siehe Gleichung 2.14) [36].

$$S(x) = \frac{1}{1 + e^{-x}} \tag{2.14}$$

Ihren Namen haben Sigmoidfunktionen durch ihren charakteristischen S-förmigen Verlauf. Durch ihre schnelle Sättigung in positive und negative Richtung, sowie den dazwischen annähernd linearen Verlauf eignen sie sich für Neuronale Netze. Meist wird eine Form der Funktion verwendet, die für große x-Werte gegen 1 und für kleine gegen 0 konvergiert [38].

Eine weitere, in LSTM-Netzen (Long Short-Term Memory) verwendete Aktivierungsfunktion ist die **tanh-Funktion** [39]. Auch diese weist, ähnlich der Sigmoidfunktion, einen S-förmigen Verlauf auf. Sie beinhaltet allerdings auch negative Werte, da sie gegen -1 und 1 konvergiert. In Bild 2.10 sind die Verläufe der Sigmoid- und tanh-Funktionen aufgetragen. Die Gleichung für die tanh-Funktion lautet:

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.15}$$

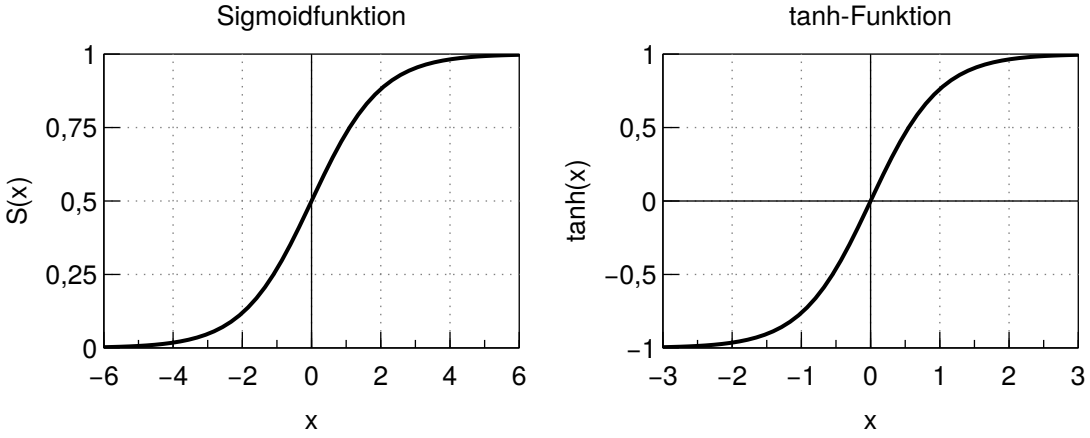


Bild 2.10: Sigmoidfunktion und tanh-Funktion

Sigmoid- und tanh-Funktion können durch ihre schnelle Sättigung zum sogenannten „Vanishing Gradient Problem“ führen, wodurch das Neuronale Netz unter Umständen nicht richtig lernt [37]. Die **Rectifier Linear Unit (ReLU)**, kurz auch Rectifier, begegnet diesem Problem [37]. Bei *Glorot et al.* [40] wurde 2011 nachgewiesen, dass Rectifier, trotz ihrer Nichtlinearität und der fehlenden Differenzierbarkeit in 0, zu besseren Ergebnissen im Training Neuronaler Netze führt. Der Rectifier kann mathematisch durch

Gleichung 2.16 ausgedrückt werden und ist in Bild 2.11 dargestellt [40].

$$rectifier(x) = max(0, x) \tag{2.16}$$

Die ReLU-Funktion führt zu Berechnungen mit dünn besetzten Daten (viele Eingängen $x_k = 0$), da im Gegensatz zur Sigmoidfunktion Werte für $x < 0$ genau gleich 0 gesetzt werden und nicht auf einen Wert gegen 0. Dies hat numerische Vorteile. So werden nicht immer alle Knoten aktiviert. Werden diese aktiviert, ist die verwendete Funktion linear. Folglich werden weniger und einfachere Rechnungen durchgeführt [40].

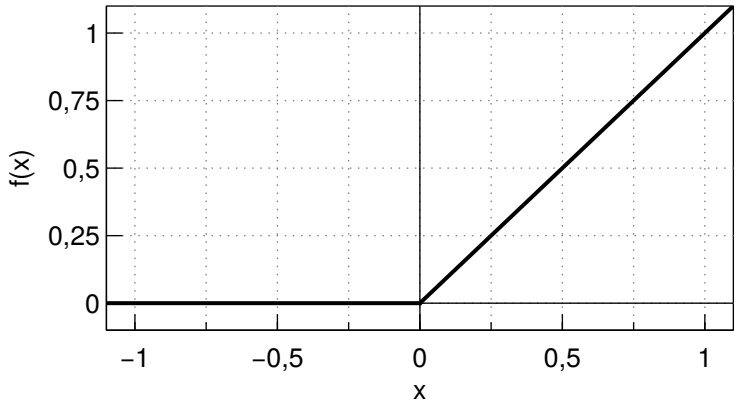


Bild 2.11: ReLU-Funktion

Kanten

Jede Kante eines Neuronalen Netzes verbindet einen Knoten mit einem anderen. In Rekurrenten Netzen kann ein Knoten auch mit sich selber verbunden sein (siehe 2.6.3). Der Ausgangswert y eines Knotens, multipliziert mit dem Gewicht der Kante w_k, ist der k-te Eingangswert x_k des verbundenen Knotens [5].

Schichten

Ein Neuronales Netz (NN) besteht aus einer Eingabeschicht (Input Layer) und einer Ausgabeschicht (Output Layer) sowie einer oder mehreren Schichten dazwischen. Die Eingabeschicht erhält Daten aus der Umgebung, beispielsweise eine Audioaufnahme oder ein Bild in Form eines Vektors oder einer Matrix. Die Ausgabeschicht gibt das Ergebnis des NN an die Umgebung ab. Die Schichten dazwischen kommen nicht mit der Umgebung in Kontakt, daher werden diese verdeckte Schichten (Hidden Layers) genannt [41]. Die verdeckten Schichten sind Spalten von Knoten. Jede Schicht ist mit der vorherigen und mit der folgenden Schicht über Kanten verbunden. Die Anzahl der Knoten kann in jeder Schicht variieren. Selbiges gilt für die Anzahl der Schichten sowie deren Art. Der beschriebene Aufbau wird in Bild 2.12 schematisch dargestellt. Eine optimale Schichtgröße und -anzahl ist nicht festlegbar und muss für jeden Anwendungs-

fall ermittelt werden [36].

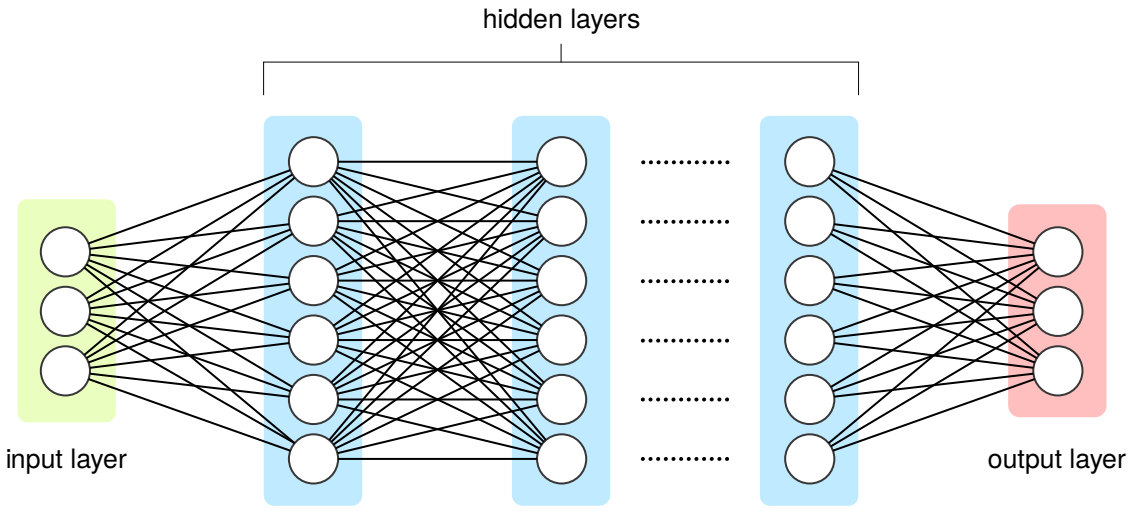


Bild 2.12: Schematischer Aufbau eines Neuronalen Netzes, nach [6]

Fully Connected Layer

Das Fully Connected Layer ist die ursprüngliche verdeckte „Standardschicht“ eines Neuronalen Netzes. Jeder einzelne Knoten dieser Schicht ist mit allen Knoten der vorherigen Schicht verbunden [6]. Die Knoten sind wie in Abschnitt 2.6.1 aufgebaut. Alternativer Name für das Fully Connected Layer sind „densely-connected neural network layer“, kurz Dense Layer [42] und Multi Layer Perceptron Layer (MLP).

Dropout

Eines der Hauptprobleme des Trainings Neuronaler Netze ist das sogenannte Overfitting (siehe 2.6.2). Um diesem Problem zu begegnen, wurde der Dropout entwickelt. Welcher Anteil der Knoten einer Schicht nicht berücksichtigt werden soll, wird durch den Dropout bestimmt. Diese Knoten werden für den aktuellen Einlernvorgang aus der Schicht entfernt. Die Gewichtungen der verbundenen Kanten und der Bias des Knotens werden durch das Entfernen in dieser Epoche nicht verändert. Die Auswahl der Knoten erfolgt zufällig. Der Dropout hat sich für verschiedenste Arten von Neuronalen Netzen und Trainingsdaten als effektives Mittel zur Milderung des Overfittings erwiesen [43].

Batch Normalization

Bei der Batch Normalization wird der Ausgang x einer Schicht durch einen Algorithmus auf Werte zwischen 0 und 1 normiert. Dies geschieht für jeden (Mini-)Batch (Paket an Daten). Die Batch Normalization kann auf jede Aktivierungsfunktion eines Netzes angewandt werden. Sie verhindert unter anderem ein „explodieren“ der Parameter (Exploding Gradiant Problem) einer Schicht. Sie führt zu einem stabilen Parameter-

anstieg und beschleunigt mit einer höheren Lernrate den Einlernprozess. Durch die regulierende Wirkung der Batch Normalization kann der Dropout verringert oder sogar weggelassen werden [44].

Für einen Batch B der Größe m werden zunächst der Mittelwert μ_B und die Varianz σ_B^2 berechnet:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{und} \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{2.17}$$

Ist x der Eingang des Netzes mit der Dimension d, werden alle Werte $x_i^{(k)}$ mit $k \in [1, d]$ und $i \in [1, m]$ wie folgt normiert:

$$\hat{x}_i^{(k)} = \frac{x_i^{(k)} - \mu_B^{(k)}}{\sqrt{\sigma_B^{(k)^2} + \epsilon}} \tag{2.18}$$

mit $\mu_B^{(k)}$ – Mittelwert der Dimension k

$\sigma_B^{(k)^2}$ – Varianz der Dimension k

ϵ ist eine Konstante, die für numerische Stabilität zur Varianz aufaddiert wird. Abschließend wird der normierte Wert $\hat{x}_i^{(k)}$ mit Gleichung 2.19 mit dem Standard Deviation Parameter $\gamma^{(k)}$ skaliert und durch den Mean Parameter $\beta^{(k)}$ verschoben. Dadurch bleiben die Verhältnisse und Informationen in den Daten erhalten.

$$y^{(k)} = \gamma^{(k)}\hat{x}_i^{(k)} + \beta^{(k)} \tag{2.19}$$

$\gamma^{(k)}$ und $\beta^{(k)}$ werden, wie die anderen Parameter des Netzes (Gewichtung, Bias, etc.), während des Trainings angepasst. Die Werte $y^{(k)}$ mit $k \in [1, d]$ sind die finalen Werte, die an die nächste Schicht des Netzes weitergegeben werden [44].

2.6.2 Training eines Neuronalen Netzes

Als Training wird der Prozess bezeichnet, bei dem die veränderlichen Parameter eines Neuronalen Netzes so lange angepasst werden, bis das Netz für die gewünschten Eingabedaten das gewünschte Ergebnis ausgibt. Die Netzarchitektur eines Neuronalen Netzes wird vor dem Training fest bestimmt und bis zum Abschließen des Trainings nicht verändert. Falls die Architektur verändert wird, muss das Netz neu trainiert werden. Vor allem Gewichtungen und Bias werden während des Trainings angepasst. Hinzukommen können Parameter, wie $\gamma^{(k)}$ oder $\beta^{(k)}$ der Batch Normalization. Anhand der Einlerndaten wird das Netz eingelernt und soll für bestimmte Merkmale der Daten einen bestimmten Ausgangswert erzeugen. Dabei werden die Daten normalerweise in

einen größeren Trainingsdatensatz und einen kleineren Testdatensatz aufgeteilt [5].

Loss

Um ein NN zu trainieren, werden Vorschriften benötigt, nach denen das Verhalten des Netzes bewertet werden kann. Die Abweichung des Ergebnisses $\hat{y}$ vom erwarteten Ergebnis y ist der Loss (Verlust). Eine weitverbreitete Art zur Berechnung des Loss ist die **mittlere quadratische Abweichung** (Gl. 2.20) für ein Ausgabeneuron und m geteste Datenpunkte.

$$MSE = \frac{1}{2m} \sum_{m} (y - \hat{y})^2 \tag{2.20}$$

Entsprechend kann für n Ausgabeneuronen die **summierte quadratische Abweichung** (Gl. 2.21) hergeleitet werden.

$$SSE = \frac{1}{2n} \sum_{n} (y_n - \hat{y}_n)^2 \tag{2.21}$$

Eine weitere weitverbreitete Loss-Funktion ist die sogenannte **Cross-Entropy** H (Gl. 2.22) [5].

$$H = -\sum_{n} y_n log \hat{y}_n \tag{2.22}$$

Zudem existieren auch Abwandlungen der Cross Entropy, wie die sogenannte Categorical Cross Entropy (Gl. 2.23). Diese wird bei vielklassigen Anwendungen verwendet, wenn jeder Eingang exakt einer Klasse zugeordnet werden kann [45].

$$H = -\frac{1}{N} \sum_{i=1}^{N} [y_i log \hat{y}_i + (1 - y_i) log(1 - \hat{y}_i)] \tag{2.23}$$

Backpropagation

Das Ziel des Trainings ist es den Loss zu minimieren. Ein niedriger Loss impliziert für einen bestimmten Eingangswert ein nur leicht abweichendes Ergebnis vom erwarteten Ergebnis. Die Minimierung geschieht iterativ. Die Trainingsdaten werden über mehrere **Epochen** eingelernt und die Parameter des Netzes angepasst, sodass der Loss minimal wird. Eine Epoche bezeichnet das einmalige Einlernen des gesamten Trainingsdatensatzes.

Das am weitesten verbreitete Verfahren zum Training ist die **Backpropagation**. Dazu werden mit dem berechneten Loss, ausgehend von der Ausgabeschicht über alle verdeckten Schichten bis zur Eingabeschicht, Netz-Parameter angepasst.

Im ersten Schritt der Backpropagation werden zufällige kleine Startwerte für alle Parameter initialisiert. Als nächstes werden dem Netz die Eingabedaten X_i übergeben und der Loss berechnet. Im letzten Schritt werden die Gewichte und Schwellenwerte mit folgenden Funktionen verändert:

$$W_{ij}^k(t+1) = W_{ij}^k(t) - \alpha \frac{\partial E}{\partial W_{ij}^k} \tag{2.24}$$

$$b_i^k(t+1) = b_i^k(t) - \alpha \frac{\partial E}{\partial b_i^k} \tag{2.25}$$

mit W_{ij}^k – Gewichte
b_i^k – Schwellenwerte
α – Lernrate
E – Loss

Die **Lernrate** α ist ein positiver, einstellbarer Wert. Abhängig von der Lernrate, konvergiert der Trainingsalgorithmus schneller, langsamer oder gar nicht. Wird α zu groß gewählt, kann dies zu Oszillation der Backpropagation führen. Wird die Lernrate zu niedrig gewählt, kann dies zu einer wesentlich langsameren Konvergenz führen. Mit den neu berechneten Parametern wird das Verfahren so lange wiederholt, bis eine zufriedenstellende Genauigkeit des Netzes erreicht wird oder das Netz nicht weiter lernt. Dies ist der Fall, wenn sich der Loss nicht mehr wesentlich ändert [36].

Kreuzvalidierung

Das Verfahren der Kreuzvalidierung wird verwendet, um das Einlernen eines Neuronalen Netzes zu bewerten. Neben den Trainingsdaten existiert ein Validierungsdatensatz, der das Netz während des Einlernens nicht durchlaufen hat. Für diesen Datensatz, wie für den Trainingsdatensatz, kann neben einer Vorhersagegenauigkeit, ein Loss berechnet werden. Dieser wird als Validation Loss bezeichnet und ist bei der Bewertung des Overfitting entscheidend [36].

Overfitting

Nimmt im Laufe des Trainings der Loss der Trainingsdaten ab und der Validation Loss zu, spricht man von Overfitting (Überanpassung). Das Netz lernt nicht mehr die gewünschten Merkmale der Daten, sondern die Daten in ihrer Gänze. Folglich kann es nicht mehr auf unbekannte Daten schließen. Beispielsweise wird bei einer Audiodatei nicht nur das zu erkennende Signal eingelernt, sondern jegliche Geräusche, die in der Datei vorkommen. Einlerndaten werden besser erkannt, nicht aber unbekannte

Validierungsdaten, die diese falschen Merkmale nicht enthalten [46].

Nach Bild 2.13 ist der optimale Zeitpunkt das Training zu unterbrechen, wenn der Validation Loss minimal ist, so dass weder Under- noch Overfitting stattfindet.

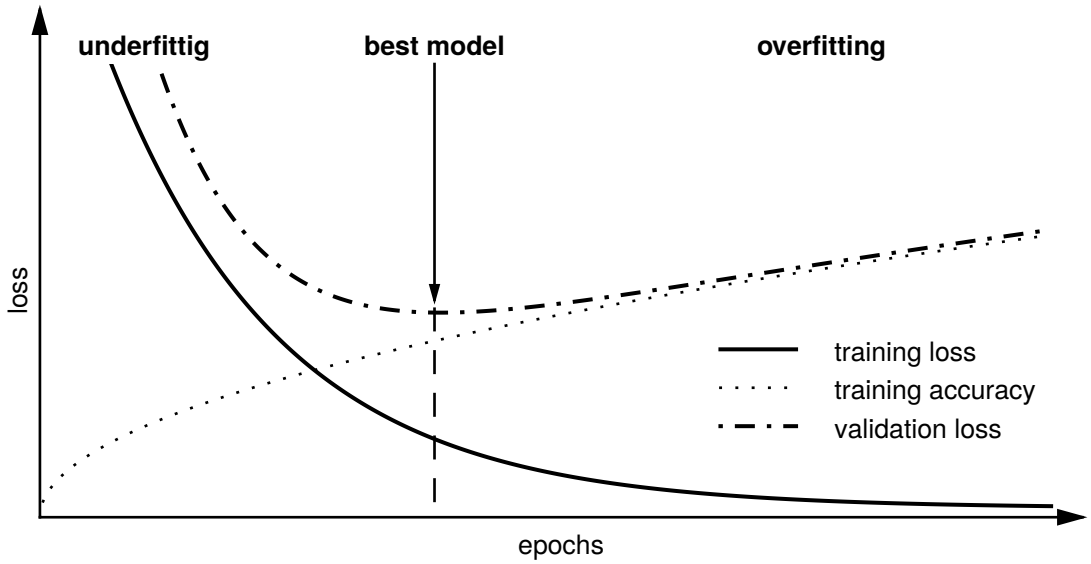


Bild 2.13: Schematische Darstellung der Beziehung zwischen Loss und Validation Loss, nach [7]

2.6.3 Weitere Netzarchitekturen

Neben den klassischen Neuronalen Netzen (Abschnitt 2.6.1) sind weitere Netzarchitekturen entwickelt worden. Sie unterscheiden sich durch den Aufbau der Knoten und deren Verknüpfungen. Im Folgenden werden in der Acoustic Event Detection (AED) gängige Architekturen vorgestellt.

Rekurrente Netze

Im Gegensatz zu klassischen, vorwärtsgerichteten Neuronalen Netzen (Feed-Forward Networks) besitzen Rekurrente Neuronale Netze (Recurrent Neural Networks) auch rückwärts- und seitwärtsgerichtete Kanten. Rückwärtsgerichtete Kanten sind Verknüpfungen mit Knoten einer vorherigen Schicht, seitwärtsgerichtete Kanten sind mit Knoten der gleichen Schicht. Zudem können Knoten mit sich selbst verknüpft werden. Diese Rückkopplungen ermöglichen, Informationen über einen Eingang hinaus zu speichern. Dadurch eignen sich Rekurrente Netze zur Erkennung von Sequenzmustern, beispielsweise in der Spracherkennung. Der Eingang wird in jedem Zeitschritt gespeichert, unabhängig von Eingabewerten. Ein neuer Eingang bekommt das gleiche Gewicht, wie alle vorherigen Eingänge zusammen. Dadurch verschwinden die Informationen länger zurückliegender Zeitpunkte mit der Zeit [47].

LSTM-Netz

Das LSTM-Netz ist eine Weiterentwicklung des RNN und wurde bereits 1995 entwickelt. LSTM steht für Long Short-Term Memory („langes Kurzzeitgedächtnis“). Erst mit Zunahme der Rechenleistung der letzten Jahre haben LSTM-Netze in der Breite Anwendung gefunden. Ein klassisches LSTM-Netz verfügt über ca. viermal mehr Parameter als ein einfaches RNN. Wie in anderen RNN können die Schichten eines LSTM-Netzes Informationen über einen Eingang hinaus speichern. Durch ihren Aufbau speichern LSTM-Netze aber nicht jeden Eingang gleichermaßen. Deshalb können auch länger zurückliegende Informationen im Zellzustand gespeichert bleiben.

Der Hauptunterschied einer LSTM-Schicht zu anderen Schichtypen ist der Aufbau der Knoten. Anstelle der Übertragungsfunktion wird die in Bild 2.14 schematisch dargestellte Zelle verwendet. S ist hier die Sigmoid-Aktivierungsfunktion und *tanh* die tanh-Funktion. Der Eingang der Zelle ist der Vektor $x(t)$, der die Ausgänge der Knoten der vorherigen Schicht enthält. Der Ausgang der Zelle ist $h(t)$. Zusätzliche Eingänge sind der Ausgang $h(t-1)$ sowie der Zellzustand $c(t-1)$ des vorherigen Zeitschritts. Der Zellzustand $c(t)$ wird auch als „memory cell“ bezeichnet, da er die von der Zelle gespeicherten Informationen enthält. Die Zelle ist in drei Gates (Schranken) geteilt: Forget Gate (Vergessensschranke), Input Gate (Eingangsschranke), Output Gate (Ausgangsschranke).

Das Forget Gate bestimmt welche Informationen des Zellzustands $c(t-1)$ erhalten bleiben. Das Input Gate bestimmt welche Informationen des Inputs $x(t)$ in den neuen Zellzustand $c(t)$ übernommen werden. Der neue Zellzustand $c(t)$ wird durch das Forget Gate und das Input Gate berechnet. Zur Berechnung des Ausgangs $h(t)$ wird $c(t)$ durch die tanh-Funktion gestaucht und im Output Gate bestimmt, welche Informationen des neuen Zellenzustands $c(t)$ für den Ausgang $h(t)$ verwendet werden [39].

Faltungsnetze

In klassischen und rekurrenten NNs ist ein Knoten jeweils mit jedem anderen Knoten der nächsten Schicht verbunden. Dadurch ist in diesen Netzen die Position jedes Eingangswerts zu jedem anderen Eingangswert gleichberechtigt. Zur Mustererkennung in der Bildverarbeitung ist die Position eines Pixels zum anderen von Bedeutung. In Faltungsnetzen (Convolutional Neural Networks, CNN) sind die rein vorwärtsgerichteten Schichten in kleineren Untergruppen (Kernels) miteinander verbunden. Lokale Strukturen werden dadurch stärker berücksichtigt [47]. Mathematisch wird dies durch diskrete Faltung erreicht.

Auf die Convolution Layer folgt nach dem Durchlaufen einer Aktivierungsfunktion in den meisten CNNs das sogenannte Pooling [7]. Dabei werden mehrere nebeneinan-

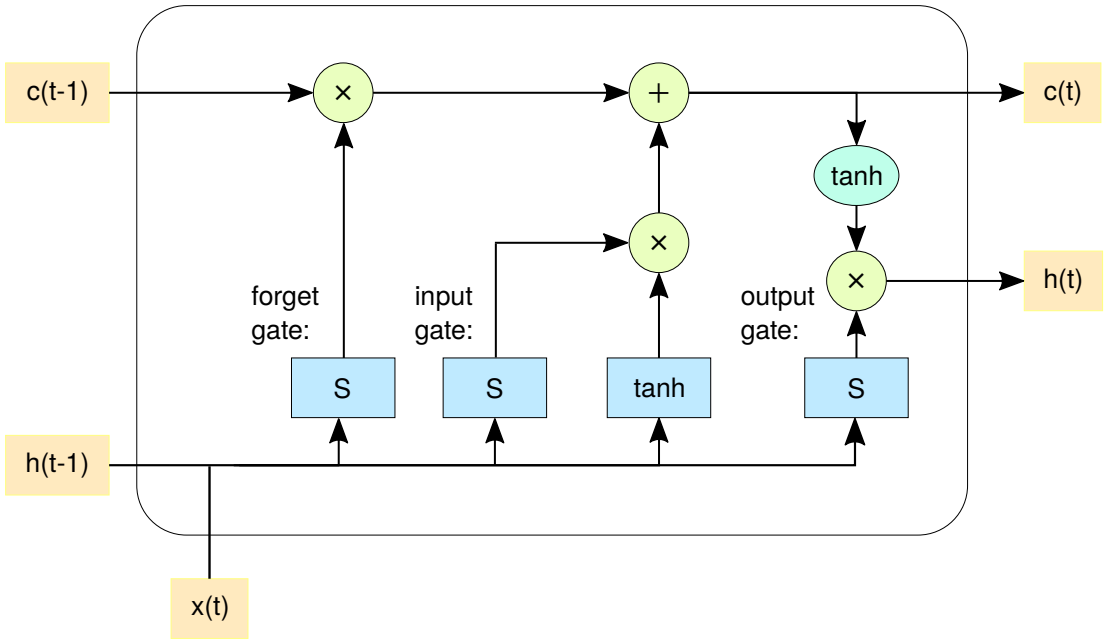


Bild 2.14: Schema einer LSTM-Zelle, nach [8]

derliegende Ausgänge der Convolution Layer zusammengefasst. Beim Max Pooling wird beispielsweise nur der maximale Ausgangswert eines Kernels der Konvolutionsmatrix weiterverwendet. Weitere Möglichkeiten sind unter anderem das Weitergeben eines Mittelwertes oder der euklidischen Norm des Kernels [7]. In Bild 2.15 ist ein Convolutional Neuronal Network (CNN) schematisch dargestellt.

Beim Verwenden mehrerer Convolution Layers werden in den ersten Schichten Zusammenhänge mit hohem Detailgrad erkannt. Mit Fortschreiten im Netz nimmt der Detailgrad kontinuierlich ab und es werden größere Zusammenhänge betrachtet [47].

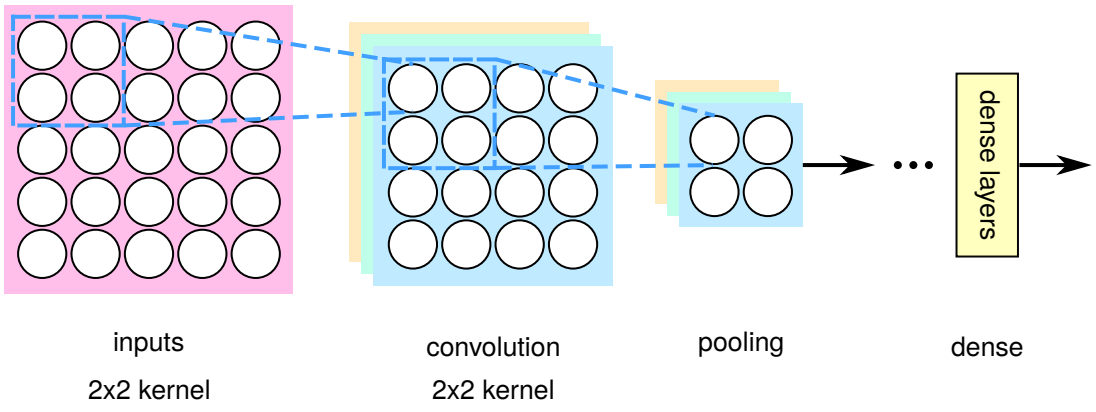


Bild 2.15: Schema eines Faltungsnetzes mit einer Konvolutionsschicht, nach [9]

Convolutional Recurrent Neural Networks

Als Convolutional Recurrent Networks (CRNNs) oder Recurrent Convolutional Networks (RCNN) werden Kombinationen aus Rekurrenten und Faltungsnetzen bezeichnet. Dadurch sollen die Vorteile der Faltungsnetze zur Erkennung lokaler Muster und die Vorteile Rekurrenter Netze zur Erkennung zeitlicher Muster kombiniert werden [48].

2.7 Signaltöne im Straßenverkehr

Bei der Erkennung von Signaltönen im Straßenverkehr konzentriert sich diese Arbeit auf deutsche und US-amerikanische Warnsignale, da die Datenlage für beide Land sehr gut ist. Die dort verwendeten Arten von Signaltönen werden so oder ähnlich auch in vielen anderen Ländern genutzt.

Die Signale beider Länder sind des Weiteren genormt und eignen sich dadurch zur Mustererkennung. Im Folgenden werden für beide Länder die Merkmale und die Energieverteilung der Signaltöne erläutert. In den Tabellen 2.2 und 2.3 sind diese zusammengefasst.

Merkmal	Werte			Einheit
Frequenzverhältnis	1:1,293	–	1:1,426	–
Frequenz tieferer Ton	360	–	487,2	Hz
Frequenz höherer Ton	465,5	–	630	Hz
Dauer Signalzyklus	1,25	–	1,75	s
Modulationsfrequenz	0,57	–	0,8	Hz
Mindestpegel Grundtöne (3.5m Entfernung)	110			dB(A)
Mindestpegel eines Obertons (1000 – 4000 Hz)	104			dB
Höchste Energiedichte der Signale	430	–	2500	Hz

Tabelle 2.2: Zusammenfassung der Merkmale deutscher Signaltöne

2.7.1 Normen und Merkmale akustischer Signaltöne in Deutschland

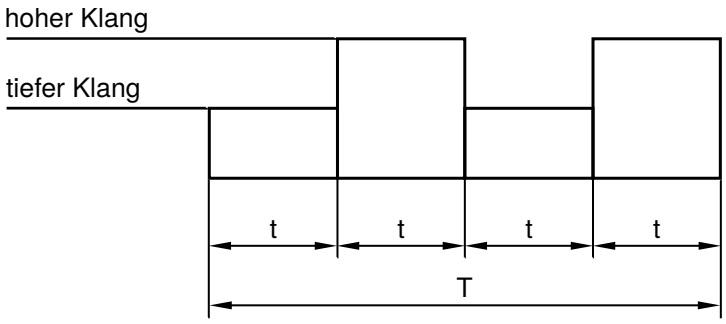
Die Umsetzung sogenannter „akustischer Warneinrichtungen für bevorrechtigte Wegebenutzer“ [10] ist in Deutschland in der DIN 14610 festgeschrieben. Die aktuelle

Merkmal	Wail			Yelp			Einheit
Untere Frequenzgrenze	500			500			Hz
Obere Frequenzgrenze	2000			2000			Hz
Frequenzverhältnis	$\geq$ 1:2			$\geq$ 1:2			–
Dauer Signalzyklus	2	–	6	0,24	–	0,4	s
Modulationsfrequenz	0,17	–	0,5	2,5	–	4,17	Hz
Höchste Energiedichte	500	–	1500	700	–	2000	Hz

Tabelle 2.3: Zusammenfassung der Merkmale US-amerikanischer Signaltöne

Fassung dieser Norm datiert von Januar 2009, wobei es im Vergleich zur vorherigen Fassung aus dem Jahr 1981 keine wesentlichen Änderungen in den Merkmalen der Signaltöne gibt.

Das deutsche Warnsignal ist eine Zweitonfolge. Es werden zwei Töne bzw. Klänge (siehe 2.2.1) abwechselnd wiederholt. In der Norm werden unter anderem das Verhältnis der Grundfrequenzen der beiden Klänge, der zulässige Frequenzbereich, die Modulationsfrequenz sowie Schallpegel und Schallspektrum vorgegeben [49].



t Dauer der einzenlen Klänge

T Gesamtablaufzeit

Bild 2.16: Klangfolge laut DIN 14610, [10]

Das Frequenzverhältnis der beiden Töne zueinander ist in der Norm mit 1:1,333 festgelegt. Gerundet entspricht es dem Intervall einer reinen Quarte (siehe Abschnitt 2.2.1). Toleriert wird zudem eine Abweichung vom Wert 1,333 zwischen 1,293 und 1,426. Das Frequenzverhältnis lässt sich im westlichen Tonsystem einordnen und entspricht in der Gleichstufigen Stimmung dem Verhältnis der fünften Stufe (siehe 2.2.3, $n = 5$). Der

zulässige Frequenzbereich der Grundfrequenzen liegt zwischen 360 Hz und 630 Hz. Folglich muss die Grundfrequenz des tieferen Klangs zwischen 360 Hz und 487,2 Hz liegen. Ein Signalzyklus wird in der Norm als zweimalige Wiederholung des Signals innerhalb von $3 \pm 0,5$ s festgelegt. Somit wiederholt sich eine Zweitonfolge alle 1,25 bis 1,75 s. Die Frequenz mit der sich das Signal wiederholt, die sogenannte Modulationsfrequenz, liegt zwischen 0,57 Hz und 0,8 Hz .

Der A-bewertete Schalldruckpegel muss in der Richtung der größten Schallabstrahlung in $3,5$ m Abstand im reflexionsfreien Raum für jeden der beiden Einzelklänge mindestens 110 dB betragen. Zudem muss für beide Töne mindestens ein harmonischer Oberton im Bereich von 1000 Hz bis 4000 Hz den unbewerteten Schallpegel von 104 dB überschreiten [10].

2.7.2 Spektrogramm und Energieverteilung akustischer Signaltöne in Deutschland

Norm DIN 14610 [10] schreibt im Hinblick auf die spektrale Zusammensetzung des Signals die beiden Grundtöne und je einen Oberton im Bereich zwischen 1000 und 4000 Hz vor. In Bild 2.17 ist die spektrale Energiedichte aller im Trainingsdatensatz verwendeten deutschen Signaltöne aufgetragen. Die Zusammensetzung der Trainingsdaten wird in Abschnitt 3.3 erläutert. Die erste Frequenzspitze des Signals findet sich bei ca. 430 Hz. Die Frequenzspitze der zweiten Grundfrequenz liegt bei 580 Hz. Damit sind die Vorgaben sowohl hinsichtlich der Frequenzen als auch bezüglich des Frequenzverhältnisses erfüllt. Unterhalb der Grundfrequenz des tieferen Tons ist die Energiefrequenz bis 50 Hz in etwa konstant und nimmt danach kontinuierlich ab. In diesem Bereich ist also kein Signal enthalten. Mit aufsteigender Frequenz sind die Obertöne deutlich sichtbar. Bis etwa 2500 Hz verfügen die einzelnen Töne über eine ähnlich hohe Energiedichte wie die Grundfrequenzen, mit einer etwa 5 dBFS höheren Frequenzspitze bei 1900 Hz. Ab ca. 2600 Hz nimmt die Energiedichte der Aufnahmen kontinuierlich ab. Im Spektrogramm einer Beispiel-Zweitonfolge (Bild 2.18) sind die alternierende Tonfolge und die Obertöne ebenfalls eindeutig zu erkennen.

2.7.3 Normen und Merkmale akustischer Signaltöne in den Vereinigten Staaten

Die Funktionsweise akustischer Signaltöne wird in den USA vor allem durch die Norm SAE J1849 „Emergency Vehicle Sirens“ geregelt. Es exisitiert für Krankenwägen die Norm GSA KKK–A–1822 „Ambulance, Emergency Medical Care Surface Vehicle“ und im Staat Kalifornien zusätzlich die Norm CCR Title 13 „Sirens“, auf welcher die SAE J1849 beruht. Die SAE J1849 wird außerdem von der National Fire Protection Asso-

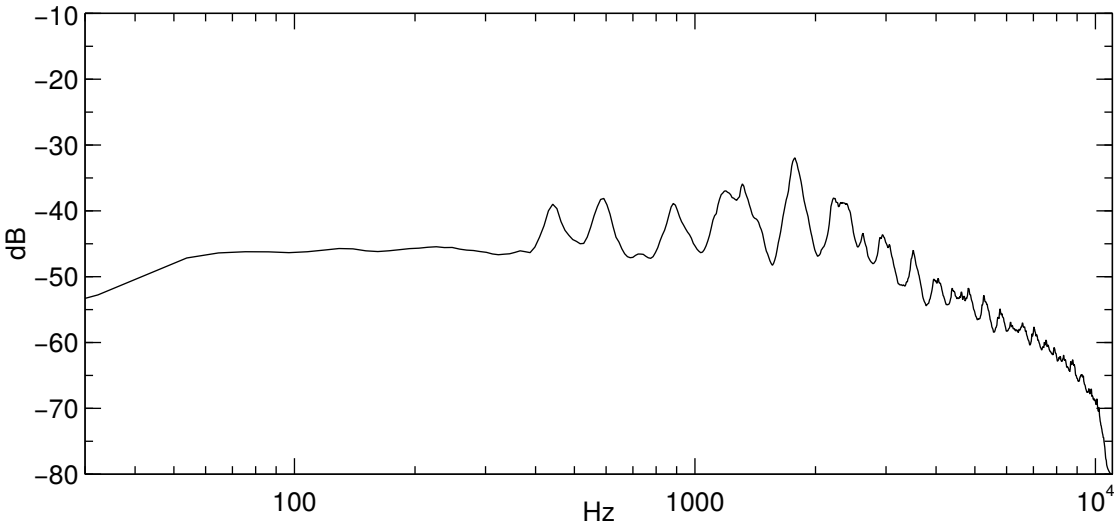


Bild 2.17: Spektrale Energiedichte Zweitonfolge

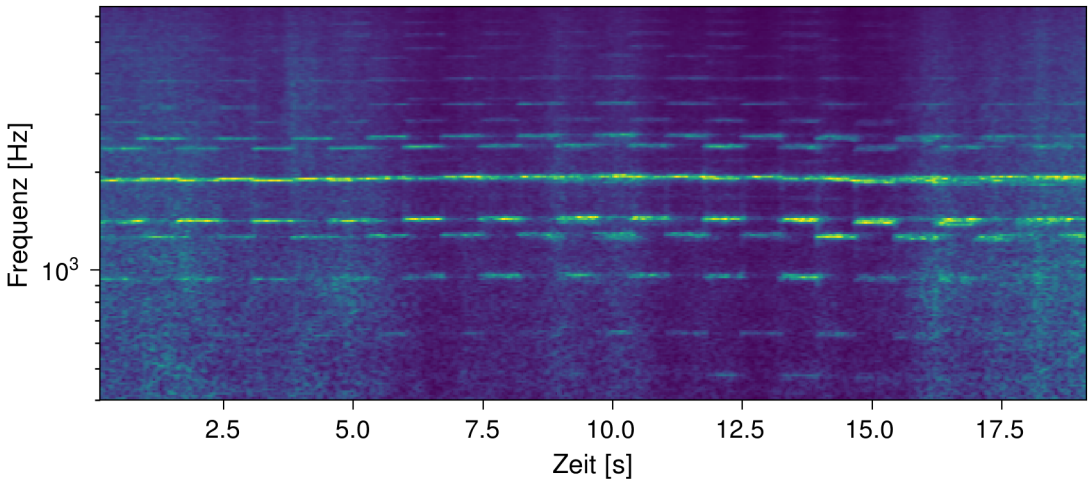


Bild 2.18: Spektrogramm Zweitonfolge

ciation anerkannt und ist somit auch für die US-Feuerwehr bindend [49].

In der SAE J1849 wird zwischen den zwei verschiedenen Signaltypen „Wail“ und „Yelp“ unterschieden. Diese unterscheiden sich lediglich in der Modulationsfrequenz. Während sich das längere Wailsignal alle 2 s bis 6 s wiederholt, sind dies beim Yelp lediglich 0,24 s bis 0,40 s. Folglich liegt die Modulationsfrequenz des Wail-Signals zwischen 0,17 Hz und 0,5 Hz und die des Yelp zwischen 2,5 Hz und 4,17 Hz.

Beide Signale sind durch einen in der Frequenz ansteigenden und anschließend wieder abfallenden Ton gekennzeichnet. Der zulässige Frequenzbereich liegt zwischen 500 Hz und 2000 Hz. Zwischen der niedrigsten und der höchsten Frequenz muss mindestens eine Oktave liegen. Folglich liegen die kleinstmöglichen Intervalle zwischen 500 Hz und 1000 Hz bzw. 1000 Hz und 2000 Hz. Des Weiteren muss der Anstieg bzw. Abstieg der Grundfrequenz stufenlos erfolgen [50].

2.7.4 Spektrogramm und spektrale Energiedichte akustischer Signaltöne in den Vereinigten Staaten

In Norm SAE J1849 [50] wird der exakte Klang eines Signaltons nicht spezifiziert. Die Norm enthält abgesehen von der Grundfrequenz keine Informationen über die spektrale Zusammensetzung des Signals. In Bild 2.19 ist die durchschnittliche spektrale Energiedichte aller in dieser Arbeit verwendeten Wail-Signale aufgetragen. Die Zusammensetzung der Trainingsdaten wird in Abschnitt 3.3 erläutert.

In dem Bild ist ein lokales Minimum bei ca. 500 Hz erkennbar. Die Energiedichten der Frequenzen im Bereich von ca. 60 bis 300 Hz und 600 bis 1500 Hz liegen auf einem ähnlichen Niveau. Oberhalb von 1500 Hz nimmt die Energiedichte deutlich ab. Aus Norm und Diagramm lässt sich herleiten, dass die höchste Energiedichte des Signaltons im Bereich von ca. 500 bis 1500 Hz liegt. Im Bereich von ca. 60 bis 300 Hz enthalten die Aufnahmen Störgeräusche mit einer ähnlichen Energiedichte wie der des Signaltons.

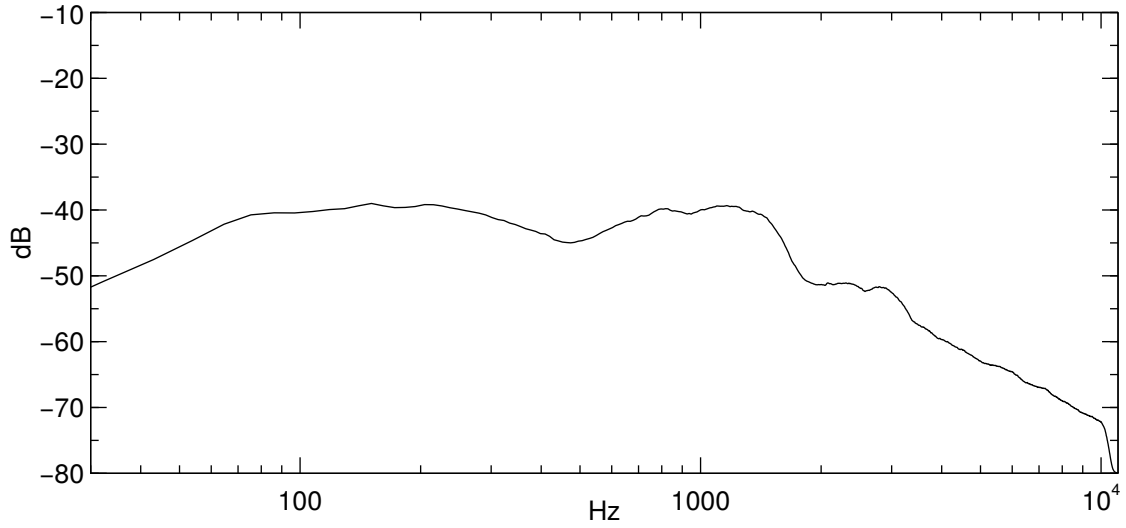


Bild 2.19: Spektrale Energiedichte Wail-Signal

Für die verwendeten Yelp-Signale ist die durchschnittliche spektrale Energiedichte in Bild 2.20 aufgetragen. Für diese Signale kann ebenfalls bei ca. 500 Hz ein lokales Minimum beobachtet werden. Im Bereich unterhalb von 400 Hz ist die Energiedichte ähnlich hoch wie beim Wail-Signal und hat abgesehen von einer Frequenzspitze bei 200 Hz einen ähnlichen Verlauf. Im Frequenzbereich von ca. 700 bis 2000 Hz erreicht die Energiedichte ein Maximum und liegt höher als beim Wail-Signal. Dieser Bereich liegt in dem von der Norm vorgeschriebenen Bereich von 500 bis 2000 Hz. Zusätzlich können zwischen 2800 und 5000 Hz weitere lokale Maxima beobachtet werden. Dieser Bereich entspricht etwa den vierfachen Frequenzen der maximalen Energiedichte.

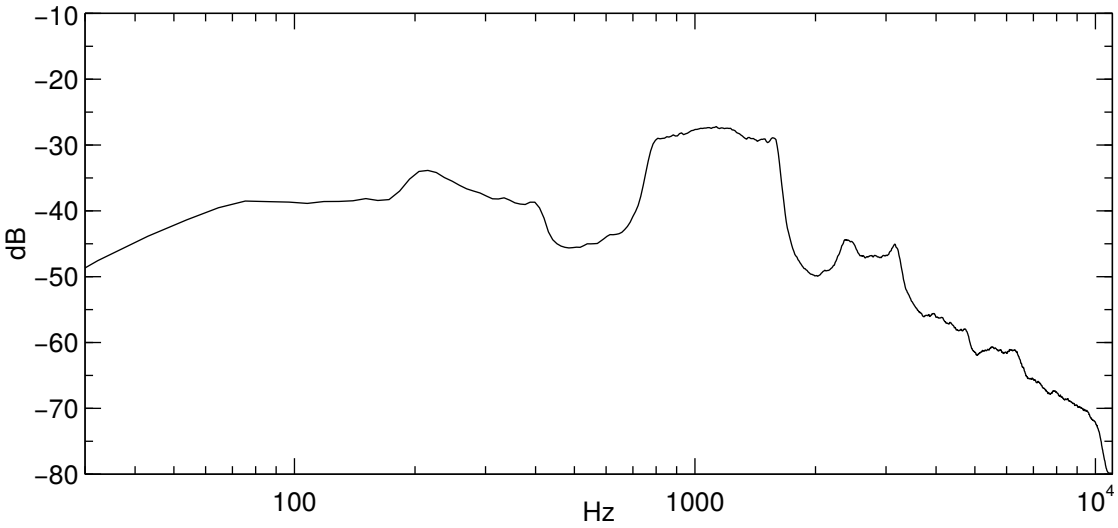


Bild 2.20: Spektrale Energiedichte Yelp-Signal

In den Spektrogrammen der Signale (Bilder 2.21 und 2.22) ist sichtbar, dass sich der Verlauf, abgesehen von der Modulationsfrequenz, gleicht. Wie von der Norm vorgeschrieben handelt es sich um einen kontinuierlichen An- und Abstieg um etwa eine Oktave. In beiden Aufnahmen sind mehrere Oktaven an Obertönen sichtbar.

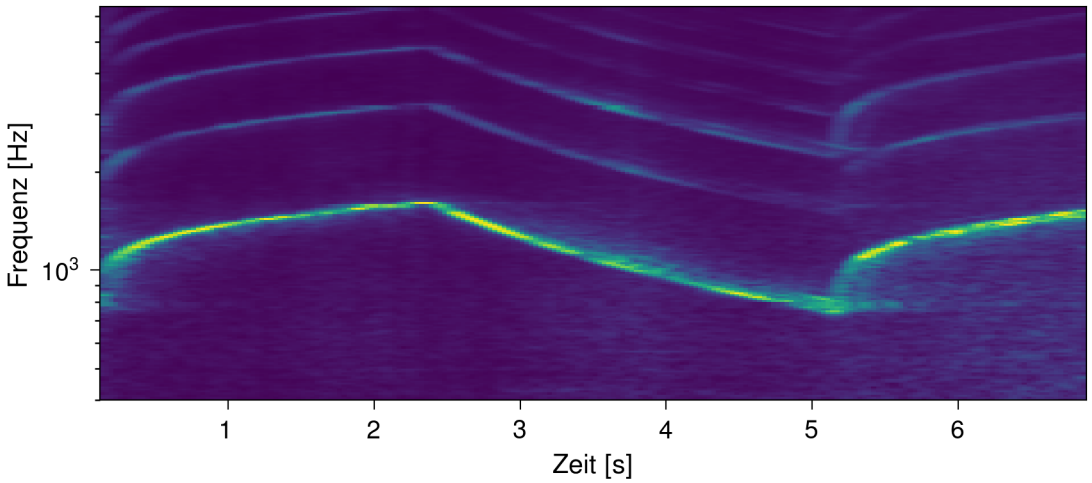


Bild 2.21: Spektrogramm Wail-Signal

2.8 Automatisierte Erkennung von Signaltönen

2.8.1 Klassifikation

Mit der fortlaufenden Digitalisierung unserer Gesellschaft, in der täglich immer größere Mengen an Daten anfallen, übernimmt die Datenanalyse eine Schlüsselrolle. Die Klassifikation ist ein Teilbereich der Datenanalyse und wird verwendet, um neue Da-

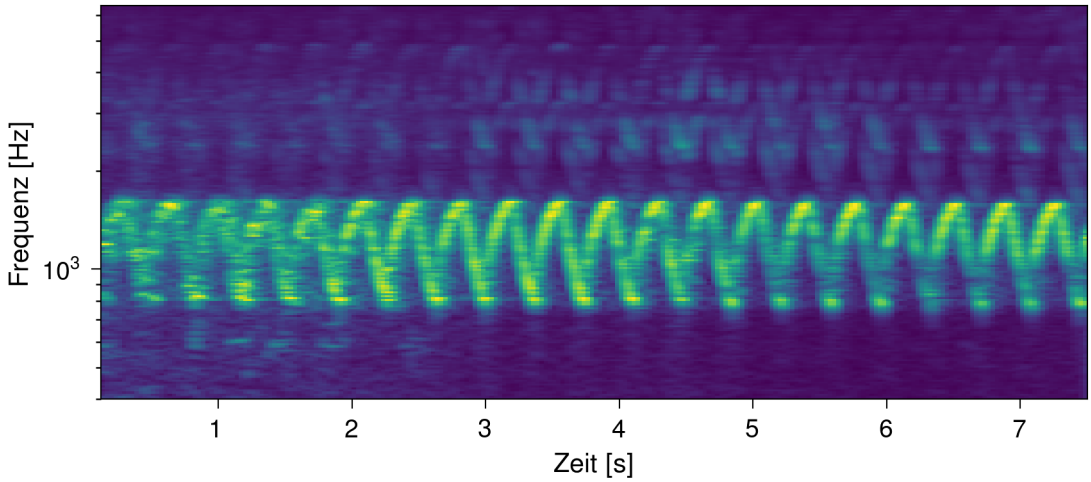


Bild 2.22: Spektrogramm Yelp-Signal

ten einer Gruppe oder Kategorie zuzuordnen. Insbesondere in der Mustererkennung bei Signal- und Bildverarbeitung besitzen Klassifikatoren einen großen Anwendungsbereich. Ein Klassifikationsverfahren hat die Aufgabe auf Basis eines klassifizierten Datensatzes, in dem jedes Objekt einer Klasse zugeordnet ist, neue Datenobjekte einer dieser Klassen zuzuordnen [47].

Bewertung von Klassifikatoren

Zur Bewertung eines Klassifikators mit der Klassifikationsfunktion f wird für einen Datensatz jedem Datenobjekt x ein Klassifikationswert y zugeordnet. Die berechneten Werte $f(x)$ für alle x des Datensatzes werden mit den jeweiligen y verglichen. Dazu werden für binäre Klassifikatoren folgende Grundkenngrößen verwendet:

- richtig positiv TP: $y = i$, $f(x) = i$
- richtig negativ TN: $y \neq i$, $f(x) \neq i$
- falsch positiv FP: $y \neq i$, $f(x) = i$
- falsch negativ FN: $y = i$, $f(x) \neq i$

Ist ein betrachteter Wert positiv und wird vom Klassifikator als solcher erkannt, so ist das Ergebnis richtig positiv (true positive). Wird es als negativ erkannt, ist es falsch negativ (false negative). Ebenso ist ein betrachteter Wert negativ und wird als negativ erkannt, ist das Ergebnis richtig negativ (true negative) und wird er als positiv erkannt ist es falsch positiv (false positive). Aus diesen Kenngrößen lassen sich nun weitere Kenngrößen herleiten. Die Relevanz $R = TP + FN$ gibt die Anzahl der tatsächlich richtigen Werte im Datensatz an, die Irrelevanz $I = FP + TN$ die Anzahl der tatsächlich falschen Werte. Auf Basis dieser Werte kann wiederum die Richtig-Positiv-Rate $TPR = TP/R$ (true positive rate) berechnet werden, die den Anteil der erkannten positiven Werte angibt, sowie die Falsch-Positiv-Rate $FPR = FP/I$ (false positive rate), die

den Anteil der nicht erkannten negativen Werte angibt [47].

2.8.2 Automatisierte Erkennung von Signaltönen im Straßenverkehr

Erste Ansätze zur automatisierten Erkennung von Signaltönen im Straßenverkehr finden sich bereits 2001 bei *Fragoulis* und *Avaritsiotis* [51] sowie bei *Ellis* [52]. Während *Fragoulis* und *Avaritsiotis* noch auf die Aktivierung mechanischer Filter auf Basis einer zuvor erfolgten manuellen Musterbestimmung setzen, unternimmt *Ellis* bereits den Versuch der Signaltonerkennung mittels eines Neuronalen Netzes aus der Spracherkennung. *Ellis* erkennt zwar das Potential des Ansatzes, kommt aber zu hohen Fehlerraten in seinen Ergebnissen. Diese führt er vor allem auf die nicht ausreichende Trainingsdatenmenge zurück. Speziell bezieht er sich auf die zu geringe Menge von negativen Daten (Geräusche). In den folgenden Jahren folgen weitere Arbeiten, beispielsweise unter der Verwendung von Hidden Markov Models oder bei *Schröder et al.* 2013 [53] mittels Part-based Models. In diesen Arbeiten können bereits klare Signaltöne erkannt werden, allerdings haben die Modelle Schwierigkeiten bei anderen Signal-To-Noise-Ratios als 0 dB.

Erfolgreiche Ansätze zur Erkennung von Signaltönen im Straßenverkehr mittels klassischer Neuronaler Netze und Rekurrenter Netz lassen sich erst ab 2018 bei *Tran et al.* [11] finden. Des Weiteren bietet *Xia et al.* [12] einen guten Überblick über die bis ins Jahr 2018 erfolgte Verwendung von „Deep Learning“-Algorithmen zur Acoustic Event Detection (AED).

Bei *Tran et al.* [11] wird versucht ein Neuronales Netz zu entwickeln, das die Signaltöne von taiwanesischen Feuerwehr- und Krankenwagen erkennt. Während das Feuerwehrsignal dem amerikanischen Wail-Signal ähnelt, handelt es sich bei dem Krankenwagensignal um eine Zweitonfolge, ähnlich den deutschen Signaltönen. Es werden dazu zwei verschiedene Modelle verglichen. Zum einen ein Multi Layer Perceptron (MLP), ein klassisches Neuronales Netz mit verdeckten Schichten, zum anderen ein rekurrentes Long Short-Term Memory (LSTM)-Netz. Als Eingang werden jeweils Mel-Frequency Coefficients (MFCC) verwendet, die im Allgemeinen für die Spracherkennung verwendet werden [54], [55]. Sie wurden aber auch schon erfolgreich zur Signalerkennung verwendet [56], [53]. Die beiden Netze werden nach dem klassischen Verfahren für Neuronale Netze implementiert, indem die Modelle zunächst mit einem Trainingsdatensatz trainiert und einem Validierungsdatensatz validiert werden. Bei zufriedenstellenden Ergebnissen wird das jeweilige Netz abschließend mit einem Testdatensatz auf seine Genauigkeit überprüft.

Schlussendlich zeigt die Arbeit, dass beide Netze zur Erkennung von Signaltönen ge-

eignet sind. Allerdings detektiert das LSTM-Netz besser und die Berechnungen sind um das 1, 38-fache schneller. Die in der Arbeit errechnete Genauigkeit für das LSTM wird mit 93, 84% im Gegensatz zu 75, 87% beim MLP angegeben. Des Weiteren verfügt das LSTM-Netz über 152, 8-mal weniger Parameter, was zur schnelleren Berechnung führt. Tabelle 2.4 listet den Umfang des verwendeten Datensatzes auf.

Data description	(SNR/SNR vehicle type)	Data amount in seconds
Training set	(fire truck/ambulance/road noise)	750/750/750
Validation set	(0dB/-12dB Ambulance)	1782/1768
Validation set	(-5dB/5dB Ambulance)	1759/1774
Validation set	(-10dB/10dB Ambulance)	1793/1792
Validation set	(-15dB/15dB Ambulance)	1783/1758
Validation set	(0dB/-12dB Fire truck)	1772/1774
Validation set	(-5dB/5dB Fire truck)	1763/1777
Validation set	(-10dB/10dB Fire truck)	1779/1793
Validation set	(-15dB/15dB Fire truck)	1780/1792
Validation set	(road noise)	3449
Testing set	(ambulance/fire truck/road noise)	319/325/330

Tabelle 2.4: Datensatz verwendet von *Tran et al.* [11, S.11]

Betrachtet man die von *Xia et al.* [12] zusammengetragenen Ergebnisse zur AED allgemein, so finden ähnliche Ansätze wie bei *Tran* [11] Anwendung. Das häufigste Verfahren zur Datenvorverarbeitung ist die MFCC. Die am häufigsten verwendeten Netztypen sind klassische „Deep Neural Networks“, Recurrent Neural Network (RNN)s, CNNs und CRNNs.

Die Autoren kommen zu dem Schluss, dass all diese Modelle zu guten Ergebnissen in der AED führen und andere maschinelle Lernverfahren an Leistung deutlich übertreffen. Herausforderungen sehen sie vor allem in begrenzten Trainingsdaten und der Anpassung der Netzparameter. In Bild 2.5 werden typische Strukturen der in der AED verwendeten Netze aufgelistet. In der AED werden Netze mit wesentlich weniger verdeckten Schichten verwendet als beispielsweise in der Bilderkennung. Ob zusätzliche Schichten vorteilhaft für die AED sind, wurde bei *Wang* und *Li* [57] untersucht. Die Autoren kommen zu dem Ergebnis, dass mehr verdeckte Schichten nicht zwangsläufig zu einer besseren Erkennung führen. Entscheidend ist die gesamte Anzahl anpassbarer Parameter. So kann ein MLP, mit nur einer verdeckten Schicht aber vielen Parametern, zu besseren Ergebnissen führen als ein vierlagiges MLP mit der gleichen Anzahl an Parametern. Somit ist bei begrenzter Rechenleistung ein Netz mit weniger Schichten einem Netz mit vielen vorzuziehen.

Reference	Neuronal network structure	Size	Error rate
Gorin et. al.	Convolution layer	80 filters (6 x 60)	0,97
	Convolutional layer	80 filters (1 x 3)	
	Feed-forward layer	1024 hidden units	
	Feed-forward layer	1024 hidden units	
Heittola et al.	Feed-forward layer	50 hidden units	0,94
	Feed-forward layer	50 hidden units	
Adavanne et al.	Recurrent layer (LSTM)	32 hidden units	0,8
	Recurrent layer (LSTM)	32 hidden units	
Meyer et al.	6 convolution layers	64 filters (1 x 3)	0,81
Adavanne et al.	3 convolution layers	128 filters (3 x 3)	0,79
	2 recurrent layers	32 hidden units	

Tabelle 2.5: Typische Netzstrukturen und Fehlerrate nach *Xia et al.* [12]

3 Durchführung

3.1 Vorgehen

Zur Entwicklung des Ereignisdetektors wird in dieser Arbeit im ersten Schritt aus Aufnahmen von Signaltönen, ähnlichen Signalen und Straßengeräuschen ein Trainingsdatensatz erstellt. Dies beinhaltet unter anderem die manuelle Selektion und Annotation der Daten. Zur genaueren Untersuchung der Merkmale deutscher Signaltöne wird eine Messung von Signaltönen an einer Hauptstraße durchgeführt und ausgewertet.

Anschließend wird basierend auf den Merkmalen der Signaltöne und dem erstellten Datensatz ein Programm zur Vorverarbeitung der Daten entwickelt. In dieser Vorverarbeitung werden die Eingangsdaten für das Neuronale Netz in ein verarbeitungsfähiges Format transformiert und Merkmale der Signaltöne herausgestellt.

Basierend auf einer Literaturrecherche wird iterativ ein Neuronales Netz zur Erkennung der vorverarbeiteten Signaltöne entwickelt. Zur Verwaltung des Datensatzes und zur Dokumentation der Ergebnisse des Trainings wird eine Datenbank aufgebaut. Nach Fertigstellung von Ereignisdetektor und Trainingsdatensatz, wird der Detektor mit dem Trainingsdatensatz trainiert und anhand eines unabhängigen Testdatensatzes getestet. Anhand der Detektion der Testdaten wird das entwickelte Neuronale Netz optimiert und final bewertet.

3.2 Verwendete Tools zur Programmierung

Bei der Programmierung des Ereignisdetektors werden ausschließlich Open-Source-Bibliotheken und -Tools verwendet. Dies ermöglicht die kostenlose Nutzung und Modifizierung der Software innerhalb der lizenzrechtlichen Bedingungen. Im Folgenden wird das verwendete Framework kurz vorgestellt. Die Auflistung der einzelnen Bibliotheken und Versionen ist im Appendix in Tabelle 6.2 zusammengefasst.

3.2.1 Python

Als Programmiersprache wurde aus mehreren Gründen Python gewählt. Python ist eine schnell zu erlernende und leserliche Programmiersprache. Für Python existiert eine Vielzahl frei zugänglicher Open-Source-Bibliotheken. So ist es möglich, ohne die Erfordernisse einer tiefgehenden Programmierung durch das Verbinden von Bibliotheken

Programme zu erstellen. Sie ist in der Forschung weit verbreitet und wird an Universitäten gelehrt.

Tensorflow ist ein Python-Framework zur Programmierung künstlicher Neuronaler Netze. Es verfügt über viele NutzerInnen und frei zugängliche Tutorials, die das Erlernen und Verwenden des Frameworks vereinfachen. Es hat sich unter anderem durch die erfolgreiche Nutzung in vielen Google-Anwendungen, wie der Spracherkennung und Google Maps, bewiesen. Mit Hilfe der high-level Programmbibliothek Keras ist es möglich, etablierte Algorithmen im Bereich Neuronaler Netze zu nutzen, ohne diese explizit programmieren zu müssen. Ein weiterer Bestandteil von Tensorflow ist Tensorflow Lite, welches das Ausführen von Modellen auf mobilen Endgeräten ermöglicht.

NumPy ist eine Open-Source-Alternative zu der kommerziellen Software `Matlab`, die in der Industrie und in der Forschung zur Lösung mathematischer Probleme weit verbreitet ist. Zusammen mit der Bibliothek `Matplotlib` zur mathematischen Darstellung ist die Numpy-Bibliothek ein vollwertiger Ersatz für Matlab. Numpy bietet eine ausführliche Dokumentation, viele frei zugängliche Tutorials und Lösungen. Die Bibliothek ermöglicht die Handhabung großer, mehrdimensionaler Arrays und gängige numerische Berechnungen mit Hilfe effizienter Algorithmen.

Des Weiteren wurde eine Vielzahl nativer Python-Bibliotheken verwendet sowie die externen Bibliotheken `pyAudio` und `soundfile` zum Importieren von Audiodateien, `librosa` und `SciPy` zur Audioverarbeitung, `ray` für **Multi-Core-Processing** und `H5py` zum Speichern und Laden von HDF5-Dateien.

3.2.2 SQLite-Datenbank

Um die verwendeten Daten systematisch zu verwalten und Ergebnisse zu dokumentieren, wird eine Datenbank verwendet. In einer lokalen Umgebung bietet sich dafür die Programmbibliothek `SQLite` an. Im Gegensatz zu SQL- oder MySQL-Datenbanken ist kein Server notwendig. Die Datenbank wird lokal gespeichert und verwendet. Über die Schnittstelle der Python-Bibliothek `sqlite3` kann die Datenbank automatisiert ausgelesen, befüllt und dynamisch erweitert werden.

3.2.3 HDF-Speicherformat

Das Hierarchical Data Format, kurz HDF, ist ein standardisiertes Dateiformat zur Datenspeicherung. Das Format ermöglicht schnelles Speichern und Auslesen großer Datenmengen. Unterschiedliche Datentypen können innerhalb einer Datei gespeichert, gruppiert und in Beziehung zueinander gesetzt werden. Das HDF-Format wird von der HDF Group entwickelt. Diese stellt ein unabhängige Bibliothek zur Nutzung in mehreren Programmiersprachen zur Verfügung [58].

3.3 Trainingsdaten

In diesem Abschnitt wird die Auswahl und das Erstellen des verwendeten Trainingsdatensatzes erläutert. Die Auswahl der Trainingsdaten hat einen großen Einfluss auf die Qualität eines Neuronalen Netzes.

3.3.1 Einteilung der Trainingsdaten

Der zu entwickelnde Detektor zur Signalerkennung soll als binärer Klassifikator genutzt werden. Die Trainingsdaten werden dazu in die zwei Kategorien **positiver** und **negativer Daten** aufgeteilt. Positive Daten sind Audiodateien, die über ihre komplette Länge einen Signalton enthalten. Negative Daten enthalten keinen Signalton und setzen sich aus zwei verschiedenen Arten von Aufnahmen zusammen. Die erste Art der Aufnahmen sind Straßengeräusche, die im Anwendungsszenario typische Störgeräusche sind. Die zweite Art sind Aufnahmen, die den Signaltönen ähneln.

3.3.2 Quelle der Trainingsdaten

Die Generierung von Datensätzen geeigneter Größe stellt in vielen Anwendungen des maschinellen Lernens eine Herausforderung dar. Es wird meist eine große Anzahl an Daten benötigt, bis Neuronale Netze die gewünschten Merkmale erkennen können [7]. Im Falle der Signaltöne bietet YouTube eine Vielzahl an Rohdaten mit minutenlangen Videos vorbeifahrender Einsatzwagen. Unbearbeitete Aufnahmen werden im Rahmen dieser Arbeit als Rohdaten bezeichnet. Die Audiospuren dieser Videos lassen sich speichern und zu Trainingsdaten weiterverarbeiten. Videos bieten im Vergleich zu reinen Audiodateien den Vorteil, die Aufnahmen anhand der Bilder annotieren zu können.

Nachteile von YouTube-Videos als Quelle sind die unbekannte Aufnahmetechnik sowie die unbekannte Komprimierung. Bei den meisten verwendeten Videos ist von Aufnahmen mittels Telefonen und Digitalkameras auszugehen, in denen typischerweise Elektret- und MEMS-Mikrofone verbaut sind. Wie in Abschnitt 2.3.1 erläutert, weisen die Frequenzgänge solcher Mikrofone vor allem außerhalb des Frequenzbereiches von 300 bis 3400 Hz Nichtlinearitäten auf. Die größte Energiedichte von Signaltönen liegt innerhalb dieses Bereiches (siehe Abschnitt 2.7). Die relevanten Informationen der Aufnahmen werden somit nicht wesentlich verzerrt.

3.3.3 Auswahl der Trainingsdaten

Aus den teilweise mehr als 15 Minuten langen Rohaufnahmen wurden für das Programm lesbare und annotierte Trainingsdaten erstellt. Dazu sind die einzelnen WAV-Dateien nach Signaltonereignissen aufgeteilt, annotiert und als neue WAV-Dateien gespeichert worden. Dieser Prozess wurde mit der Audioverarbeitungssoftware Ableton

Live [59] durchgeführt. In der Software können Audiodateien instantan abgespielt, visuell dargestellt und benannt werden. Dadurch konnten die Aufnahmen sowohl anhand optischer, als auch akustischer Merkmale kategorisiert und geschnitten werden. Um mögliche Fehler in der Vorverarbeitung zu vermeiden, wurde eine einheitliche Abtastrate gewählt und jede WAV-Datei in diese konvertiert und gespeichert. Auf Grund der Kompatibilität mit einer mobilen Anwendung, auf die der Detektor in Zukunft angepasst werden soll, liegt die Abtastrate bei 2^5 = 32768 Hz.

Werden nicht eindeutig zuzuordnende Daten anhand einer subjektiven Einschätzung annotiert, kann dies zu einem Bias des Trainingsdatensatzes führen. Folglich wurden für die positiven Daten nur Aufnahmen verwendet, in denen eindeutige Signaltöne zu erkennen sind. Durch anschließendes Mischen mit Geräuschen und Rauschen in verschiedenen Signal-Geräusch-Verhältnissen, können auch leisere Signale trainiert werden. Forschungen im Bereich Neuronaler Netze zeigen, dass ein Neuronales Netz durch die Addition von Rauschen auf Eingangsdaten weniger anfällig gegenüber Störungen ist. Zusätzlich können durch Rauschen neue Trainingsdaten generiert werden [7].

Die verwendeten negativen Daten setzen sich aus zwei verschiedenen Arten von Aufnahmen zusammen. Der erste Teil enthält Aufnahmen von Straßengeräuschen, die keine Signaltöne enthalten. Diese wurden unter anderem aus den Teilen der positiven Rohdaten generiert, die kein Signalton enthalten. Da mit realen Aufnahmen von Signaltönen trainiert wird, sind diese Störgeräusche zwangsläufig in den positiven Daten enthalten. Durch das Einlernen als negative Daten soll verhindert werden, dass der Ereignisdetektor die Merkmale von Straßengeräusche als positives Signal erlernt.

Die zweite Art negativer Daten sind Aufnahmen, die Signaltönen ähneln. Dadurch soll der Detektor die entscheidenden Merkmale des Signaltons erlernen. Zu den eingelernten ähnlichen negativen Daten zählen unter anderem Aufnahmen von Hupen, Trompeten und Alarmen, die in einem ähnlichen Frequenzbereich liegen wie Signaltöne.

3.3.4 Annotation der Trainingsdaten

Die Daten wurden zu Beginn so ausführlich wie möglich annotiert. Damit sollte sichergestellt werden, dass die benötigten Informationen zu den Trainingsdaten zur Verfügung stehen, ohne dass durch Anpassungen eine erneute Annotation des Datensatzes nötig wird. Tabelle 3.1 enthält Beispiele verwendeter Kategorien.

3.3.5 Umfang und Aufteilung der Trainingsdaten

Ausgangspunkt zur Bestimmung einer geeigneten Datenmenge für das Neuronale Netz ist der von *Tran et al.* [11] verwendete Datensatz. In dieser Arbeit werden für

Kategorie	*Wert*
Wetter	trocken nass Regen
Ort	Stadt Land Autobahn
Fahrzeugtyp (nur bei positiven)	Limousine oder Kombi Van oder Kleintransporter LKW Diverse
Anzahl der Fahrzeuge	1 2 . . . Mehrere
Ähnliche (nur bei negativen)	nicht ähnlich Hupe Trompete etc.

Tabelle 3.1: Auswahl zur Annotation verwendeter Kategorien

die einzelnen Kategorien 750 s Trainingsdaten und 3500 s Validierungsdaten, für die positiven Daten jeweils in verschiedenen Signal-Rausch-Verhältnissen, verwendet. Die ursprüngliche Menge erstellter Trainings- und Validierungsdaten ist in Bild 3.1 dargestellt.

Eine einfache Methode zur Erweiterung der Trainingsdaten ist das Verfälschen der vorhandenen Daten [7]. Dies wird in dieser Arbeit im ersten Schritt dadurch erreicht, dass zur Generierung zusätzlicher positiver Daten diese mit negativen Daten zufällig vermischt werden. Im zweiten Schritt werden alle Daten zusätzlich mit Rauschen versetzt. Auf diese Weise wird der Umfang der Trainingsdaten vervielfacht.

Es existiert keine einheitliche Empfehlung für die Aufteilung des Datensatzes in Trainings- und Validierungsdaten. Grundsätzlich muss der Validierungsdatensatz eine ausreichende Größe haben, um eine repräsentative Aussage über die Genauigkeit des Neuronalen Netzes bei unbekannten Daten treffen zu können. Für große Datensätze kann der Validierungsdatensatz im Verhältnis zum Trainingsdatensatz kleiner gewählt werden als bei einem kleinen Datensatz. Für das Training in dieser Arbeit wurden unterschiedliche Verhältnisse von Trainings- und Validierungsdaten getestet.

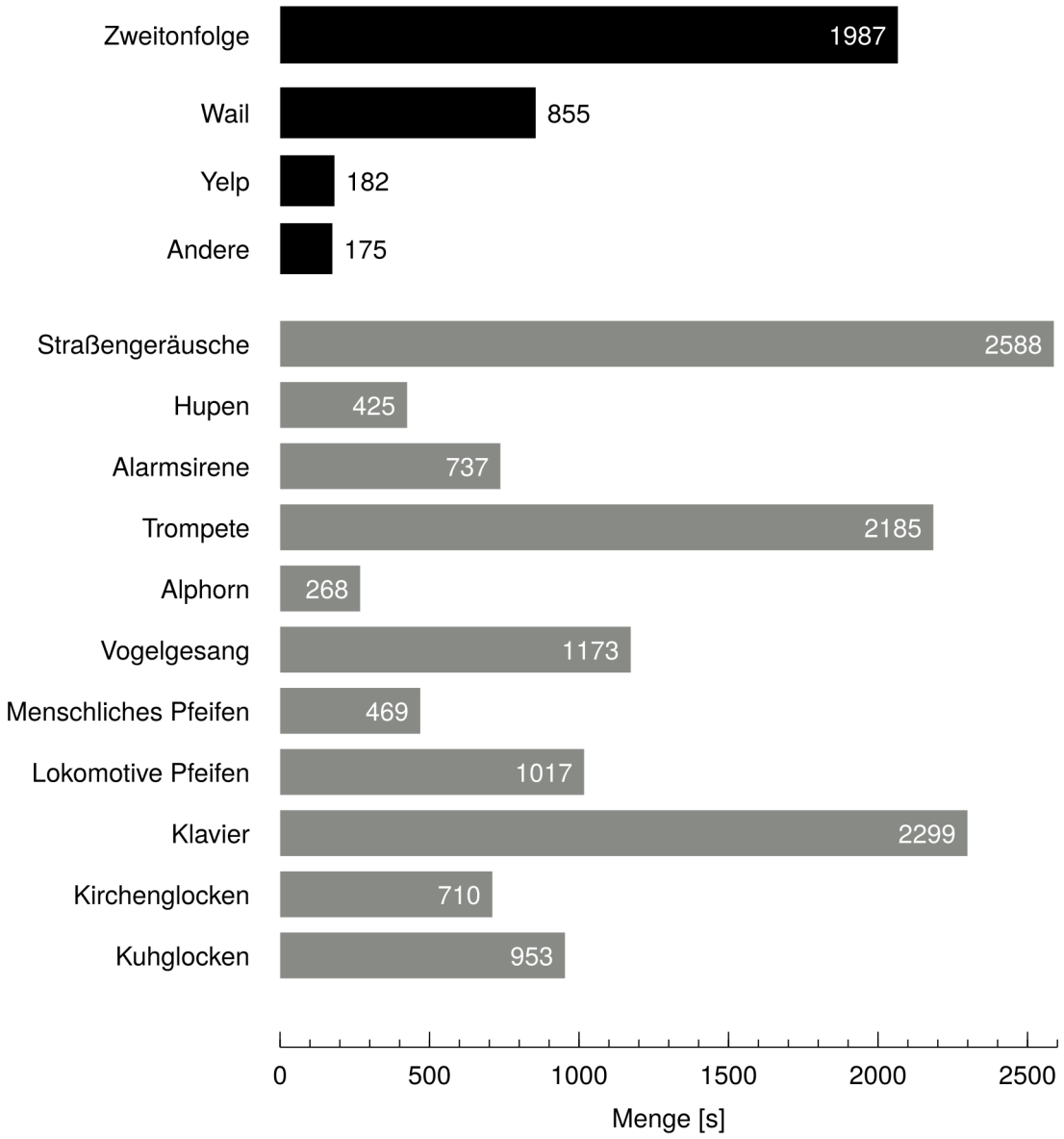


Bild 3.1: Umfang des Trainings- und Validierungsdatensatzes

3.4 Messung

Aus Testgründen wurde für deutsche Signaltöne ein kleiner Datensatz mit eigenen Aufnahmen erstellt. Die Testdaten sollen hinsichtlich ihrer spektralen Zusammensetzung, der Energieverteilung sowie der Lautstärke überprüft werden. Hierfür wurde ein Kleinmembran-Mess-Mikrofon auf einem Balkon an einer stark befahrenen Straße positioniert. Der Aufbau ist in 3.3 dargestellt. In vier Stunden Messung konnten acht Signaltöne aufgenommen werden. Auf Grund der exponierten Lage auf dem Balkon sind die Signaltöne bei wenig Störgeräuschen bereits aus einer Entfernung von ca. 500 m hör- und im Spektrogramm erkennbar (siehe Bild 3.2). Ein direkt vor dem Balkon vorbeifahrendes Signalhorn befindet sich in 5 m bis 15 m Entfernung. Für die direkt vorbeifahrenden Signaltöne konnte mit dem vorhandenen Aufbau keine eindeutige

Lautstärke gemessen werden, da der Schalldruckpegel über den maximal messbaren Wert von 121 dB(A) lag. Folglich liegt der Schalldruckpegel über dem gesetzlich normierten Mindestschalldruckpegel von 110 dB(A) für die Grundtöne. Zum Vergleich liegt der Schalldruckpegel eines vorbeifahrenden Autos bei ca. 60 dB(A), bei Bussen und größeren Fahrzeugen bei bis zu 80 dB(A).

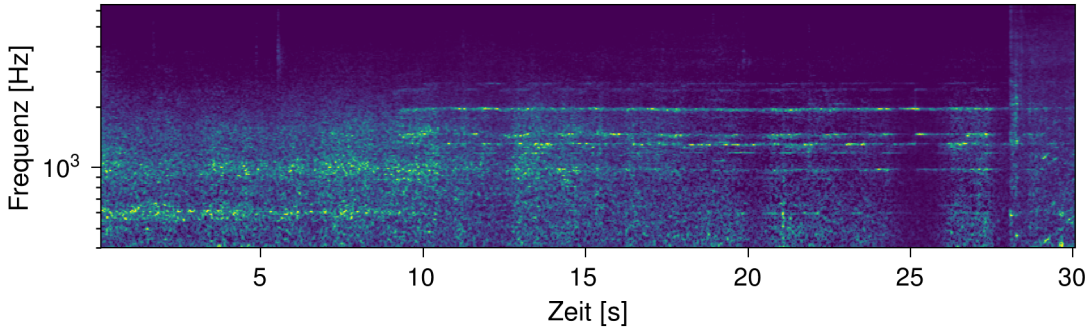


Bild 3.2: Spektrogramm eines Signals in 500m Distanz

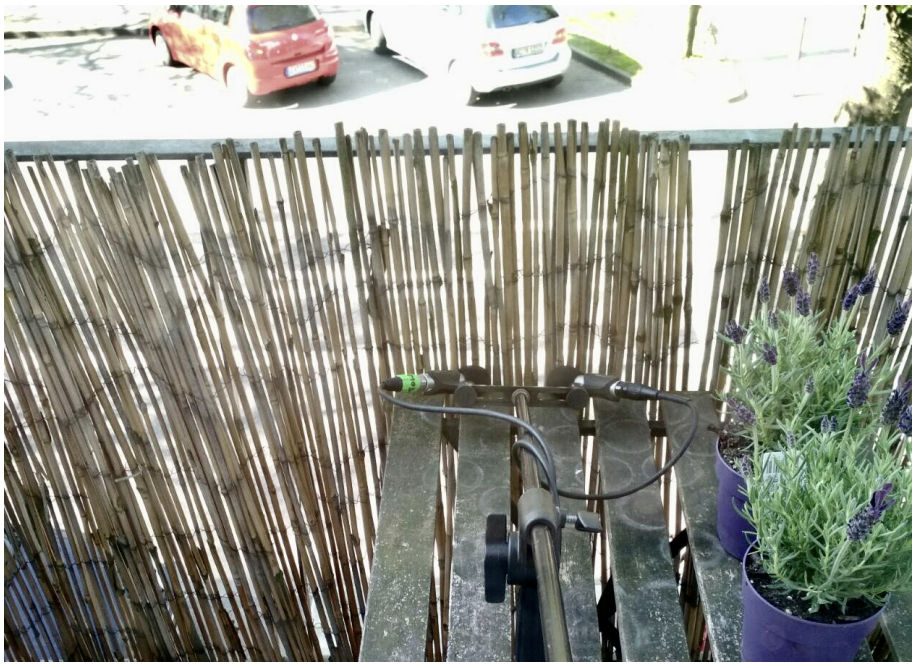


Bild 3.3: Testaufbau

Im Vergleich der gemittelten Energiedichte der eigenen Aufnahmen und der gemittelten Energiedichte der verwendeten Trainingsdaten, siehe Bild 3.4, lässt sich für die eigenen Aufnahmen lediglich eine leichte Verschiebung hin zu höheren Frequenzen feststellen. Die eigenen Aufnahmen verfügen darüber hinaus über eindeutigere Frequenzspitzen, was an einem niedrigere Signal-Rausch-Verhältnis gemittelt über den gesamten Trainingsdatensatz liegen kann. Der grundsätzliche Frequenzgang ist gleich und bestätigt somit die Repräsentativität der verwendeten YouTube-Aufnahmen.

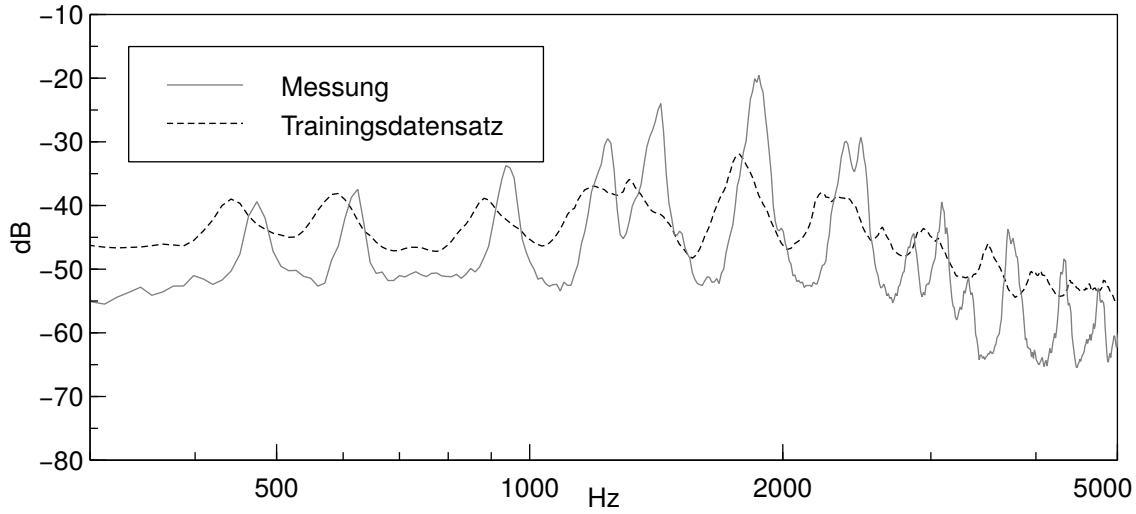


Bild 3.4: Spektrale Energiedichte des Trainingsdatensatzes und der Messung

3.4.1 Eigene Aufnahmen als Trainingsdaten

Vorteil selbst erstellter Trainingsdaten ist die bekannte Aufnahmetechnik. Hieraus lassen sich zusätzliche Informationen ermitteln, die bei unbekannten Quellen fehlen. Größen wie der Schalldruckpegel und die Phase der Aufnahme lassen sich mit Hilfe der technischen Daten des verwendeten Mikrofons ermitteln. Die zusätzlichen Informationen können beispielsweise als Eingang in das Neuronale Netz oder zur Ortung des Signalträgers verwendet werden.

Alle Aufnahmen der Messung wurden unter den gleichen Bedingungen erstellt. Bei Betrachtung der Spektrogramme der aufgenommenen Signaltöne (siehe Bild 3.5) ist erkennbar, dass die Muster ähnlich sind. Auch akustisch sind keine großen Unterschiede erkennbar. Werden viele ähnlich Daten zum Training eines Neuronalen Netzes genutzt, kann dies sowohl zu Overfitting, als auch zu einem Bias im Datensatz führen. Wird ein Neuronales Netz nur mit Daten trainiert, die an einem Ort entstanden sind, kann dies dazu führen, das Merkmale erlernt werden, die nur für diesen Ort gültig sind. Das hat zur Folge, dass das Netz nur in der Lage ist, Daten zu unterscheiden, die an diesem Ort entstanden sind. Wird ein Trainingsdatensatz aus eigenen Aufnahmen erstellt, sollten die Aufnahmen daher an unterschiedlichen Orten und zu unterschiedlichen Zeiten erfolgen.

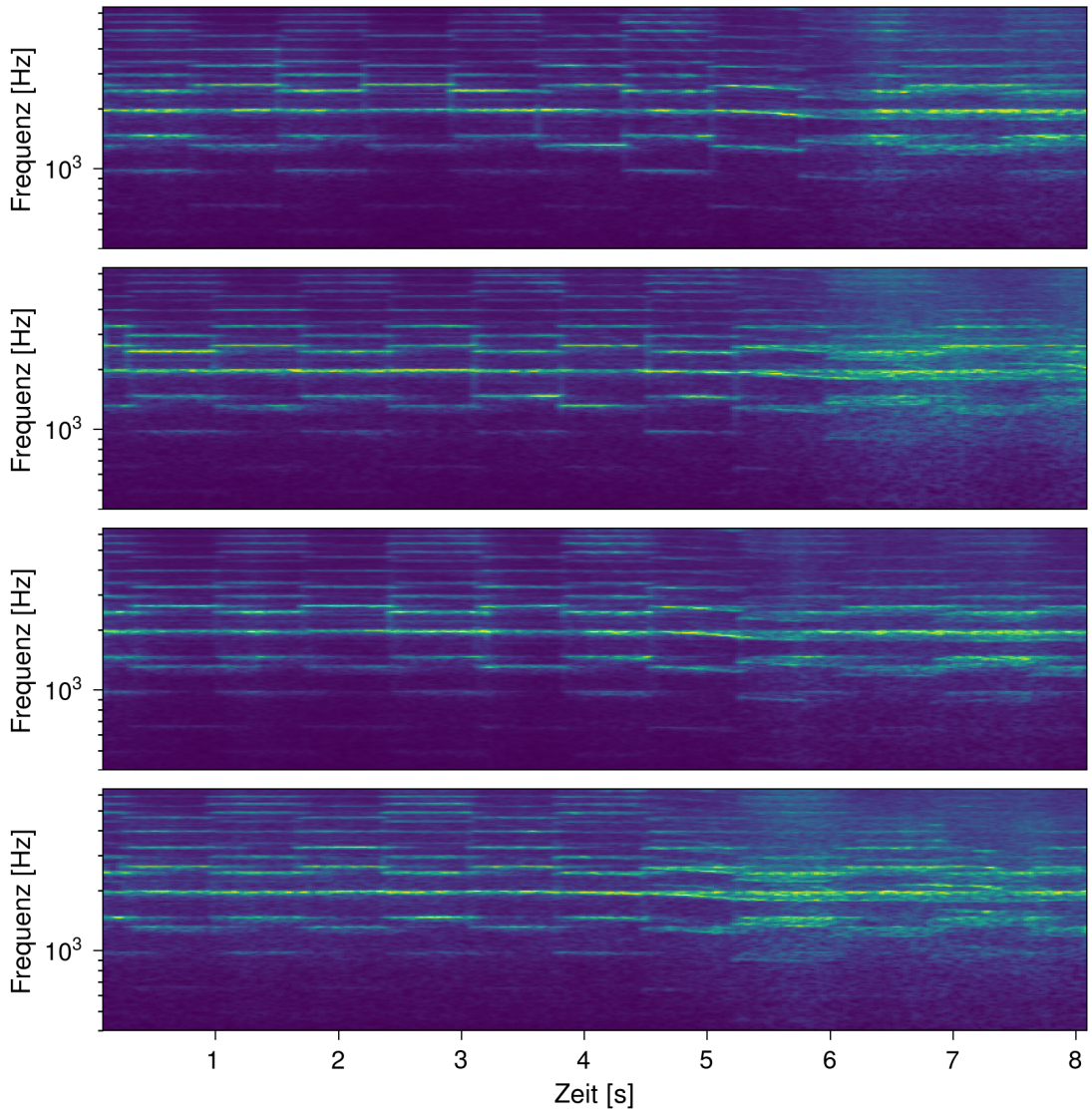


Bild 3.5: Spektrogramme mehrerer aufgenommener Signaltöne des Messdatensatzes

3.5 Vorverarbeitung

Als Vorverarbeitung wird in dieser Arbeit das sogenannte Data Preprocessing bezeichnet. Die Vorverarbeitung im Prozess der Ereignisdetektion erfüllt primär zwei Ziele. Das erste ist die Transformation der Audiosignale in ein für das gewählte Neuronale Netz lesbares Format. Das zweite Ziel ist, anhand von Algorithmen und Funktionen, die Merkmale der Eingangsdaten so eindeutig wie möglich darzustellen. Hierfür sollen Daten, die keine Merkmale der Signaltöne enthalten, herausgefiltert und Daten, die Merkmale enthalten, zusätzlich herausgestellt werden.

Im folgenden Abschnitt werden die einzelnen Schritte der Vorverarbeitung im Training und im Betrieb erläutert. Der gesamte Vorverarbeitungs-Prozess ist in Bild 3.6 schematisch dargestellt. Er lässt sich in vier Stufen teilen. Die erste Stufe ist die Verarbeitung der WAV-Dateien vor der Kurzzeit-Fourier-Transformation, engl. Short-Time-Fourier-

Transformation (STFT). Die zweite Stufe ist die Transformation selber. In der dritten Stufe wird die Transformation nachverarbeitet und in der letzten Stufe werden die Werte in die richtige Eingangsform gebracht.

3.5.1 Definition häufig verwendeter Begriffe

Zu Beginn werden einige häufig verwendete Begriffe definiert, um die Erläuterungen im folgenden Abschnitt verständlicher gestalten zu können.

Mit einer **FFT** wird im Folgenden eine Spalte der STFT-Matrix in Form eines Vektors $\vec{F}$ definiert. Dieser enthält die Amplituden des transformierten Zeitschritts.

Ein **Paket** ist eine festgelegte Anzahl zu einer Matrix zusammengefasster FFTs, die zum Einlernen bzw. Berechnen des NN verwendet werden. Als **Paketgröße** wird die Anzahl der FFTs in einem Paket bezeichnet. Der Zeitraum, den ein Paket abdeckt, ist von der Paketgröße, Hop-Länge der STFT und der Abtastrate abhängig. Als Konvention wird für Hop-Länge ein ganzzahliger Bruch der Blockgröße verwendet.

$$\Delta t_{Paket} = \frac{\text{Hop-Länge} \cdot \text{Paketgröße}}{\text{Abtastrate}} \tag{3.1}$$

3.5.2 Verarbeitung vor der STFT

WAV-Dateien (siehe 2.4.5) können auf unterschiedliche Weise, zum Beispiel mit unterschiedlichen Bittiefen (2.4.1), codiert sein. Dadurch ergeben sich für die ausgelesenen Werte einer WAV-Datei unterschiedliche Größen. Bei einer Bittiefe von 8 Bit liegen die Werte zwischen 0 und 255, bei 16 Bit zwischen -32.768 und 32.767. Da durch die Art der im Trainingsdatensatz verwendeten Quellen (3.3.2) keine Informationen über die Lautstärke der ursprünglichen Signale vorliegt, ist diese auch nicht auswertbar. Es können nur Pegelverhältnisse ausgewertet werden. Daher werden die Daten auf den Betrag des Maximalwerts normiert. Die Normierung erfolgt über den ganzen Vektor $\vec{x}$ nach Gleichung 3.2.

$$f(x_i) = \frac{x_i}{|\max \vec{x}|} \quad \text{mit} \quad \vec{x} = \left(x_1, \dots, x_i, \dots, x_k\right)^T \quad \text{für} \quad 1 \leq i \leq k \tag{3.2}$$

Im Ergebnis liegen die Werte mit unveränderten Verhältnissen mit den Maximalwerten ± 1 vor. Dadurch werden alle Daten auf den größten Amplitudenwert bezogen. Dies ermöglicht die anschließende Addition von Rauschen.

Zur Addition von Rauschen werden zufällige Werte in einem festzulegenden Intervall generiert und auf die Werte der WAV-Datei addiert. Anschließend wird die Normierung nach Gleichung 3.2 nochmals durchgeführt, um die Daten konsistent im Intervall

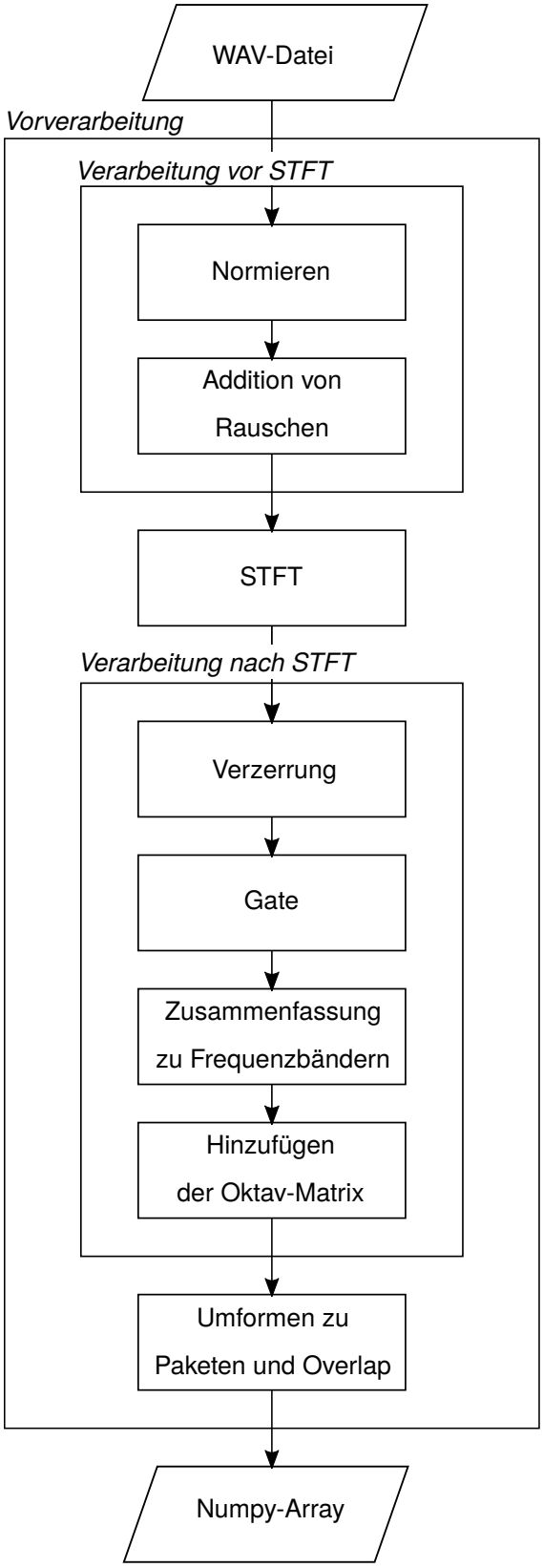


Bild 3.6: Ablaufdiagramm der Vorverarbeitung

$[-1, 1]$ zu halten. Die Addition von Rauschen simuliert folglich ein leiseres Signal mit zusätzlichen, zufälligen Störgeräuschen. In Bild 3.7 ist die Aufnahme eines Martinshorns mit verschiedenen „Noise“-Einstellungen dargestellt. Bei 100% Noise werden zufällige Werte im Intervall $[-1, 1]$ addiert, bei 25% entsprechend im Intervall $[-0{,}25, 0{,}25]$.

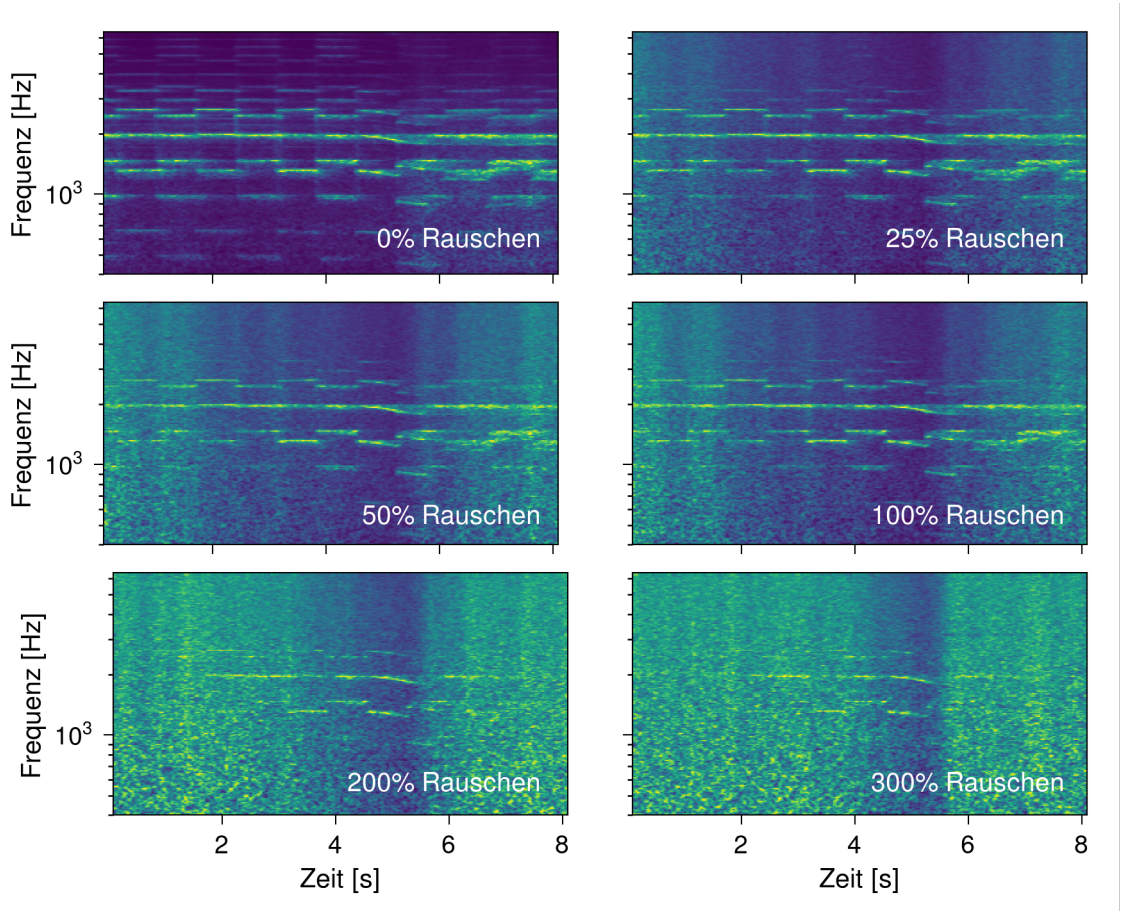


Bild 3.7: Addition von Rauschen auf Signaltonaufnahme

Dieser Schritt kann optional in mehreren Durchgängen mit verschiedenen Rauschverhältnissen durchgeführt werden. Es werden folglich zusätzliche, manipulierte Trainingsdaten generiert. Das Einlernen von Daten, die mit moderatem Rauschen versetzt wurden, kann den Generalisierungsfehler und die Anfälligkeit gegenüber Störgeräuschen verbessern [7].

3.5.3 Short-Time-Fourier-Transformation

Im Anschluss an die Verarbeitung der WAV-Dateien folgt die STFT (siehe 2.5.3), die den zentralen Schritt der Vorverarbeitung darstellt. Wie in Abschnitt 2.7.2 ersichtlich wird, lässt die spektrale Darstellung der Signaltöne klare Muster erkennen, die sich zur Mustererkennung in einem Neuronalen Netz eignen. Die STFT ist ein effizienter und schneller Algorithmus, um das Signal in die Zeit-Frequenz-Darstellung zu transformieren. Sie lässt sich über das `librosa`-Paket `librosa.core.stft()` im Pro-

grammcode integrieren. Die Blockgröße, Hop-Länge und Art der Fensterung werden in dieser Funktion angepasst. Durch die Verwendung des Amplitudenspektrums liegen nur positive Daten vor.

3.5.4 Verarbeitung nach der STFT

Nach der STFT können die Werte der STFT-Matrix „verzerrt“ werden. Durch die Normierung der WAV-Dateien liegen alle Werte der Matrix im Intervall $[0, 1)$. Wird die Matrix beispielsweise elementweise quadriert, werden kleine Werte verhältnismäßig kleiner als große Werte. Dadurch wird der Kontrast in den Daten erhöht. Laute Signale werden deutlicher herausgestellt, gleichzeitig gehen Details verloren. Visuelle Beispiele zu den Verzerrungsfunktionen sind in Bild 3.8 dargestellt. Durch das Quadrieren entspricht die Matrix den Werten eines Spektrogramms. Sollen dagegen kleine und mittlere Werte gegenüber den größeren Werten angehoben werden, kann elementweise die Wurzel gezogen werden. Dadurch werden die Details in den Daten mehr hervorgehoben.

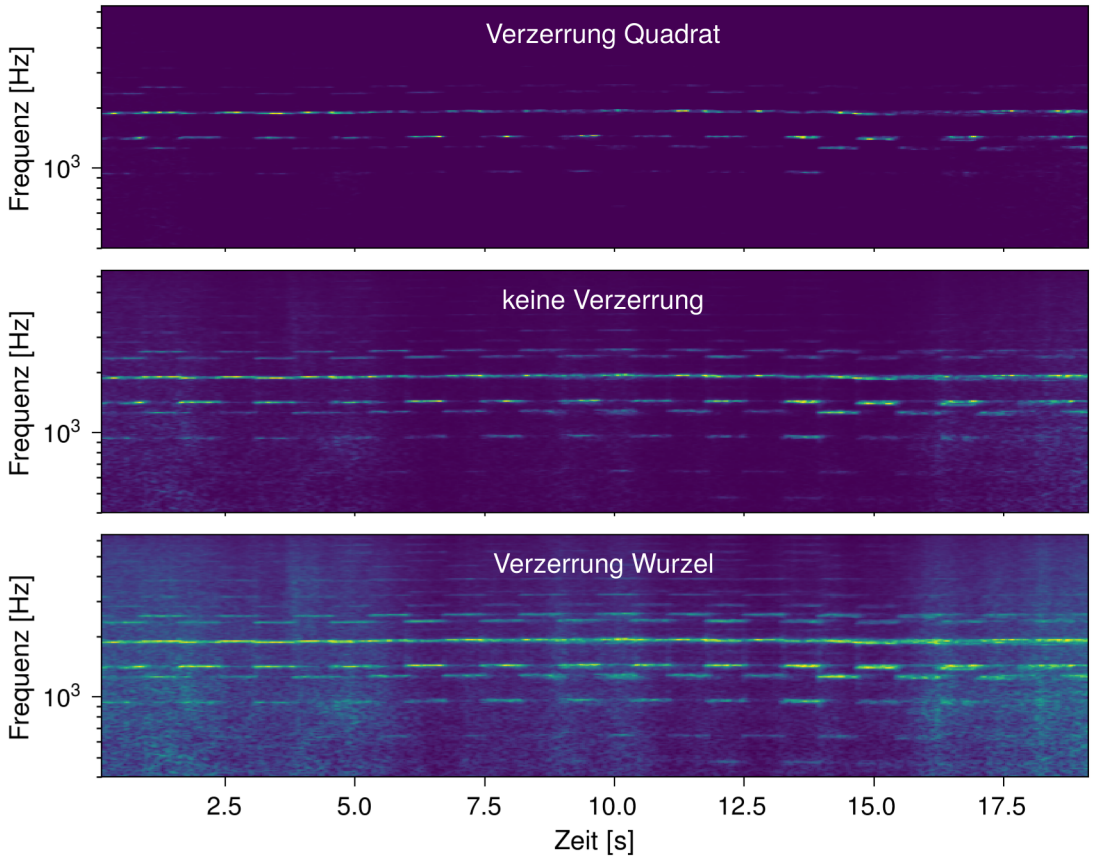


Bild 3.8: Beispiel einer Verzerrung durch Quadratfunktion und Wurzelfunktion und ohne Verzerrung

Die Frequenzen eines Signaltons sind schon bei einer relativ großen Entfernung im Spektrogramm erkennbar. Das bedeutet, dass auch bei der Matrix der STFT für die-

sen Fall über größere Werte verfügen, als der Großteil der restlichen Störgeräusche. Mittels eines Schwellenwertes können die leiseren Frequenzanteile der Störgeräusche herausgefiltert werden. Diese Funktion wird hier Gate genannt. Das Gate ist in seiner Funktionalität dem im maschinellen Sehen zur Segmentierung verbreiteten Schwellenwertverfahren ähnlich (siehe beispielsweise [60]). FFT-Werte, die unterhalb eines einstellbaren Schwellenwertes liegen, werden gleich 0 gesetzt. In Bild 3.9 ist zur Veranschaulichung das Spektrogramm eines gerade eben hörbaren Signaltons ohne und mit Gate-Einstellungen dargestellt. Die gemessene Distanz beträgt ca. 500 m.

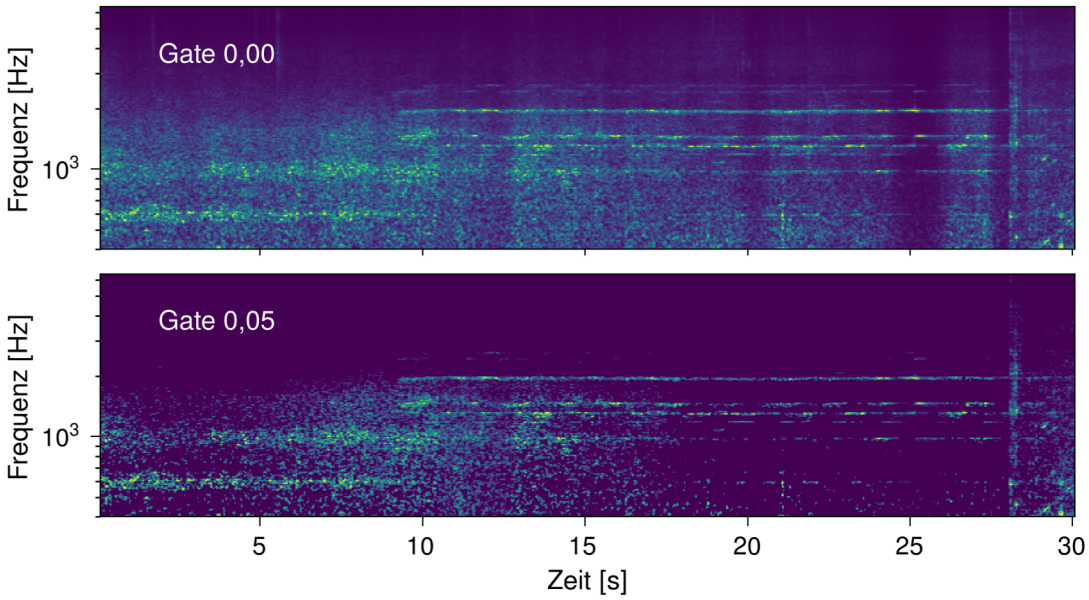


Bild 3.9: Spektrogramm ohne Gate und mit 0,05

3.5.5 Zusammenfassung zu neuen Frequenzbändern

In diesem Schritt werden die Ergebnisse der STFT in neuen Frequenzbändern zusammengefasst. Das Zusammenfassen zu neuen Frequenzbändern erfüllt mehrere Zwecke. Ein Ziel dieses Vorgangs ist die Entfernung tief- und hochfrequenter Bereiche der STFT. Die Signaltöne sind nur in einem begrenzten Frequenzbereich signifikant messbar. Das Einlernen und die spätere Eingabe der Frequenzen außerhalb dieses Bereiches kann im besten Fall keinen Vorteil bringen und im schlechtesten eine Verschlechterung der Erkennung zur Folge haben. Diese Frequenzbereiche enthalten keine Informationen über die zu erkennenden Signale. Es können nur andersgeartete Merkmale erlernt werden, die mit dem Signal selber nichts zu tun haben. Ein sinnvoller Frequenzbereich lässt sich aus der in Abschnitt 2.7.2 untersuchten spektralen Energiedichteverteilung der Signaltöne herleiten. Für die Trainingsdaten, mit denen der Ereignisdetektor schlussendlich trainiert wird, ist die Energiedichte im Bereich von 430 Hz bis 2500 Hz am höchsten. Daher werden die Daten weit außerhalb dieses Bereichs

herausgefiltert. Auf Grund des Aufbaus der Frequenzbänder liegt der verwendete Frequenzbereich zwischen 400 Hz und 3200 Hz.

Das zweite Ziel ist das Herausstellen tonaler Zusammenhänge der Daten. Wie in Abschnitt 2.7.1 beschrieben, setzt sich ein Signalton nicht nur aus den in den Normen festgelegten Grundtönen und einem Oberton zusammen, sondern aus einer ganzen Reihe von Obertönen. In Abschnitt 2.2.1 wird erläutert, dass die Frequenzen von Obertönen immer ganzzahligen Vielfachen der jeweiligen Grundfrequenz entsprechen. Zwischen den einzelnen Frequenzen gibt es einen exponentiellen Zusammenhang. Durch eine neue Einteilung der Frequenzen soll der exponentielle in einen linearen Zusammenhang überführt werden. Einem ähnlichen Ansatz in der Wahl der Frequenzbänder folgt die sogenannte „Constant-Q“ oder Wavelet Transformation, die u.a. bei *Schörkhuber* [61] erläutert wird. Das dadurch entstehende Spektrum ähnelt dem Frequenzempfinden des Menschen und wird unter anderem zur Analyse von Musik verwendet. Die Berechnung der Frequenzen nach dieser Aufteilung folgt der gleichen Logik, nach der die Frequenzverhältnisse bei der Gleichstufigen Stimmung berechnet werden (siehe 2.2.3). Die so errechneten Frequenzen entsprechen zwar nicht exakt den natürlichen Obertonfrequenzen, liegen aber bei den verwendeten Frequenzauflösungen ausreichend nah an diesen. Im Ergebnis ergeben sich keine Unterschiede. Ein weiterer Vorteil, der sich aus der Transformation ergibt, ist eine wesentliche Reduktion der Daten. Bei einer ursprünglichen Blockgröße von 8192 und 3 Oktaven mit je 12 Stufen wird Zahl der Frequenzbänder um das 227,6-fache reduziert. Dies führt zu einem geringeren Speicherplatzbedarf und einer schnelleren Berechnung aufgrund kleinerer Matrizen.

Die Berechnung der neuen Frequenzen erfolgt ausgehend von oberer und unterer Frequenzgrenze, sowie der gewünschten Frequenzauflösung. Zur Vereinfachung wird nicht die exakte obere Grenzfrequenz verwendet, sondern die sinnvolle nächste Oktave. Bei einer Grundfrequenz von 400 Hz sind mögliche obere Grenzwerte also 800, 1600, 3200, 6400, usw. Hz. In Gleichung 2.8 wurde bereits die Berechnung der Töne einer Oktave ausgehend von einer Referenzfrequenz dargestellt. In Gleichung 3.3 wird dieser Zusammenhang für M Oktaven und n Stufen pro Oktave erweitert. Die einzelnen Frequenzen für die Grundfrequenz f_G lassen sich wie folgt berechnen:

$$f_{ij} = f_G \cdot 2^{j+\frac{i}{n}} \qquad \text{für} \qquad 0 \leq j \leq M, \qquad 0 \leq i \leq n \tag{3.3}$$

mit f_{ij} – Frequenzband
f_G – Grundfrequenz
j – j-te Oktave
i – i-te Stufe der Oktave
M – Anzahl der Oktaven
n – Stufen pro Oktaven

Da für alle Daten eine konstante Blockgröße und Abtastrate verwendet wird und die Variablen für Gleichung 3.3 auch einmalig eingestellt werden, sind die alten und neuen Frequenzen für das gesamte Training und den Betrieb fest bestimmt. Daher wird einmalig mit den Vektoren der STFT-Frequenzen und der transformierten Frequenzen eine Zuordnungsmatrix berechnet. Die Neuzuweisung der STFT-Ergebnisse erfolgt im weiteren Verlauf durch die Matrixmultiplikation von STFT-Matrix und Zuordnungsmatrix. Die Frequenzen der STFT werden dabei gewichtet auf die neuen Frequenzen verteilt. In Tabelle 6.1 im Appendix ist der Ausschnitt einer Transformationsmatrix für die Blockgröße 8192 und 12 Stufen pro Oktave dargestellt.

Oktaven-Matrix

Als zusätzlichen Eingang in das NN kann der STFT-Matrix eine Oktaven-Matrix angefügt werden. Bei diesem werden alle Oktaven aufeinander addiert und anschließend normiert. Es entsteht eine Matrix, in der die Anzahl der Zeilen der verwendeten Stufenanzahl entspricht. Die tonalen Zusammenhänge sind implizit bereits in den Daten enthalten, können allerdings durch die Hinzugabe der Oktaven-Matrix explizit herausgestellt werden. Bei relativ leisem Signal kann sich durch das Addieren der Obertöne das zu erkennende Muster deutlich herausbilden. Die Summierung der Oktaven-Matrix wird ebenfalls mittels einer Matrizenmultiplikation berechnet.

3.5.6 Umformung und Overlap

Um die Trainingsdaten einlernen zu können, müssen diese zunächst in Pakete umgeformt werden. Hierzu werden die einzelnen Matrizen, die zusammenhängend aus der WAV-Datei vorverarbeitet wurden, in Pakete der gewünschten Paketgröße aufgeteilt.

Optional kann der Overlap (dt. Überlappung) angewandt werden. Dieser ermöglicht es, statt abgegrenzte Pakete, Überlappungen zwischen ihnen zu verwenden. Beispielsweise werden bei einem *Overlap* = 5 die zeitlich fünf ältesten Blöcke entfernt und fünf neue hinten angefügt. Im Training ist ein erwartbarer Vorteil, dass eine Signaltonaufnahme nicht nur mit einer Startposition des Signals, sondern in verschiedenen Positionen eingelernt wird. Als Beispiel kann das Sample einer Zweitonfolge einmal mit dem Start

beim tieferen und einmal beim höheren Ton eingelernt werden.

3.5.7 Vorverarbeitung im Betrieb

Der Ereignisdetektor soll in der Lage sein, die Signaltöne in Echtzeit zu erkennen. In Echtzeit bedeutet in diesem Zusammenhang, dass das Signal nach Auftreten innerhalb weniger Sekunden erkannt wird. Um dieses Ziel zu erreichen, müssen mehrere Voraussetzungen erfüllt werden. Zum einen muss die jeweils betrachtete Aufnahme ausreichend kurz sein, zum anderen muss das Neuronale Netz in der Lage sein, das Signal in einer so kurzen Aufnahme zu erkennen. Hinzu kommen Beschränkungen im Hinblick auf die Hardware. So wird das Modell auf einem vollwertigen Computer trainiert, auf dem Multiprocessing möglich ist. Dies ist in einer mobilen Anwendung nicht ohne Weiteres möglich. Die einzigen Unterschiede zur Vorverarbeitung im Training ist das Auslassen der Addition von Rauschen und logischerweise des Mischens mit negativen Trainingsdaten.

Signaleingang

Im Gegensatz zum Training des Neuronalen Netzes liegen die verwendeten Daten im Echtzeitbetrieb nicht als WAV-Datei vor. Dafür wird das mit einem Mikrofon aufgenommene Signal mit Hilfe des `pyAudio`-Pakets in einen Vektor umgewandelt. Mit Hilfe des Paketes können alle systemkompatiblen Mikrofone verwendet werden. Die beim Training verwendeten Modell-Parameter, wie Abtastrate und Blockgröße, müssen für die Aufnahme übernommen werden. Die verwendete Paketgröße bleibt die gleiche wie im Training. Jeder neu aufgenommene Block wird dem Eingangsvektor angefügt und der älteste Block entsprechend entfernt. Im Ergebnis wird eine Matrix generiert die dem Format des Trainings gleicht.

Overlap im Echtzeitbetrieb

Im Echtzeitbetrieb können durch einen Overlap kontinuierlich Vorhersagen getroffen werden. Durch die Überlappung ist der Zeitabstand der berechneten Vorhersagen kürzer als die zeitliche Dauer eines Paketes selbst. Wird der Overlap nicht verwendet, kann bei einer Paketdauer von 1,5 s nur jede 1,5 s eine Vorhersage getroffen werden. Mit Overlap kann beispielsweise bei gleichbleibender Paketgröße in einem Abstand von 100 ms eine Wahrscheinlichkeit berechnet werden.

3.6 Neuronales Netz

Die Architektur eines Neuronalen Netzes ist das Ergebnis eines iterativen Prozesses. Zur genauen Architektur existieren trotz intensiver Forschung nur Empfehlungen. Je-

des Neuronale Netz stellt eine maßgeschneiderte Anpassung an das jeweilige Problem dar [5].

3.6.1 Auswahl des Typs des Neuronalen Netzes

Die in Abschnitt 2.8.2 vorgestellten Ergebnisse von *Xia et al.* [12] zeigen, dass neben klassischen Feed-Forward-Netzen, den MLP, die CNN, RNN und CRNN erfolgreich in der AED zum Einsatz kommen. Während CNN sich vor allem in der Bilderkennung bewährt haben, eignen sich RNN zur Erkennung sequenzieller Daten, zum Beispiel zur Spracherkennung. CRNN versprechen die Stärken beider Netztypen zu vereinen, sind momentan aber noch Gegenstand der Forschung und werden nicht in der Breite angewandt. *Tran et al.* [11] haben in ihrer Arbeit die Nutzung eines MLP- und eines LSTM-Netzes zur Erkennung von Signaltönen untersucht und kommen zu dem Ergebnis, dass beide grundsätzlich geeignet sind Signaltöne zu erkennen. Das LSTM-Netz erreicht aber eine wesentlich höhere Genauigkeit und Rechengeschwindigkeit. Bei *Sak* [62] wird das Ergebnis im Hinblick auf die höhere Genauigkeit bestätigt.

Die Ergebnisse lassen den Schluss zu, dass das LSTM-Netz ein geeignetes Netz zur Erkennung von Signaltönen ist. Auch mit anderen Ansätzen lässt sich ein funktionierender Detektor implementieren. Das LSTM hat aber seine grundsätzliche Funktionalität bei *Tran et al.* [11] erwiesen. Da die Erkennung im Echtzeitbetrieb erfolgen soll, spricht gegenüber dem MLP besonders die höhere Rechengeschwindigkeit für das LSTM. Auch der Ansatz eines CRNN scheint vielversprechend. Im Gegensatz zu diesem haben sich reine LSTM-Netze in der AED bereits bewährt. Diese Umstände führten auf die Wahl eines LSTM-Netzes.

3.6.2 Keras

Die Verwendung von Keras erleichtert das Erzeugen von gewünschten Neuronalen Netzarchitekturen. Die Keras Bibliothek beinhaltet eine Vielzahl an Schichttypen und Funktionen, die zur Erstellung einer Netzarchitektur benötigt werden. Dabei ist die Programmierung modular und es ist möglich innerhalb weniger Zeilen Programmcode eine komplexe Netzarchitekur zu generieren. Ebenso beinhaltet die Bibliothek Module für das Training des erzeugten Netzwerkes. Trotz einfachen Handhabung, ist das Netz noch hochgradig verstellbar und es können Änderungen bis tief in einzelne Funktionen getätigt werden.

3.6.3 Einstellungen des Neuronalen Netzes

In der AED haben sich besonders Netze mit wenigen Schichten als erfolgreich erwiesen [57]. Gleichzeitig können zu wenige Schichten zu Overfitting führen [62]. Als

Ausgangsarchitekturen werden drei Architekturen getestet. Auf die Eingangsschicht folgen ein, zwei oder drei LSTM-Schichten, gefolgt einer Fully Connected Layer und der Ausgangsschicht. Von diesen Architekturen ausgehend, werden unterschiedliche Parametereinstellungen getestet. Bei den einstellbaren Parametern handelt es sich um:

- die Anzahl der Knoten
- die Größe der Dropouts
- die Verwendung der Batch-Normalization
- die Aktivierungsfunktionen
- den Optimierungsalgorithmus
- die Lernrate des Optimierungsalgorithmus
- die Loss-Funktion

Die Anzahl der anpassbaren Parameter ist also recht groß und es lassen sich nur begrenzt Vorhersagen treffen, welche Parameter einen positiven Einfluss auf die Resultate haben. Die Abhängigkeit verschiedener Parameter voneinander spielt hierbei eine entscheidende Rolle. Beispielsweise kann sich durch austauschen der Aktivierungsfunktionen der Wertebereich verändern und nachfolgende Funktionen müssen an diesen angepasst werden. Letztlich lässt sich erst nach dem Einlernen und Testen abschätzen, welche Änderungen zu besseren Ergebnissen geführt haben.

Für die **Anzahl der Knoten** existieren Empfehlungen, die als Ausgangspunkt verwendet und darauf folgend iterativ angepasst werden. Eine Faustregel für die erste verdeckte Schicht ist die (Anzahl Eingaben + Anzahl Ausgaben) / 2 zu wählen [41]. Bei der verwendeten Vorverarbeitung entspricht dies etwa 64 Knoten. Als Ausgangsarchitektur bezüglich der Knoten wurden folgende Werte gewählt:

Eingang	1. Schicht	2. Schicht	3. Schicht	4. Schicht	Ausgang
121	LSTM 128	MLP 32			2
121	LSTM 64	LSTM 128	MLP 32		2
121	LSTM 64	LSTM 128	LSTM 64	MLP 32	2

Tabelle 3.2: Verwendete Ausgangsarchitekturen der Neuronalen Netze

Ausgehend von diesen Werten wurden unterschiedliche Knotenanzahlen getestet. Zusammenhängend mit der Anzahl der Knoten ist die Größe des Dropouts. Wird ein großer **Dropout** gewählt, werden während des Trainings mehr Knoten entfernt. Dementsprechend muss die Knotenanzahl erhöht werden [43]. Für die Größe des Dropouts

wird ein Wert zwischen 0,5 und 0,8 empfohlen. Es werden somit zwischen 50% und 80% der Knoten nicht entfernt. In Keras muss beachtet werden, dass der Anteil der zu entfernenden Knoten angegeben wird. Folglich liegen die Werte hierbei zwischen 0,2 und 0,5.

Die **Batch Normalization** wird grundsätzlich nach rekurrenten Schichten verwendet, da sie die Lernrate steigert und Konvergenz-Probleme der Gewichte mildert [44].

Die **Aktivierungsfunktion** der LSTM-Schichten ist standardmäßig die tanh-Funktion. Für die Fully Connected Layer wurden die Sigmoidfunktion und ReLU getestet.

Als **Optimierungsalgorithmus** wird der Adam Optimizer verwendet, wie von *Kingma und Ba* [63] vorgestellt; als **Loss-Funktion** die Cross Entropy. Das verwendete Netz hat zwei Ausgänge und die verwendete **Aktivierungsfunktion des Ausgangs** ist die Softmax-Funktion, welche der Sigmoidfunktion für mehrere Klassen entspricht [64]. Im Ergebnis erhält man für den ersten Ausgang die Wahrscheinlichkeit für ein positives Signal und für den zweiten Ausgang die Wahrscheinlichkeit eines negativen Signals. Durch die Softmax-Funktion ergibt die Summe der Wahrscheinlichkeiten 1.

Zum Training des Neuronalen Netzes wird das sogenannte **Early Stopping** verwendet [7]. Das trainierte Netz mit den verwendeten Netzparameter (Gewichte, Bias, etc.) wird für jede Epoche gespeichert, in der der Validation Loss über einen bestimmten Wert sinkt. Das Netz wird so lange trainiert, bis der Validation Loss sich für eine bestimmte Anzahl von Epochen nicht verbessert hat. Die Parametrisierung mit dem kleinsten Validation Loss wird als finales Modell gespeichert. Die Anzahl der Epochen, in denen sich der Validation Loss bis zum Abbruch nicht ändert, ist individuell einstellbar.

3.6.4 Test und Anpassung der Parameter

Die vorgestellten Ausgangsarchitekturen werden für verschiedene Vorverarbeitungs- und Netzeinstellungen trainiert und anhand der in Abschnitt 3.8 vorgestellten Methodik ausgewertet. Auf Basis der Auswertung werden neue Einstellungen getestet, so dass im Zuge dieses iterativen Prozesses Netze gefunden werden sollen, die die Klassifizierungsaufgabe mit einer zufriedenstellenden Genauigkeit lösen.

3.7 Datenbank

Im Zuge des Trainings eines Neuronalen Netzes fallen eine Vielzahl unterschiedlicher Daten an. Diese Daten lassen sich durch Anlegen einer relationalen Datenbank automatisiert verwalten und speichern. Für diese Arbeit wurde eine SQLite-Datenbank

(siehe 3.2.2) erstellt. Im Folgenden wird auf den Aufbau eingegangen. Die Datenbank lässt sich in zwei Teile aufteilen. Im ersten Teil werden die Trainingsdaten und dessen Annotation verwaltet, im zweiten Teil die Einstellungen und Ergebnisse.

3.7.1 Trainingsdaten

Im Wesentlichen beinhaltet der erste Teil der Datenbank die Informationen bezüglich der Herkunft der Trainingsdaten und ihre Annotation (siehe Abschnitt 3.3). Zunächst werden die Rohdaten in die Datenbank eingepflegt. Die geschnittenen Einlerndaten werden umbenannt und können automatisiert in der Datenbank hinterlegt und mit der Ursprungsdatei verknüpft werden. Die Annotation der geschnittenen Daten bleibt bis auf das Ereignis unverändert und erbt die Informationen der Rohdatei.

3.7.2 Einstellungen und Ergebnisse

Der zweite Teil der Datenbank wird für den Trainingsbetrieb, dessen Dokumentation und zur Auswertung der Ergebnisse verwendet. Für jeden Einlernvorgang werden alle verwendeten Einstellungen gespeichert. Unterschieden wird zwischen Einstellungen, die die Vorverarbeitung betreffen und die nur Einfluss auf das Tensorflow-Model haben. Vor jedem Einlernvorgang werden die Einstellungen mit vorherigen Einlernvorgängen verglichen. Wurde beispielsweise nur das Modell verändert, aber nicht die Vorverarbeitung, werden die abgespeicherten Einlerndaten geladen. Dadurch können wiederholte Rechnungen vermieden werden. Verwendete Einstellungen können zu einem späteren Zeitpunkt nachvollzogen und anhand von Identifikationsnummern (IDs) aus der Datenbank geladen werden. Auch können Trainings sowie Tests mit Hilfe der IDs wiederholt werden.

Die Ergebnisse des Trainings werden ebenfalls gespeichert. Dafür werden für jede Epoche und für das endgültige Modell die Kenngrößen Accuracy, Loss, Validation Accuracy und Validation Loss in die Datenbank geschrieben. Hinzukommen die Anzahl der Trainings- und Validierungssamples, sowie das Verhältnis negativer zu positiven Daten.

Durch die Verknüpfung aller Daten und Informationen, lässt sich für jedes eingelernte Netz die verwendeten Trainingsdaten und jede Einstellung nachvollziehen. Netze können nach allen Parametern gefiltert, sortiert und verglichen werden. Letztlich werden auch die Ergebnisse des Tests mit dem Testdatensatz in der Datenbank gespeichert. Diese Tests werden in Abschnitt 3.8.2 detailliert erläutert.

3.8 Methodik der Auswertung

3.8.1 Einlernergebnisse Tensorflow

Die grundlegenden Ergebnisse des Trainings eines Tensorflow-Modells unter Verwendung der Kreuzvalidieurng sind Accuracy, Loss, Validation Accuracy und Validation Loss. Die Accuracy gibt die Genauigkeit an, mit der die Trainingsdaten mit der gewählten Parametrisierung erkannt werden. Der Loss ist ein Maß für die Abweichung der Ergebnisse des Netzes vom erwarteten Ergebnis für die Trainingsdaten. Die Validation Accuracy und der Validation Loss beziehen sich jeweils auf die Validerungsdaten.

Für die erste Einschätzung der Qualität eines Neuronalen Netzes können diese Größen betrachtet werden. Wird keine hohe Accuracy erreicht, ist das Neuronale Netz für die gewählten Einstellungen nicht in der Lage, die gewünschten Merkmale zu erkennen. Accuracy und Validation Accuracy sollten in einem ähnlichen Bereich liegen. Wird eine ausreichend hohe Accuracy aber keine hohe Validation Accuracy erreicht, ist von Overfitting auszugehen.

3.8.2 Zusätzliche Validierung mit ausgewählten Daten

Die Tensorflow-Ergebnisse erlauben eine erste Einschätzung, ob das Netz für die gewählten Einstellungen trainiert werden kann. Allerdings lässt sich daraus nicht herleiten, welche Art von Daten durch das Netz richtig klassifiziert werden und welche nicht. Zur Bewertung der Funktionalität der Netze wurden mehrere Testaufnahmen erstellt, mit denen jeweils die Erkennung spezieller Eigenschaften getestet werden soll. Im Einzelnen wurden Aufnahmen erstellt von:

- einem klaren Signalton
- weißem Rauschen gefolgt von einem klaren Signalton
- weißem Rauschen mit einem ansteigendem Signalton
- mehreren Signaltönen gleichzeitig
- kurzen Signaltönen (keine vollständige Modulation)
- dem Signalton ähnlichen Signalen
- Aufnahmen von Straßengeräuschen gefolgt von Straßengeräuschen mit Signalton

Die Testaufnahmen werden zusätzlich mit Zeitstempeln versehen, die positive Signale in den Aufnahmen markieren. So können die Netze für diese Aufnahmen automatisiert getestet werden und die Genauigkeit für die jeweilige Aufnahme in der Datenbank hinterlegt werden. Auf diese Weise können beispielsweise Netze, die die Aufnahme des klaren Signaltons nicht ausreichend erkennen, über die Datenbank direkt herausgefiltert werden.

Über die Aufnahme von weißem Rauschen gefolgt von einem Signalton (im Folgenden Drop-Test genannt) wird untersucht, wie schnell das Netz bei einem Wechsel von negativ zu positiv reagiert. Bei Rauschen mit einem ansteigenden Signalton (Ramp-Test) wird untersucht, wie anfällig das Netz für Rauschen ist. In dem Test steigt die Signallautstärke zunächst von -65 dB auf -1 dB kontinuierlich um 0,5 dB pro Sekunde. Das weiße Rauschen ist durchgängig hinterlegt und liegt bis zu diesem Punkt konstant bei -25 dB. Damit das Signal nicht übersteuert, bleibt der Signalton ab diesem Zeitpunkt bei -1 dB und das Rauschen sinkt mit 0,5 dB/s bis auf -65 dB.

Mit der Aufnahme von mehreren Signaltönen gleichzeitig (Multiple-Test) und den kurzen, unvollständigen Signaltonaufnahmen (Short-Test) wird das Verhalten für nicht explizit trainierte Situationen im Straßenverkehr untersucht. Mit den dem Signalton ähnlichen Aufnahmen (Similar-Test) wird die Anfälligkeit für falsch-positive Vorhersagen getestet. Dafür werden unter anderem Aufnahmen von Instrumenten verwendet, die den Signalton imitieren. Mit der Aufnahme von Geräuschen des Straßenverkehrs und Signaltönen im Straßenverkehr (Street-Test) wird das Verhalten für den vorhergesehenen Einsatzort getestet.

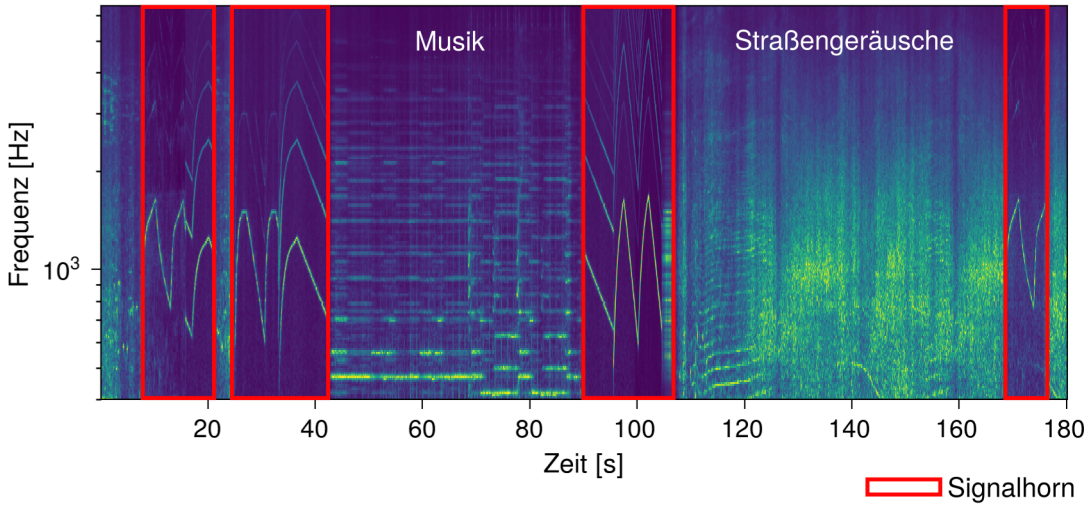


Bild 3.10: Score-Test Aufteilung und Signalvorkommnisse

Final werden die Netze mit einer 180 s langen Aufnahme (Score-Test) getestet, die aus positiven und negativen Aufnahmen zusammengestellt ist. In Bild 3.10 ist ein Spektrogramm des Score-Test abgebildet. Die Aufnahmen sind unabhängig von Trainings- und Validierungsdatensatz. Das Netz trifft für jedes betrachtete Paket eine Vorhersage, mit welcher Wahrscheinlichkeit das untersuchte Paket einen Signalton enthält. Für die getroffenen Vorhersagen werden die Richtig-Positiv-Rate (true positive rate, TPR) und die Falsch-Positiv-Rate (false positive rate, FPR) bestimmt. Dafür wird der festgelegte Schwellenwert von 90% verwendet. Für eine Wahrscheinlichkeit $\geq$ 0,9 wird eine positive Vorhersage angenommen, darunter eine negative. Anhand der TPR und der FPR

können die Netze untereinander verglichen werden.

4 Auswertung

4.1 Trainingsdaten

Für die verwendeten Trainingsdaten lässt sich grundsätzlich feststellen, dass die Verwendung von YouTube-Videos zum Erstellen eines Trainingsdatensatzes geeignet ist. Im Vergleich der Energiedichteverteilung von YouTube-Aufnahmen und eigenen Aufnahmen lassen sich keine wesentlichen Unterschiede erkennen.

Ein Großteil der gefundenen Videos zeigt nicht nur Einsatzwagen während der Nutzung der Signaltöne, sondern auch den Verkehr vorher und nachher, so dass mit der Generierung positiver Daten gleichzeitig auch negative Daten generiert werden können. Durch die Addition von Rauschen in verschiedenen Verhältnissen und durch den Overlap der Pakete kann die Menge der Trainingsdaten auf einfache Art und Weise vervielfacht werden. Durch das Training mit diesen Daten konnte kein Overfitting festgestellt werden. Das Training mit ähnlichen negativen Daten verbessert die Unterscheidung von positiven und ähnlichen Daten.

Für US-Signaltöne konnte im Zuge der Erstellung des Trainingsdatensatzes festgestellt werden, dass neben den genormten Signaltönen Wail und Yelp, weitere Töne zum Einsatz kommen. Diese werden direkt vor Hindernissen im Straßenverkehr und meist nur für Bruchteile von Sekunden eingesetzt. Das Erstellen eines Datensatzes bestehend aus diesen Aufnahmen gestaltet sich daher schwierig, da wenig Rohaufnahmen mit Signalen in ausreichender Länge vorhanden sind.

4.1.1 Annotation der Trainingsdaten

Die durchgeführte, ausführliche Annotation der Aufnahmen hatte zu großen Teilen keinen großen Nutzen. Probleme in der Erkennung von Signalen, zum Beispiel bei nasser Straße, die zu Beginn vermutet wurden, sind nicht eingetreten. Der entwickelte Detektor zeigte keine Probleme positive Signale zu erkennen. Folglich war eine Unterscheidung zwischen positiven und negativen Daten ausreichend. Die Annotation war für die Trennung der unterschiedlichen Kategorien negativer Daten hilfreich. Die Komplexität der Trainingsdaten-Zusammensetzung konnte dadurch angepasst werden.

4.1.2 Umfang der Trainingsdaten

Im Rahmen der Auswertung konnte festgestellt werden, dass der hier verwendete Umfang der Trainingsdaten zur sicheren Erkennung der Signale ausreichend ist. Im Ver-

hältnis von positiven zu negativen Daten muss darauf geachtet werden, dass diese ungefähr im Verhältnis 1 : 1 trainiert werden. Überwiegt eine Klasse wesentlich in der Menge, werden zu erkennende Daten eher der Klasse mit mehr Einlerndaten zugeordnet. Der größeren Menge negativer Daten im ursprünglichen Datensatz wird durch das zusätzliche Mischen positiver mit negativen Daten begegnet.

4.2 Generierung zusätzlicher Trainingsdaten

Werden als negative Daten nur Straßengeräusche verwendet, kommt man in der Unterscheidung dieser Daten zügig und mit vielen unterschiedlichen Einstellungen zu guten Ergebnissen. Auch für verrauschte Signale gelingt die Erkennung gut. Diese Unterscheidung stellt somit keine schwierige Aufgabe für die untersuchten Neuronalen Netze dar.

Allerdings kann es bei der Erkennung durch diese Netze auch zur falsch positiven Erkennung von Signalen kommen, die, wie Signaltöne, tonalen Charakter haben. In Bild 4.1 ist als Beispiel die Vorhersage eines Neuronalen Netzes für eine Testaufnahme mit Pfeifen und deutschen Signaltönen dargestellt. Das Netz wurde mit US-Signaltönen und Straßengeräuschen trainiert. Folglich sollte es bei der Testaufnahme nicht reagieren. Durch die Addition von Rauschen auf duplizierte Trainingsdaten wird das Netz weniger anfällig für die falsch positive Erkennung, wie in Bild 4.1 im unteren Plot zu sehen ist.

Sollen falsch positive Ergebnisse sicher ausgeschlossen werden, ist es unumgänglich, das Netz auch auf die falsch erkannten Daten als negative zu trainieren. Bild 4.2 zeigt ein Beispiel, in dem das Neuronale Netz zusätzlich zu Straßengeräuschen mit den Signaltönen ähnlichen Aufnahmen als negative Daten trainiert wurde.

4.3 Vorverarbeitung

Im folgenden Abschnitt werden die gewählten Methoden der Vorverarbeitung analysiert und anhand Neuronaler Netze zur Erkennung deutscher Signaltöne bewertet. Die einzelnen Vorverarbeitungsschritte können die Erkennung eines Signals lediglich begrenzt verbessern. Entscheidend ist ihr Zusammenwirken. Der Einfluss von Einstellungen, die sich nicht direkt aus den Merkmalen der Signaltöne ergeben, wurden iterativ untersucht. Dafür wurden einzelne Parameter sukzessive verändert und die Erkennung anhand der Testdaten bewertet. Über die erstellten Testdaten konnten mit den in 4.1 und 4.2 dargestellten Einstellungen gute Ergebnisse erreicht werden. Bild 4.3 zeigt das Spektrogramm nach der Vorverarbeitung sowie die Vorhersage des Netzes für den

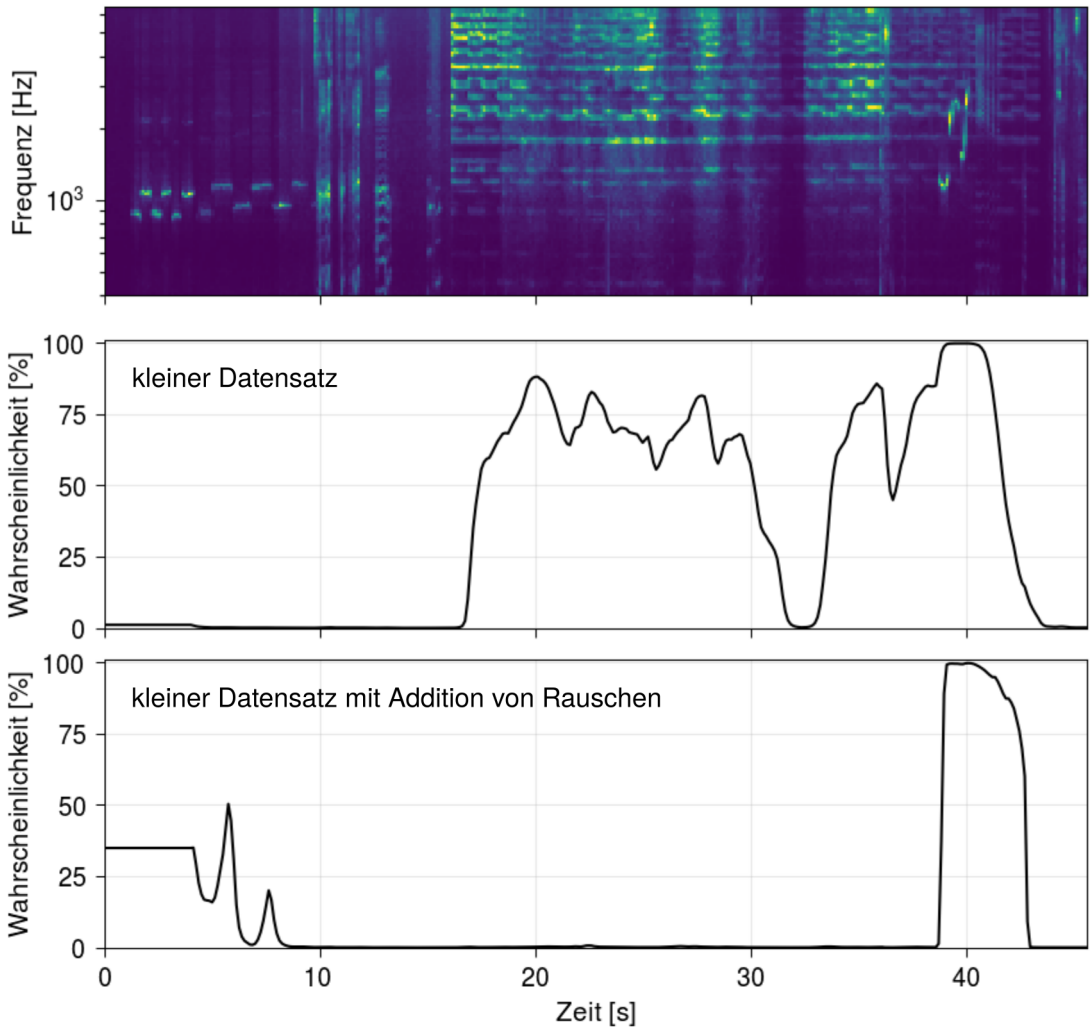


Bild 4.1: Netze trainiert ohne ähnliche Signale

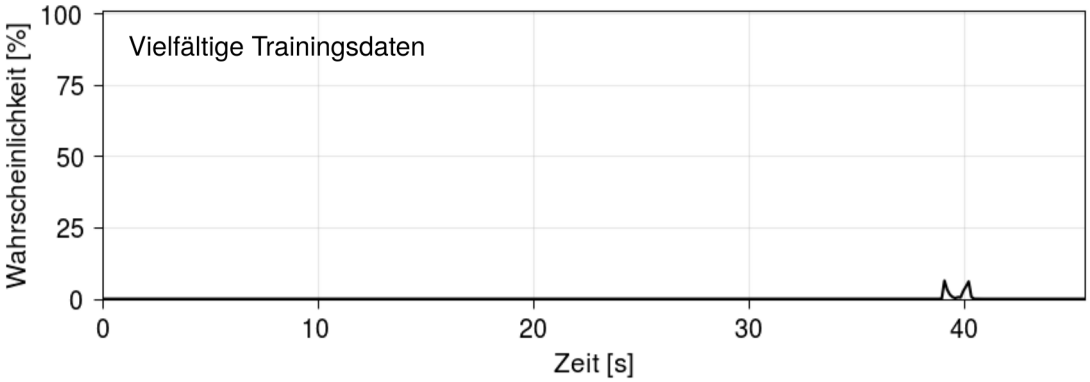


Bild 4.2: Netz trainiert mit ähnlichen Signalen

Score-Test.

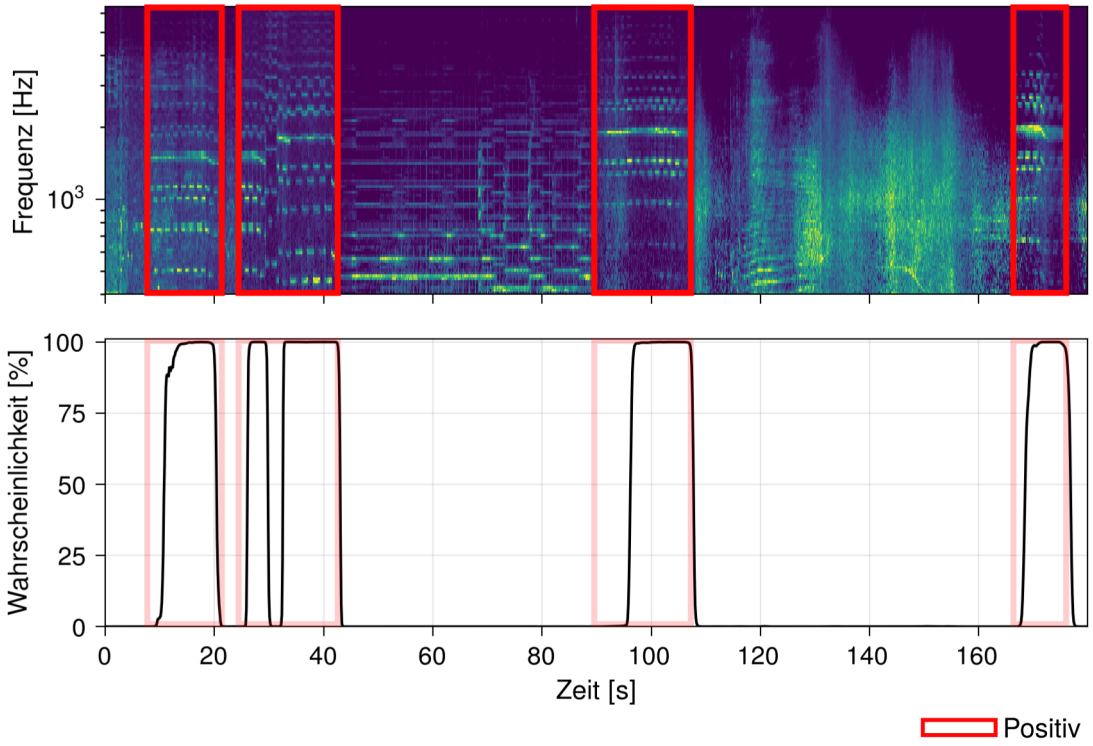


Bild 4.3: Ergebnis des ausgewählten Netzes für den Score-Test

4.3.1 Training des Neuronalen Netzes

Für das vorgestellte Netz sind die Verläufe von Loss, Accuracy, Validation Loss und Validation Accuracy in Bild 4.4 dargestellt. Das Netz wurde mit Early Stopping eingelernt. Nachdem der Validation Loss über fünf Epochen nicht gesunken ist, wurde das Training beendet. Den niedrigsten Validation Loss erreicht das Netz nach 8 Epochen. Folglich wurde für die Tests die Parametrisierung nach 8 Epochen verwendet. Die genauen Ergebnisse für dieses Netz können in Tabelle 4.3 nachvollzogen werden. Das Netz erreicht diese Ergebnisse bereits nach wenigen Epochen, da die trainierten Signale trotz unterschiedlicher Aufnahmen grundsätzlich ähnlich sind und durch die zusätzlich generierten Trainingsdaten in jeder Epoche sehr viele Aufnahmen trainiert werden. Bei dem betrachteten Netz sind es mit den gewählten Einstellungen in jeder Epoche 30680 kurze Aufnahme von Signaltönen.

4.3.2 Addition von Rauschen

Durch die Erweiterung des Trainingsdatensatzes durch Daten mit addiertem Rauschen wird die Sensitivität für verrauschte Aufnahmen von Signaltönen gesteigert. In Bild 4.5 ist das Reaktionsverhalten zweier identisch aufgebauter Netze aufgetragen. Das erste Netz wurde nur mit dem normalen Trainingsdatensatz trainiert, das zweite Netz

Einstellungen	Werte
Mischungsverhältnis für zusätzliche Positive mit Negativen Daten	0,3
Grundaddition von Rauschen	0,001
Durchläufe mit zusätzlichem Rauschen	3
Noise	100
Blockgröße	8192
Hop-Länge	4096
Fensterfunktion STFT	Hann
Verzerrung	Wurzel
Gate	0,05
Grundfrequenz	400 Hz
Anzahl der Oktaven	4
Stufen pro Oktave	24
Paket-Overlap	6

Tabelle 4.1: Einstellungen der Vorverarbeitung

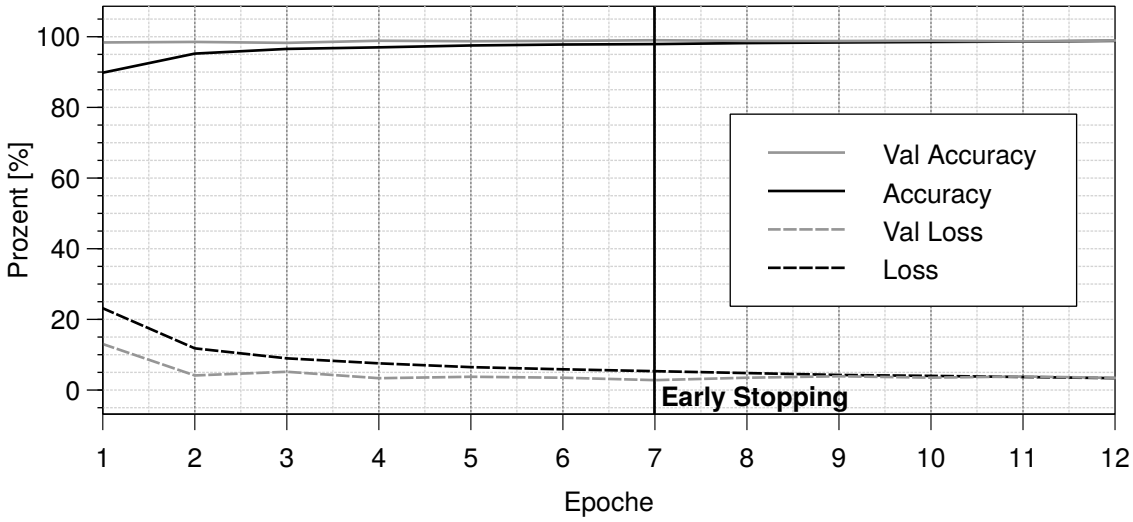


Bild 4.4: Verlauf von Accuracy, Loss, Validation Accuracy und Validation Loss

mit den zusätzlichen Daten. Das Netz, das mit Rauschen trainiert wurde, reagiert im Ramp-Test ungefähr ab einer Signal-Noise-Ratio (SNR) von 0 dB und erkennt es sicher ab einem Verhältnis von 15 dB. Das Netz ohne Addition von Rauschen beginnt erst bei einem Verhältnis von 6 dB zu reagieren und erkennt es sicher bei ca. 25 dB. Es erreicht aber nicht die Sicherheit des anderen Netzes. Ein weiter Effekt kann die verringerte

Einstellung	Wert
LSTM-1	256 Knoten
Recurrent Dropout	0,2
Dropout	0,2
LSTM-2	128 Knoten
Recurrent Dropout	0,1
Dropout	0,1
Dense	128 Knoten
Dropout	0,1
Activation	ReLU
Final Activation	Softmax
Loss Function	Sparse Categorical Entropy
Optimizer	Adam
Learn Rate	0,0001

Tabelle 4.2: Netzeinstellungen

Ergebnis	Wert	Ergebnis	Wert
Accuracy	0,988	Trainingssamples	61360
Loss	0,034	Validierungssamples	59336
Val Accuracy	0,989	Epochen	12
Val Loss	0,034	Early Stopping	5

Tabelle 4.3: Trainingsergebnisse

Störanfälligkeit durch die Anwendung von Rauschen auf Trainingsdaten sein. Vor allem bei kleineren Datensätzen kann die Anfälligkeit stark reduziert werden.

Ein geeigneter Maximalwert des Rauschens liegt zwischen 50% und 200% der maximalen Amplitude des Paketes. Die Wahl des Maximalwertes ist stark von der Vorverarbeitung abhängig. In Kombination mit einem zu hohen Gate kann das Netz teilweise nicht mehr trainiert werden, da die Trainingsdaten aus zu dünn besetzten Matrizen bestehen. Eine sinnvolle Anzahl der Durchläufe ist anhand des gewählten Maximalwertes festzulegen. Um mögliches Overfitting durch zu ähnliche Daten zu vermeiden, wurden die Maximalwerte in mindestens 20%-Schritten gewählt.

4.3.3 Overlap Kurzzeit-Fourier-Transformation

Die Hop-Länge der STFT wurde nach Empfehlung von *Zhivomirov* [35] mit Blockgröße/2 und Blockgröße/4 getestet. Durch diesen Schritt verringert sich der Zeitschritt pro Spalte der Matrix, gleichzeitig werden dadurch auch zweimal bzw. viermal mehr Daten

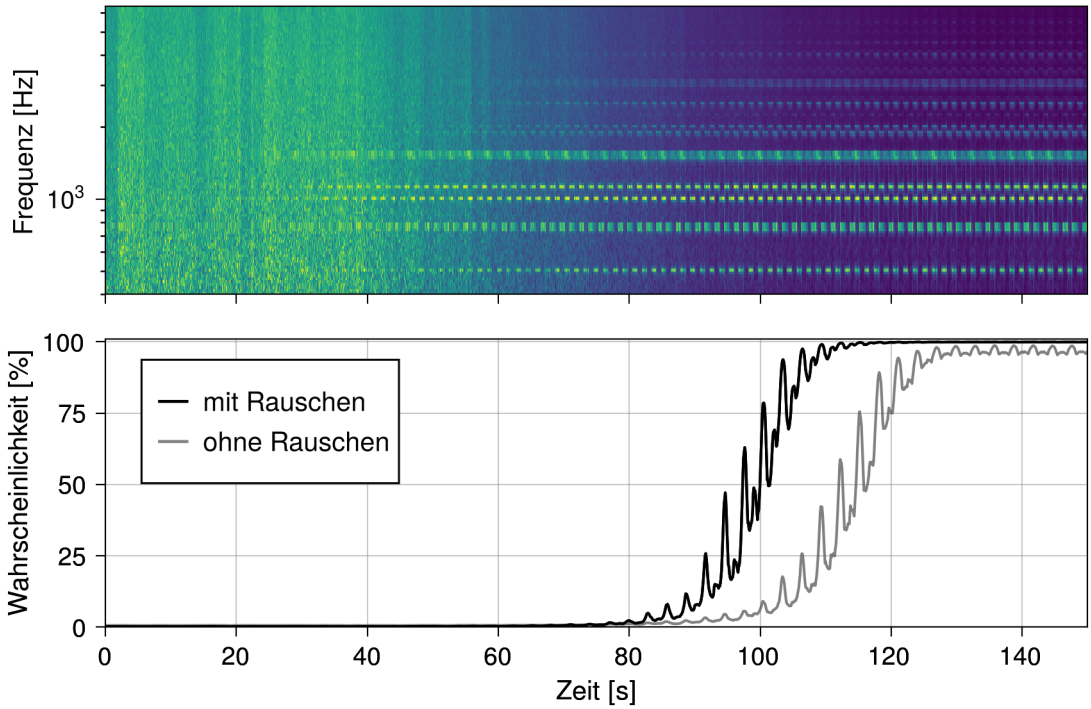


Bild 4.5: Ramp-Test für Netz mit und ohne addiertes Rauschen

generiert. Für einen Overlap der STFT der Größe Blockgröße/4 konnte keine bessere Erkennung festgestellt werden. Im Vergleich zwischen Hop-Länge gleich Blockgröße/2 und keinem Overlap (Hop-Länge = Blockgröße), kann mit Overlap eine minimal bessere Erkennung festgestellt werden.

4.3.4 Verzerrung der STFT-Matrix

Für die unterschiedlichen Verzerrungen der STFTs führt lediglich die Wurzelfunktion vereinzelt zu besseren Ergebnissen. Mit Hilfe dieser Funktion werden Details im mittleren Lautstärke-Bereich angehoben. Für die verwendeten Netze ließ sich allerdings kein messbarer Vorteil herausstellen. Netze mit und ohne Verwendung der Wurzelfunktion konnten vergleichbare Ergebnisse erzielen.

4.3.5 Gate auf die Werte der STFT

Durch die Verwendung eines Gates können Signaltöne herausgestellt werden. In Bild 4.7 ist sichtbar, wie sich ein Gate im Vergleich zum Netz in Bild 4.5 positiv auf das Filtern von Rauschen auswirken kann. Die Signaltöne sind deutlich erkennbar und leisere Störgeräusche werden teilweise vollständig herausgefiltert. Für die Erkennung im Ramp-Test reagiert das untersuchte Netz mit einem Gate von 0,05 bereits bei einer negativen SNR. Gegenüber dem Netz in Abb. 4.5 ohne Gate erkennt das Netz das Signal bereits bei der halben SNR.

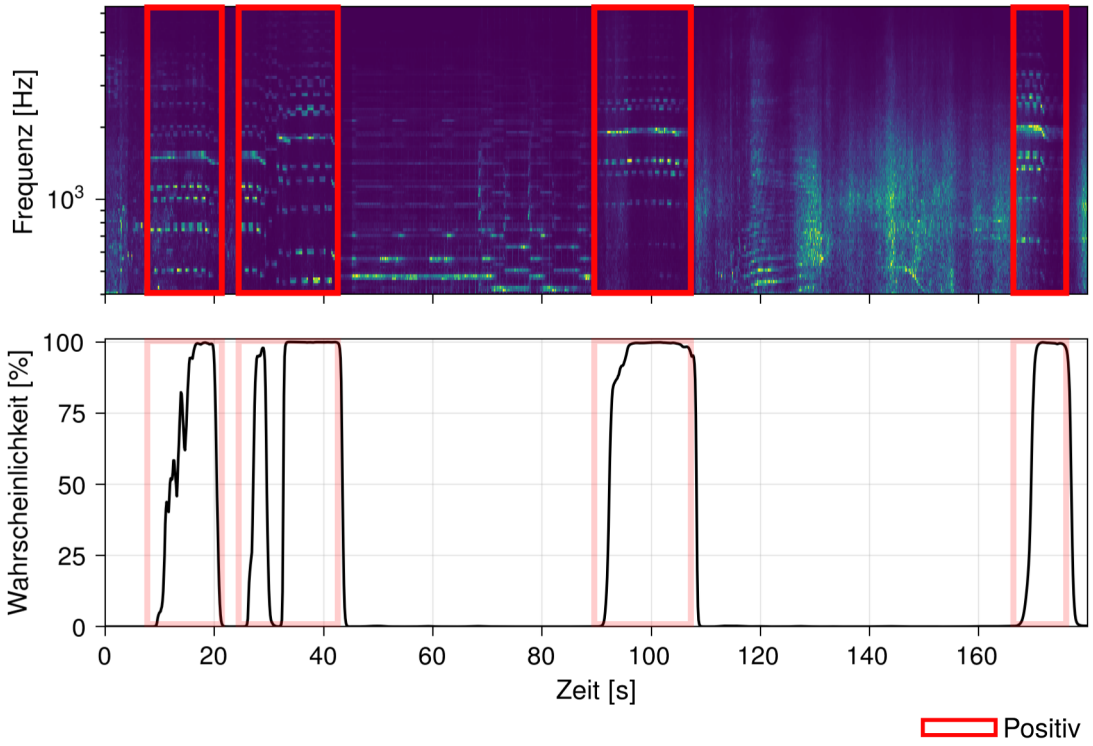


Bild 4.6: Netz ohne Verwendung der Wurzelfunktion

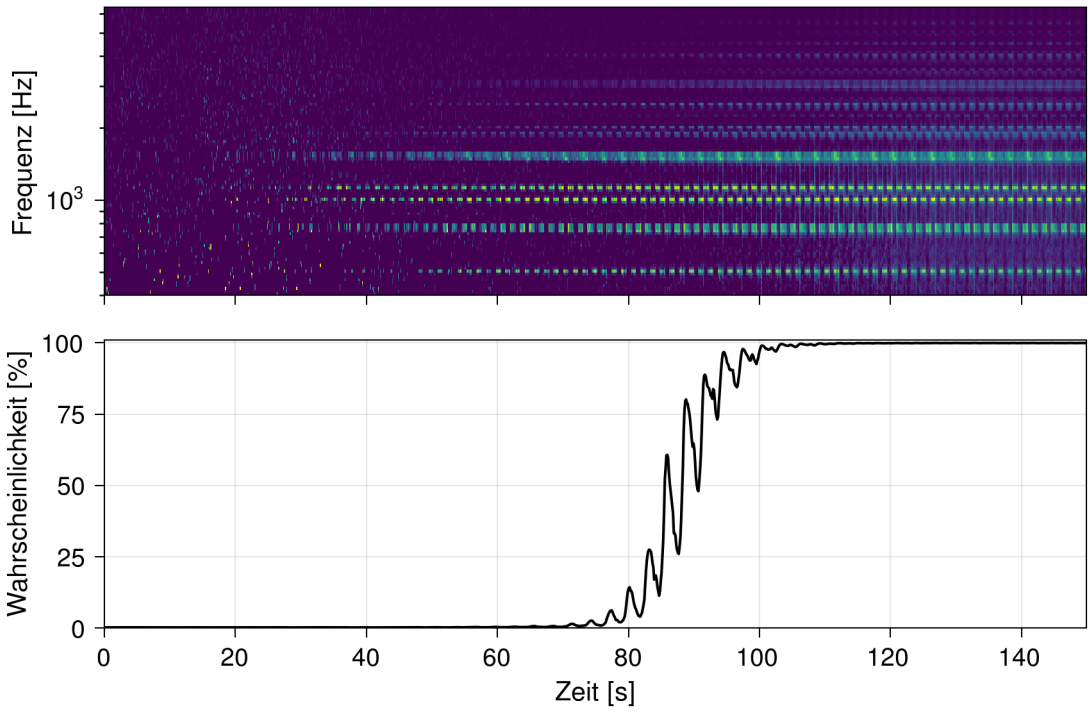


Bild 4.7: Netz mit Gate von 0,05

Die Wahl des richtigen Schwelle ist stark abhängig von der Art der Transformation, Verzerrung, Normalisierung und der Einlerndaten. Ein geeigneter Wert für das Ga-

te, in Verwendung mit der Verzerrung durch die Wurzelfunktion und einem additiven Rausch-Maximalwert von 100% liegt zwischen 0,01 und 0,15. Teilweise konnten Netze ohne Verzerrung und Rauschen ab einem Wert von über 0,15 keine Merkmale mehr erkennen und die Accuracy fiel auf 50%.

4.3.6 Zusammenfassung zu Frequenzbändern

Durch die Zusammenfassung der Frequenzbänder kann bei 12 Stufen pro Oktave die Datenmenge zum Training des Neuronalen Netzes um den Faktor 227,56 verringert werden. Dies reduziert die Berechnungsdauer der Weiterverarbeitung und den benötigten Arbeitsspeicherbedarf. Die Dauer eines Einlernvorgangs kann wesentlich beschleunigt werden. Es wurden Netze mit 12, 24 und 48 Stufen pro Oktave verglichen. Eine höhere Anzahl an Stufen als 24 hat zu keiner besseren Erkennung geführt. Im Vergleich zwischen einem Netz mit 12 und 24 Stufen erreicht das Netz mit 24 Stufen eine leicht höhere Genauigkeit.

4.3.7 Overlap Pakete

Ohne einen Overlap der Pakete werden die Einlerndaten nur mit bestimmten Startpunkten der Aufnahme eingelernt. Dies führt bei dem fertigen Modell zu einer Oszillation in der Erkennung, da nur Pakete, die mit dem tieferen Klang beginnen eindeutig erkannt werden. Durch den Overlap werden die Einlerndaten für verschiedene Startpunkte eingelernt. Dadurch konnte die Oszillation gemindert und die Generalisierungseigenschaften des Netzes verbessert werden. Beim Ramp-Test konnte ein identisches Netz mit Overlap den Signalton früher erkennen, als das ohne.

4.4 Neuronale Netze

Im folgenden Abschnitt werden mehrere Neuronale Netze zur Erkennung US-amerikanischer Signaltöne auf ihre Genauigkeit untersucht. Für alle Netze werden dieselben Trainingsdaten und Einstellungen verwendet. Es wird lediglich die Netzarchitektur variiert. In Tabelle 4.4 sind die getesteten Netze aufgelistet. Test Accuracy, FPR und TPR beziehen sich auf den Score-Test.

Zwischen den Netzen konnten im Laufe der Tests nur minimale Unterschiede festgestellt werden, die sich auf die Anzahl der Schichten zurückführen lassen. Dies wird in Tabelle 4.4 daran deutlich, dass unter den drei besten Netzen alle drei getesteten Layer-Anzahlen vertreten sind. Vorteil des Netzes mit zwei LSTM-Schichten ist die niedrigere Berechnungsdauer von 2 ms pro Paket, gegenüber 4 ms bei dem getesteten Ein-Layer-LSTM und 3 ms bei dem Drei-Layer-LSTM. Entscheidend für die

ID	LSTM1	LSTM2	LSTM3	Dense	TestAcc	TPR	FPR	ValAcc	ValLoss
1107	256	128	-	64	0,842	0,498	0,010	0,986	0,100
1110	512	-	-	192	0,831	0,447	0,005	0,984	0,147
1111	128	256	128	64	0,828	0,438	0,005	0,979	0,121
1108	256	64	-	64	0,811	0,373	0,002	0,984	0,082
1109	128	128	-	64	0,802	0,356	0,008	0,980	0,107
1116	256	64	-	32	0,795	0,329	0,005	0,973	0,090
1113	128	64	-	32	0,781	0,282	0,005	0,974	0,091
1112	128	128	-	32	0,767	0,227	0,002	0,973	0,100
1114	256	128	-	32	0,767	0,222	0	0,977	0,096
1115	128	64	-	64	0,764	0,220	0,003	0,978	0,090

Tabelle 4.4: Vergleich verschiedener Netzarchitekturen

Berechnungsdauer ist nicht die Anzahl der Schichten, sondern die Anzahl der Knoten. Hinsichtlich des Vergleichs von Ein- und Zwei-Layer-LSTM konnten bei dem getesteten Zwei-Layer-LSTM mit weniger Knoten minimal bessere Ergebnisse erzielt werden.

Im Hinblick auf die Knotenanzahl lässt sich keine allgemeingültige optimale Konfiguration bestimmen. Es wurden verschiedene Konfigurationen für das Zwei-Layer-LSTM getestet. Bei Betrachtung von Tabelle 4.4 fällt auf, dass bis auf das kleine Netz *ID*1115 die Netze mit der größeren Dense Layer positive Signale besser erkennen. Allgemein führten besonders die Zwei-Layer-LSTM mit 128 oder 256 Knoten in den LSTM-Schichten zu den besten Ergebnissen.

Auffällig ist die niedrige False Positive Rate aller Netze. Gleichzeitig ist True Positive Rate auch nicht sehr hoch. Als negative Daten sind im Score-Test Straßengeräusche und ein Musikstück enthalten. Diese werden von allen Netzen problemlos unterschieden. Die niedrige True Positive Rate lässt sich anhand von Bild 4.8 für Netz *ID*1107 erklären. Das Netz reagiert bei allen im Testdatensatz vorhandenen Signaltönen, aber bei unvollständigen Modulationen erst verspätet. Zusätzlich wird zur Berechnung der TPR ein positives Signal erst ab einer Wahrscheinlichkeit von 90% gewertet. Für Netz *ID*1107 können in den Tabellen 4.5 und 4.6 die verwendeten Einstellungen nachvollzogen werden.

4.4.1 Training des Neuronalen Netzes

Für das vorgestellte Netz sind die Verläufe von Loss, Accuracy, Validation Loss und Validation Accuracy in Bild 4.9 dargestellt. Das Netz wurde mit Early Stopping eingelernt. Nachdem der Validation Loss über fünf Epochen nicht gesunken ist, wurde das Training beendet. Den niedrigsten Validation Loss erreicht das Netz nach 11 Epochen. Folglich wurde für die Tests die Parametrisierung nach 11 Epochen verwendet. Die genauen Ergebnisse für dieses Netz können in Tabelle 4.7 nachvollzogen werden.

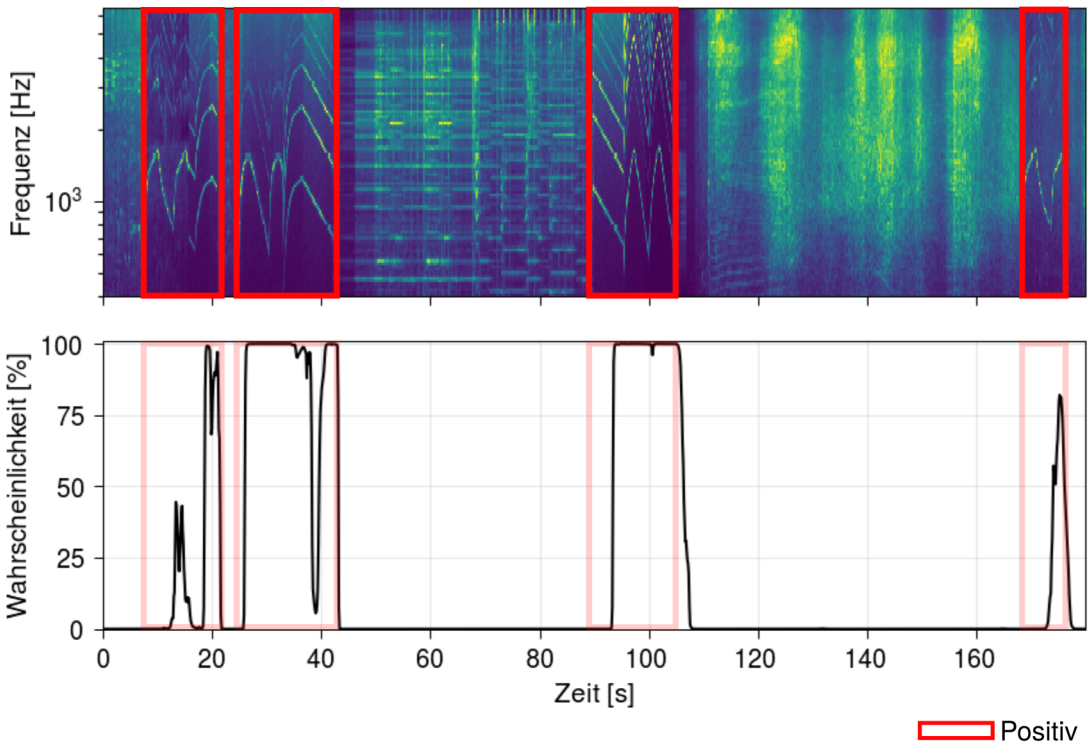


Bild 4.8: Ergebnis von Netz *ID*1107 für den Score-Test

Das Netz erreicht diese Ergebnisse bereits nach wenigen Epochen, da die trainierten Signale trotz unterschiedlicher Aufnahmen grundsätzlich ähnlich sind und durch die zusätzlich generierten Trainingsdaten in jeder Epoche viele Aufnahmen trainiert werden. Bei dem betrachteten Netz sind es mit den gewählten Einstellungen in jeder Epoche 91426 kurze Aufnahmen von Signaltönen.

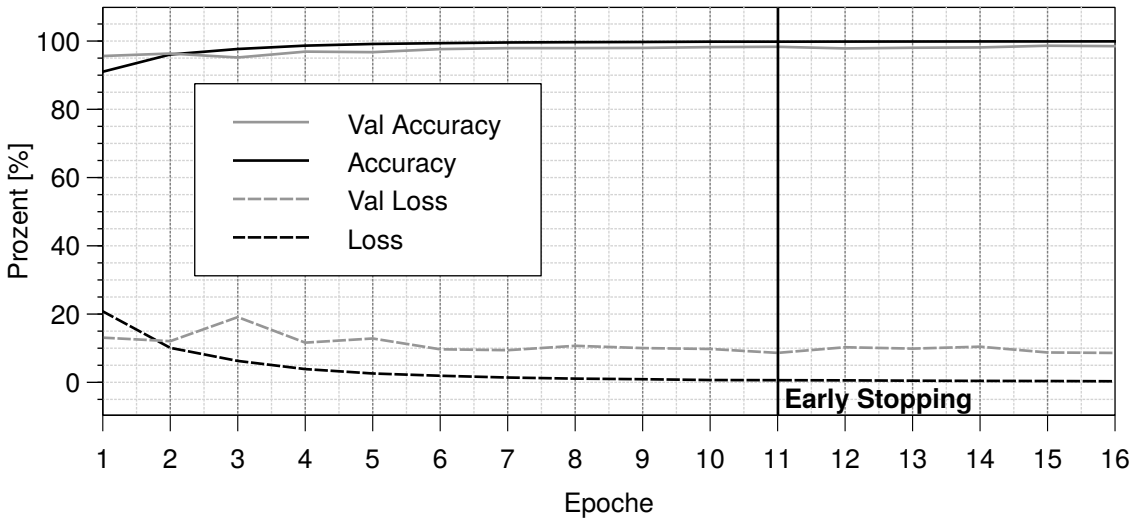


Bild 4.9: Verlauf von Accuracy, Loss, Validation Accuracy und Validation Loss

Einstellungen	Werte
Mischungsverhältnis für zusätzliche Positive mit Negativen Daten	0,3
Grundaddition von Rauschen	0,001
Durchläufe mit zusätzlichem Rauschen	1
Noise	100
Blockgröße	8192
Hop-Länge	4096
Fensterfunktion STFT	Hann
Verzerrung	Wurzel
Gate	0,001
Grundfrequenz	400 Hz
Anzahl der Oktaven	4
Stufen pro Oktave	24
Paket-Overlap	1

Tabelle 4.5: Einstellungen der Vorverarbeitung von Netz *ID*1107

4.5 Einordnung der Ergebnisse

Mit der in dieser Arbeit verwendeten Methodik konnte ein Ereignisdetektor entwickelt werden, der in einem festgelegten Rahmen in der Lage ist, Signaltöne im Straßenverkehr von anderen Geräuschen zu unterscheiden. Die Ergebnisse vorangegangener Forschung werden im Grundsatz bestätigt.

Die hier beschriebenen trainierten Neuronalen Netzwerte sind im Endergebnis gut geeignet Signaltöne im Straßenverkehr zu erkennen und ähnliche zu differenzieren. Optimierungspotenzial bietet die Erkennung von Signaltönen in einer Umgebung mit lauten Störgeräuschen. Mit Hilfe des Gates konnten stellenweise auch stark verrauschte Aufnahmen mit einer negativen Signal-Noise-Ratio richtig erkannt werden. Allerdings ist dies nur für einzelne Aufnahmen gelungen und nicht weiter in der nötigen Breite untersucht worden.

4.5.1 Vergleich der Netze für Signaltöne in Deutschland und in den Vereinigten Staaten

Es wurde für beide Länder ein breites Spektrum an ein Einstellungen und verschiedene Netzarchitekturen getestet. Es konnten keine wesentlichen Unterschiede in der

Einstellung	Wert
LSTM-1	256 Knoten
Recurrent Dropout	0,2
Dropout	0,2
LSTM-2	128 Knoten
Recurrent Dropout	0,1
Dropout	0,1
Dense	64 Knoten
Dropout	0,1
Activation	ReLU
Final Activation	Softmax
Loss Function	Sparse Categorical Entropy
Optimizer	Adam
Learn Rate	0,0001

Tabelle 4.6: Netzeinstellungen von Netz *ID*1107

Ergebnis	Wert	Ergebnis	Wert
Accuracy	0,998	Trainingssamples	182852
Loss	0,006	Validierungssamples	119002
Val Accuracy	0,983	Epochen	11
Val Loss	0,086	Early Stopping	5

Tabelle 4.7: Trainingsergebnisse von Netz *ID*1107

Vorverarbeitung und der Netzarchitektur für Netze zur Erkennung deutscher und US-Signaltöne im Straßenverkehr festgestellt werden. Für beide Signaltonarten führten die gleichen Einstellungen zu guten Ergebnissen. Der einzige Einstellungsparameter, der grundsätzlich anders gewählt wurde, ist die Paketgröße. Der Signalzyklus eines deutschen Signaltons dauert zwischen 1,25 und 1,75 s, für die Dauer eines Pakets haben Werte zwischen 2 und 3 s zu den besten Ergebnissen geführt. Für US-Signaltöne kann die Dauer eines Signalzyklus des Wail-Signals zwischen 2 und 6 s betragen, daher wurde hier eine größere Paketgröße gewählt. Für die finalen Netze zur Erkennung der US-Signaltöne führten 4 s lange Pakete zu den besten Ergebnissen.

4.5.2 Ausblick

Um dem Problem fälschlich erkannter akustischer Ereignisse zu begegnen, sollte der entwickelte Detektor zunächst dazu genutzt werden, weitere Trainingsdaten zu generieren. Dazu sollten mit diesem Detektor längere Aufnahmen verarbeitet werden und

anschließend die Ergebnisse auf positive und falsch interpretierte Signale bewertet werden. Mit den entsprechend annotierten Aufnahmen könnte der Detektor neu oder weiter trainiert werden.

Im Spektrogramm der Signaltöne sind teils klare Muster zu erkennen, bevor das entwickelte Neuronale Netz reagiert. Durch eine Kombination aus CNN und LSTM könnten diese früher erkannt werden.

Für die Anwendung in einem Fahrzeug sollte, aufgrund der Fahrgeräusche als Störquelle, getestet werden, ob der entwickelte Detektor unter diesen Umständen die Signaltöne weiterhin zutreffend detektiert. Gegebenenfalls müsste das Neuronale Netz mit Daten trainiert werden, die von einem Fahrzeug aus aufgenommen wurden oder die dies simulieren.

Generell ist nach Erkenntnissen dieser Arbeit davon auszugehen, dass die automatisierte akustische Erkennung von Signaltönen in Fahrzeugen mit Hilfe Neuronaler Netze implementiert werden kann. Der hier vorgeschlagene Weg sollte Gegenstand ergänzender Arbeiten sein.

5 Zusammenfassung

In der vorliegenden Arbeit werden im ersten Schritt die Merkmale in Deutschland und in den USA im Straßenverkehr verwendeter Signaltöne erfasst. Basierend auf diesen Merkmalen wird ein Datensatz zum Training eines Neuronalen Netzes erstellt und eine Methodik zur Vorverarbeitung der Trainingsdaten für den Eingang eines Neuronales Netz implementiert. Ferner wird ein Neuronales Netz zur Erkennung der Signaltöne entwickelt. Abschließend wird das Verfahren und die Funktionalität des finalen Ereignisdetektors bewertet.

Die Merkmale der Signaltöne werden anhand der geltenden Norm, der Spektrogramme von Signaltonaufnahmen und der über einen Datensatz von Signaltonaufnahmen berechneten Energiedichte ermittelt. Der Datensatz wird mit Audiospuren von Youtube-Videos erstellt. Die Aufnahmen werden in positive Daten (mit Signalton) und negative Daten (ohne Signalton) aufgeteilt und annotiert. Die Vorverarbeitung erfolgt hauptsächlich durch die STFT der Daten. Zur Datenreduktion und Herausstellung tonaler Zusammenhänge werden die Ergebnisse der Transformation in Frequenzbändern der Gleichstufigen Stimmung entsprechend zusammengefasst und Frequenzen außerhalb der größten Energiedichte des Signals abgeschnitten.

Anhand einer Literaturrecherche wird das LSTM-Netz (Long Short-Term Memory) als geeigneter Algorithmus zur Erkennung von Signaltönen ermittelt und implementiert. Unter Verwendung der Trainingsdaten und einer Daten-Vorverarbeitung wird das Neuronale Netz trainiert und mit einem selektierten Datensatz getestet. Die Einstellungen des Netzes werden auf Basis der Ergebnisse angepasst, um die Erkennung zu optimieren.

Das im Rahmen dieser Arbeit entwickelte Neuronale Netz kann deutsche und US-amerikanische im Straßenverkehr verwendete Signaltöne von Straßengeräuschen und dem Signalton ähnlichen Geräuschen unterscheiden. Im Training erreicht das Netz für deutsche Signaltöne eine Validation Accuracy von 98,9%. Für den selbst erstellten Testdatensatz erreicht das Netz eine Test Accuracy von 89,7% bei einer TPR von 72,3% und einer FPR von 2,4%.

Das Neuronale Netz zur Erkennung US-amerikanischer erreicht im Training eine Validation Accuracy von 98,3%. Für den selbst erstellten Testdatensatz erreicht das Netz eine Test Accuracy von 84,2% bei einer TPR von 49,8% und einer FPR von 1,0%.

Es kann abschließend nicht ausgeschlossen werden, dass durch Anwendung des De-

tektors Geräusche ermittelt werden, die fälschlicherweise als Signalton erkannt werden. Diese Geräusche können in einer Weiterentwicklung dieser Methode zum Training des Neuronalen Netzes verwendet werden. Wünschend wäre im nächsten Schritt, den entwickelten Detektor in einem Fahrzeug zu testen und den Trainingsdatensatz gegebenenfalls zu erweitern.

6 Appendix

6.1 Beispiel Transformationsmatrix

Neue Frequenzen [Hz]	400	424	449	476	504
Frequenzen STFT [Hz]					
376	0	0	0	0	0
380	0,112	0	0	0	0
384	0,293	0	0	0	0
388	0,473	0	0	0	0
392	0,650	0	0	0	0
396	0,826	0	0	0	0
400	1	0	0	0	0
404	0,828	0,172	0	0	0
408	0,657	0,343	0	0	0
412	0,488	0,502	0	0	0
416	0,321	0,679	0	0	0
420	0,155	0,845		0	0
424	0	1	0	0	0
428	0	0,837	0,163	0	0
432	0	0,676	0,324	0	0
440	0	0,517	0,483	0	0
444	0	0,359	0,641	0	0
448	0	0,202	0,798	0	0
452	0	0	0,961	0,039	0
456	0	0	0,885	0,115	0
460	0	0	0,732	0,268	0
464	0	0	0,581	0,419	0
468	0	0	0,431	0,569	0
472	0	0	0,282	0,718	0
476	0	0	0	1	0
480	0	0	0	0,855	0,145
484	0	0	0	0,712	0,288
488	0	0	0	0,569	0,431
492	0	0	0	0,428	0,572
496	0	0	0	0,288	0,712
500	0	0	0	0,148	0,852
504	0	0	0	0	1

Tabelle 6.1: Auschnitt aus einer Transformationsmatrix von 5 Frequenzbändern

6.2 Verwendete Python-Programmbibliotheken

Package	Version	Kurzbeschreibung
python	3.7	Programmiersprache
h5py	2.10.0	Python-Interface für das HDF5-Dateiformat
librosa	0.7.2	Programmbibliothek zur Musik- und Audioanalyse
matplotlib	3.2.1	Programmbibliothek für mathematische Darstellungen
numpy	1.18.2	Numerische Bibliothek für wissenschaftliches Rechnen
pandas	1.0.3	Programmbibliothek für die Verwaltung von Daten und deren Analyse
pyaudio	0.2.11	Verbindung der PortAudio Bibliothek mit Python
ray	0.8.2	Framework für Verteilte Anwendungen
scipy	1.4.1	Programmbibliothek mit numerischen Algorithmen und mathematischen Werkzeugen
soundfile	0.10.3.post1	Programmbibliothek zum Auslesen und Schreiben von Audiodateien
sqlite3	2.8.3	DB-API 2.0 Interface für SQLite-Datenbanken
tensorflow	2.1.0	Framework zur datenstromorientierten Programmierung
youtube-dl	2020.3.8	Bibliothek zum Download von YouTube-Videos

Tabelle 6.2: Verwendete Python-Programmbibliotheken

Literaturverzeichnis

[1] Krump, G., 2012. *Akustische messmöglichkeiten mit smartphones*, In: DAGA 2012 - Darmstadt

[2] Stotz, D., 2019. *Computergestützte Audio- und Videotechnik*, Springer Vieweg, Wiesbaden

[3] National Instruments, 2019. *Understanding FFTs and Windowing*, tech. rep.

[4] Smith, J. O., 2011. *Spectral Audio Signal Processing*, W3K Publishing

[5] James, A. P., 2020. *Deep Learning Classifiers with Memristive Networks*, vol. 14, Springer, Berlin

[6] Li, F.F. und J. Johnson und S. Yeung, 2019. *Module 2: Convolutional Neural Networks*, lecture notes, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University

[7] Goodfellow, I. und Y. Bengio und A. Courville, 2016. *Deep Learning*, MIT Press, Cambridge (MA), `http://www.deeplearningbook.org`

[8] Gers, F. und J. Schmidhuber und F. Cummins, 1999. *Learning to forget: Continual Prediction with LSTM*, In: Neural Computation, 12, S. 2451–2471

[9] Sun, Y. et al., 2019. *Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues*, In: IEEE Communications Surveys & Tutorials, 21, S. 3072–3108

[10] Deutsches Institut für Normung, 2009. *DIN14610 - Akustische Warneinrichtungen für Bevorrechtigte Wegebenutzer*, Norm

[11] Tran, V.-T. und Y.-C. Yan und W.-H. Tsai, 2018. *Detection of Ambulance and Fire Truck Siren Sounds*, In: 51st Research World International Conference, Hanoi, S. 9–14

[12] Xia, X. et al., 2019. *A Survey: Neural Network-Based Deep Learning for Acoustic Event Detection*, In: Circuits, Systems, and Signal Processing, 38, S. 3433–3453

[13] Lerch, R. und G. Sessler und D. Wolf, 2009. *Technische Akustik - Grundlagen und Anwendungen*, Springer, Berlin

[14] Guicking, D., 2016. *Schwingungen*, Springer Vieweg, Wiesbaden

[15] Möser, M., 2015. *Technische Akustik*, Springer Vieweg, Wiesbaden

[16] Günther, H., 1996. *Grenzgeschwindigkeiten und ihre Paradoxa*, In: TEUBNER-Texte zur Physik, Springer Fachmedien, Wiesbaden

[17] Massenkeil, G. (Hrsg.), 1998. *Metzler Sachlexikon Musik*, J.B. Metzler, Stuttgart

[18] Verhulst, R., 2019. *Im Banne der Mathematik*, Springer Spektrum, Berlin

[19] Dickreiter, M., 1997. *Handbuch der Tonstudiotechnik Band 1*, vol. 1, K.G. Saur, München

[20] Braun, M. und V. Chaloupka, 2005. *Carbamazepine induced Pitch Shift and Octave Space Representation*, In: Hearing Research

[21] Vorländer, M., 2016. *Akustische Messtechnik*, Springer, Berlin

[22] Kruse Brandão, T. und G.Wolfram, 2018. *Digital Connection*, Springer Gabler, Wiesbaden

[23] Büttgenbach, S., 2016. *Mikrosystemtechnik*, Springer, Berlin

[24] Goyal, V. und H. Heller, 2013. *MEMS-Mikrofone - Richtungsweisende Innovation in der Schall-Erfassung*, In: Elektronik Praxis - Fachjournal für Elektronik Professionals

[25] Friedrich, H. J., 2008. *Tontechnik für Mediengestalter*, Springer, Berlin

[26] Vorländer, M., 2018. *Digitale Signalverarbeitung in der Messtechnik*, Springer Vieweg, Wiesbaden

[27] Lu, L. und A. Hanjalic, 2016. *Audio*, Springer New York, New York, S. 1–3

[28] Fischer, W., 2016. *Digitale Fernseh- und Hörfunktechnik in Theorie und Praxis*, Springer Vieweg, Wiesbaden

[29] Diverse, 1998. *Spektrum*, In: Lexikon der Physik, Spektrum Akademischer Verlag, Heidelberg

[30] Diverse, 1998. *Klangspektrum*, In: Lexikon der Physik, Spektrum Akademischer Verlag, Heidelberg

[31] Hoffmann, R. and M. Wolff, 2014. *Intelligente Signalverarbeitung 1: Signalanalyse*, Springer Vieweg, Wiesbaden

[32] Ohm, J. und H. D. Lüke, 2014. *Signalübertragung - Grundlagen der digitalen und analogen Nachrichtenübertragungssysteme*, Springer Vieweg, Wiesbaden

[33] Petersen, C. und H.Werkle, 2017. *Dynamik der Baukonstruktionen*, Springer Vieweg, Wiesbaden

[34] Mertins, A., 2010. *Signaltheorie*, Vieweg+Teubner, Wiesbaden

[35] Zhivomirov, H., 2019. *On the Development of STFT-Analysis and ISTFT-Synthesis Routines and their Practical Implementation*, In: TEM Journal, 8, S. 56–64

[36] Awad, M. und R. Khanna, 2015. *Efficient Learning Machines*, Apress, New York

[37] Salehinejad, H. et al., 2018. *Recent Advances in Recurrent Neural Networks*, arXiv:1801.01078v3 [cs.NE]

[38] Han, J. und C. Moraga, 1995. *The Influence of the Sigmoid Function Parameters on the Speed of Backpropagation Learning*, In: Lecture Notes in Computer Science, vol. 930, Springer, Berlin, S. 195–201

[39] Greff, K. et al., 2016. *LSTM: A Search Space Odyssey*, IEEE Transactions on Neural Networks and Learning Systems, 28, S. 2222

[40] Glorot, X. und A. Bordes und Y. Bengio, 2011. *Deep Sparse Rectifier Neural Networks*, In: 14th International Conference on Artificial Intelligence and Statistics, 15, S. 315–323

[41] Kruse, R. et al., 2015. *Computational Intelligence*, Springer Vieweg, Wiesbaden

[42] Tensorflow, Dokumentation, 2020. *Class Dense*, `https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense`, Zugriff: 29.03.2020

[43] Srivastava, N. et al., 2014. *Dropout: A Simple Way to prevent Neural Networks from Overfitting*, In: Journal of Machine Learning Research, 15, S. 1929–1958

[44] Ioffe, S. und C. Szegedy, 2015. *Batch normalization: Accelerating deep network training byreducing internal covariate shift*, In: 32nd International Conference on International Conference on Machine Learning, vol. 37, S. 448-–456

[45] Tensorflow, Dokumentation, 2020. *CategoricalCrossentropy*, `https://www.tensorflow.org/api_docs/python/tf/keras/losses/CategoricalCrossentropy`, Zugriff: 29.03.2020

[46] Ertel, W., 2016. *Grundkurs Künstliche Intelligenz*, Springer Vieweg, Wiesbaden

[47] Dörn, S., 2018. *Programmieren für Ingenieure und Naturwissenschaftler*, Springer Vieweg, Wiesbaden

[48] Choi, K. et al., *Convolutional Recurrent Neural Networks for Music Classification*, 2017. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, S. 2392–2396

[49] Kolb, L., 2016. *Akustische Wahrnehmung von Fahrzeugen mit Sondersignal im internationalen Vergleich Deutschland - USA*, Masterarbeit Brandschutzingenieurwesen, TH Köln

[50] SAE International, 2012. *SAE J1849: Emergency Vehicle Sirens*, Norm

[51] Fragoulis, D. K. und J. N. Avaritsiotis, 2001. *A Siren Detection System based on Mechanical Resonant Filters*, In: Sensors, 1, S. 121–137

[52] Ellis, D. P. W., 2001. *Detecting Alarm Sounds*, In: Consistent & Reliable Acoustic Cues for Sound Analysis: One-day Workshop, 1, S. 59–62

[53] Schröder, J. et al., 2013. *Automatic Acoustic Siren Detection in Traffic Noise by Part-Based Models*, In :2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), S. 493–497

[54] Sunitha, C. und E. Chandra, 2015. *Speaker Recognition using Cepstral Coefficient and Machine Learning Technique*, In: Research Journal of Applied Sciences, Engineering and Technology, 11, S. 983–987

[55] Yücesoy, E. und V. Nabiyev, 2013. *Gender Identification of a Speaker using MFCC and GMM*, In: 8th International Conference on Electrical and Electronics Engineering (ELECO), 11, S. 626–629

[56] Beritelli, F. et al., 2006. *An Automatic Emergency Signal Recognition System for the Hearing Impaired*, In: IEEE 12th Digital Signal Processing Workshop & 4th IEEE Signal Processing Education Workshop, Teton National Park, WY, S. 179–182

[57] Wang, J. und S. Li, 2018. *Comparing the Influence of Depth and Width of Deep Neural Network based on Fixed Number of Parameters for Audio Event Detection*, In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Calgary, S. 2681–2685

[58] HDF Group, 2017. *Hdf5 sell sheet*, `https://www.hdfgroup.org/solutions/hdf5/`, Zugriff: 29.03.2020

[59] Ableton Live Hersteller-Website. `https://www.ableton.com/live/`, Zugriff: 29.03.2020

[60] Chen, H. und R. Gururajan, 2012. *Otsu's Threshold Selection Method Applied in De-noising Heart Sound of the Digital Stethoscope Record*, In: Advances in Information Technology and Industry Applications, Springer, Berlin, S. 239–244

[61] Schörkhuber, C. und A. Klapuri, 2010. *Constant-q transform toolbox for music processing*, In: 7th Sound and Music Computing Conference, Barcelona

[62] Sak, Hasim und A. Senior und F. Beaufays, 2014. *Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling*, In: INTERSPEECH 2014, S. 338–342

[63] Kingma, D. P. und J. Ba, *Adam: A Method for Stochastic Optimization*, 2015. In: 3rd International Conference for Learning Representations, San Diego

[64] Li, F.F. und J. Johnson und S. Yeung, 2019. *Module 1: Neural Networks*, lecture notes, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University