



Voice Cloning

Seminararbeit

von Alexander Melde

Bearbeitungszeitraum 09.10.2019 – 12.12.2019

Abgabedatum 12.12.2019

Studiengang M. Sc. Informatik

Hochschule Hochschule Karlsruhe – Technik und Wirtschaft

Matrikelnummer 64749

Betreuer Dozent Prof. Dr.-Ing. Matthias Wölfel

Eidesstattliche Erklärung

Ich, Alexander Melde, versichere hiermit, dass ich die vorliegende Arbeit mit dem Titel „Voice Cloning“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Falls sowohl eine gedruckte als auch elektronische Fassung abgegeben wurde, versichere ich zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Ort, Datum

Unterschrift

Abstract

This paper describes techniques on cloning voices using artificial intelligence. There is an explanation of the foundation techniques, a review of the state of the art in science and enterprise products and a comparison of pros and cons of voice cloning and a mention of use cases.

Keywords

Natural Language Processing (NLP), voice synthesis, speech recognition, speech synthesis, digital voice avatar, clone voice

Kurzbeschreibung

In dieser Seminararbeit geht es um das Thema Voice Cloning. Der Begriff Voice Cloning bezeichnet den Vorgang, bei dem menschliche Stimmen digital abgebildet werden. Nach dem „Klonen“ ist es möglich, beliebige Sätze durch die geklonten Stimme sprechen zu lassen.

In dieser Arbeit werden nicht nur die technischen Hintergründe der genannten Techniken sowie mögliche Chancen und Risiken des Voice Clonings erläutert, sondern auch der aktuellen Stand der Wissenschaft sowie die Praxistauglichkeit dieser Ansätze mithilfe aktueller Papers und deren Code-Implementationen erarbeitet und getestet.

Stichwörter

Natürliche Sprachverarbeitung, Verarbeitung natürlicher Sprache, Stimmen klonen, Sprachsynthese

Inhaltsverzeichnis

1 Einleitung	1
1.1 Umfeld der Arbeit	1
1.2 Motivation und Problemstellung	1
1.3 These	2
1.4 Ziele der Arbeit	2
1.5 Vorgehensweise	2
2 Grundlagen	3
2.1 Digitale Sprachverarbeitung	3
2.1.1 Methoden zur Sprachsynthese	4
2.2 Machine Learning (ML)	6
2.2.1 Lernmethoden	6
2.2.2 Klassifikation	7
2.2.3 Künstliche neuronale Netze	8
2.2.4 Convolutional Neural Networks (CNNs)	11
2.2.5 Recurrent Neural Networks (RNNs)	13
2.3 Voice Cloning	15
3 Chancen und Risiken des Voice Clonings	16
3.1 Chancen	16
3.2 Risiken	17
3.3 Rechtslage	17
4 Untersuchung bestehender Ansätze und Implementationen	18
4.1 Forschung	18
4.1.1 WaveNet (2016)	18
4.1.2 Baidu DeepVoice (2017)	19
4.1.3 Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis (2018)	20
4.1.4 MelNet (2019)	24
4.2 Kommerzielle Angebote	25
4.2.1 Adobe Project VoCo	25
4.2.2 Descript Overdub	26
4.3 Vergleich	28
5 Fazit	29

Abkürzungsverzeichnis

3D	Dreidimensional	
Abb.	Abbildung	3
BRNN	Bidirectional RNN	13
CNN	Convolutional Neural Network	11
dB	Dezibel	3
Hz	Hertz	3
KI	Künstliche Intelligenz	6
LSTM	Long Short-Term Memory	14
ML	Machine Learning	6
NLP	Natural Language Processing	III
ReLU	Rectified Linear Unit	10
RNN	Recurrent Neural Network	13
SVM	Support Vector Machine	7
venv	Virtual Environment	
VODER	Voice Operation DEmonstratoR	4

Abbildungsverzeichnis

2.1	Darstellung eines Audiosignals als Wellenform und Spektrogramm	3
2.2	Umwandlung von gesprochener Sprache zu Text	4
2.3	Umwandlung von Text zu gesprochener Sprache	4
2.4	Mechanische und Elektronische Verfahren zur Sprachsynthese	5
2.5	SVM zur Klassifikation von sportlichen Personen.	7
2.6	Minimales neuronales Netz	8
2.7	Aus drei Schichten bestehendes neuronales Netz	9
2.8	Verschiedene Aktivierungsfunktionen für künstliche neuronale Netze	10
2.9	Convolutional Neural Network (CNN)	11
2.10	Beispielhaftes CNN zur Bildklassifikation	12
2.11	Darstellung eines klassischen neuronalen Netzes als RNN	13
2.12	Verschiedene Typen von RNN	14
4.1	Architekturen der Ansätze von Baidu DeepVoice	19
4.2	Architektur des Ansatzes von Jia et al. 2018	20
4.3	Synthese des gleichen Texts für verschiedene Stimmen	21
4.4	Screenshot der Voice Cloning Toolbox	23
4.5	Darstellung eines Audiosignals bei MelNet als Wellenform und Spektrogramm . .	24
4.6	Ersetzen von Wörtern mit Adobe VoCo	25
4.7	Grafische Oberfläche von Adobe VoCo	25
4.8	Grafische Oberfläche von Descript	26
4.9	Ersetzen von Wörtern mit Descript Overdub	27

1 Einleitung

In diesem Kapitel werden zur Orientierung das Projektumfeld, die Motivation sowie die Problemstellung beschrieben, eine These und verschiedene Ziele der Arbeit aufgestellt und die zum Erreichen dieser Ziele verwendete Vorgehensweise beschrieben.

1.1 Umfeld der Arbeit

Diese Arbeit stellt die schriftliche Dokumentation der Lehrveranstaltung *Seminar* an der Hochschule für Technik und Wirtschaft in Karlsruhe dar. Diese ist für Studierende des Masterstudiengangs Informatik fester Teil des Modulplans.

Der Autor dieser Arbeit, Alexander Melde, hat die Studiengangs-Vertiefung *Maschinelles Lernen* gewählt und das Thema dieser Arbeit auf eigenen Wunsch dem betreuenden Professor zur Prüfung vorgeschlagen.

Der betreuende Professor Dr.-Ing. Matthias Wölfel ist Dozent für wahrnehmungsbasierte Interaktion und Benutzerschnittstellen an der Hochschule für Technik und Wirtschaft in Karlsruhe. Zu seinen Forschungsgebieten zählen Spracherkennung und -verarbeitung, Interaktionsdesign, Mensch-Computer-Mensch Interaktion, Künstliche Intelligenz, Augmented- und Virtual Reality, Digitalkultur sowie Kunst- und Medien-Theorie (Wölfel 2019).

1.2 Motivation und Problemstellung

In den letzten Wochen und Monaten kam es vermehrt zu Publikationen (Jemine 2019a; Vasquez und Lewis 2019; Jia et al. 2018) und Nachrichtenmeldungen (Brown 2019) im Bereich der Text-to-Speech Synthese mithilfe moderner Machine Learning Ansätze.

Besonders dominant ist hierbei das Thema Voice Cloning. Darunter versteht man das Lernen von sprechtypischen Details anhand kurzer Beispielsätze, um den Computer anschließend mit der Betonung dieses Sprechers reden zu lassen.

Diese Idee ist nicht neu, denn selbst die Roboter aus dem Film „Terminator 2“ aus dem Jahr 1991 konnten menschliche Stimmen bereits perfekt imitieren (IMDb 2019). Damals war das aber noch Zukunftstechnologie. Heute ist vermutlich einiges mehr möglich, weshalb in dieser Arbeit geprüft werden soll, was der aktuelle Stand der Technik ist.

Diese Verfahren sind nicht nur aus technischer Sicht interessant, sondern ermöglichen auch zahlreiche praktische Anwendungsfälle. Aufgrund der zunehmend einfachen Zugänglichkeit dieser Technologie (Jemine 2019b) kam es aber bereits auch schon zu ersten Missbrauchs- und Betrugsfällen (Brown 2019).

1.3 These

In dieser Arbeit soll die folgende These geprüft und entweder bestätigt oder widerlegt werden:

Voice Cloning ist möglich und bereits heute eine Gefahr.

1.4 Ziele der Arbeit

In dieser Arbeit soll untersucht werden, in welchem Umfang Voice Cloning bereits heute möglich ist und ob die Qualität bisheriger Ansätze ausreicht, um die Technologie missbräuchlich zu verwenden.

1.5 Vorgehensweise

Am Anfang dieser Arbeit werden die technischen Hintergründe der Digitalen Sprachverarbeitung und einzelne Ansätze des Bereichs Künstliche Intelligenz grundlegend erläutert.

Der erste Hauptteil dieser Arbeit beschreibt mögliche Chancen und Risiken des Voice Clonings.

Im zweiten Hauptteil dieser Arbeit wird der aktuelle Stand der Wissenschaft sowie die Praxistauglichkeit dieser Ansätze mithilfe aktueller Papers und deren Code-Implementationen erarbeitet und getestet.

Abschließend wird ein Resümee gezogen.

2 Grundlagen

In diesem Kapitel werden zunächst einige technische Hintergründe der Digitalen Sprachverarbeitung aufgezeigt, um ein Verständnis von Audiosignalen, Signalverarbeitung und Sprachsynthese zu vermitteln. Für letztere können Methoden des maschinellen Lernens eingesetzt werden, weshalb diese Technik grundlegend erläutert wird. Abschließend wird ein Ansatz zur Kombination beider Techniken am Beispiel von Voice Cloning erklärt.

2.1 Digitale Sprachverarbeitung

Für das Verständnis von digitaler Sprache ist Wissen über Audio-(Ton-)Signale erforderlich. Audiosignale können anhand ihres Pegels (Amplitude der elektrischen Spannung) und ihrer Tonhöhe (Frequenz) im Verlauf der Zeit beschrieben werden.

Bei Sprache entspricht der Pegel der Lautstärke, zum Beispiel gemessen in Dezibel (dB). Die Tonhöhe entspricht bei Sprache der Stimmlage, gemessen in Hertz (Hz) und kategorisiert in Klassen wie Bass, Tenor und Sopran.

Zur Visualisierung von Audiosignalen haben sich zwei Darstellungsformen etabliert, die in Abbildung (Abb.) 2.1 dargestellt sind. Während die Wellenformdarstellung den Pegel einer konstanten Frequenz im Verlauf der Zeit zeigt, visualisiert das Spektrogramm Pegel-Änderungen aller Frequenzen im Verlauf der Zeit (Pfister und Kaufmann 2017, S. 43–45, 48–50).

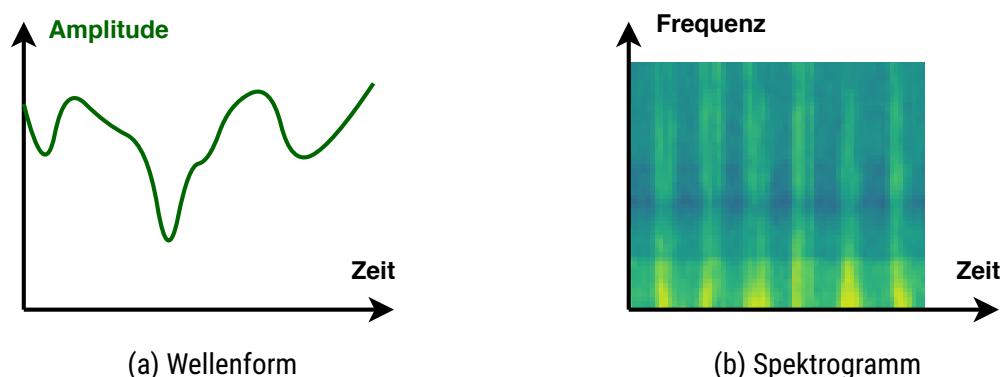


Abb. 2.1: Darstellung eines Audiosignals als Wellenform und Spektrogramm. Je heller die Farbe eines Punktes beim Spektrogramm ist, desto höher ist an diesem Zeit- und Frequenzpunkt die Amplitude des Signals.

Um Audiosignale digital verarbeiten zu können ist eine Umwandlung der analogen gesprochenen Sprache zu digitalen Signalen notwendig. Hierfür wird eine Abtastung mit Diskretisierung vorgenommen. Darauf aufbauende Ansätze, wie die Erkennung von gesprochener Sprache als Text (Spracherkennung beziehungsweise Speech-to-Text) arbeiten auf Basis dieser digitalen Signale. Dieser Vorgang wird in Abb. 2.2 gezeigt.



Abb. 2.2: Umwandlung von gesprochener Sprache zu Text. Die natürliche gesprochene Sprache wird zunächst digitalisiert und je nach Implementierung zusätzlich kodiert. Eine anschließende Merkmalsanalyse (beispielsweise mittels maschinellem Lernens) versucht Muster zu erkennen, die zur Erkennung der gesprochenen Wörter verwendet werden können. So kann abschließend ein digitaler geschriebener Text generiert werden (Nach Rabiner und Schafer 2011, S. 9–13).

Der in Abb. 2.2 gezeigte Vorgang kann auf einer ähnlichen Weise umgekehrt ausgeführt werden, um eine Sprachausgabe (Text-To-Speech) zu ermöglichen. Diese ist in Abb. 2.3 dargestellt.

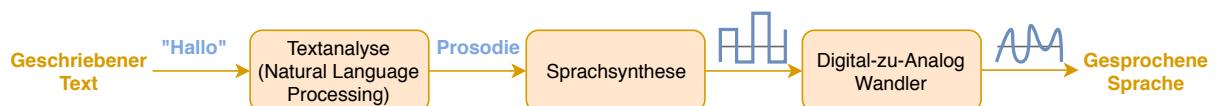


Abb. 2.3: Umwandlung von Text zu gesprochener Sprache. Der geschriebene Text wird zunächst anhand linguistischer Grundlagen analysiert um seine Prosodie zu ermitteln. Anschließend findet mithilfe Digitaler Signalverarbeitung eine Sprachsynthese statt. Abschließend rekonstruiert ein Digital-Analog-Umsetzer die für ein kontinuierliches analoges Signal notwendigen Schwingungsparameter auf Grundlage des digitalen Signals (Nach Mihkla 2008, S. 56).

Moderne Sprachassistenten wie Google Assistant oder Amazon Alexa kombinieren diese beiden Techniken, um eine natürlich klingende Konversation zu ermöglichen. Zur Interpretation des nach der Umwandlung als geschriebenen Text vorliegenden Wunsches des Nutzers, ist allerdings ein Verständnis der Intention des Satzes notwendig. Diese Bedeutung kann mittels künstlicher Intelligenz ermittelt werden, die im folgenden Abschnitt grundlegend beschrieben wird.

2.1.1 Methoden zur Sprachsynthese

Bereits seit dem siebzehnten Jahrhundert werden Ansätze zur Synthese von Sprache erarbeitet (Kempelen 1791). Der erste bekannte Ansatz von Kempelen (1791) war rein mechanischer Art, wie in Abb. 2.4a gezeigt.

Die ersten elektronischen Ansätze zur Sprachsynthese basieren auf der sogenannten Formantsynthese, bei der einzelne Formante („Bereiche im Klangspektrum, in denen sich unabhängig von der Tonhöhe Schallenergie konzentriert“ (Spektrum Akademischer Verlag 1998)) mithilfe elektronischer Bauteile wie beispielsweise Filter erzeugt und kombiniert werden. Mithilfe von jeweils zwei Formanten können Laute erzeugt werden, die für das menschliche Ohr wie ein beliebiger Vokal klingen. Durch Kombination mehrerer Formanten lassen sich daher Wörter und Sätze sprechen. Zu diesen Verfahren gehört unter anderem der „VODER“ (Voice Operation DEMonstrator) von Dudley, Riesz und Watkins (1939), dessen Aufbau in Abb. 2.4b gezeigt ist.

Ein sehr interessanter, in der Praxis bisher aber noch nicht eingesetzter Ansatz ist die „Artikulatorische Synthese“, bei der menschliche Sprechorgane und deren Akustik modelliert wird. Es kann hierbei unterschieden werden zwischen dem Modell der schwingenden Stimmlippen und dem Modell des Vokaltrakts, das neben den Lippen auch die Nase berücksichtigt. Die artikulatorische

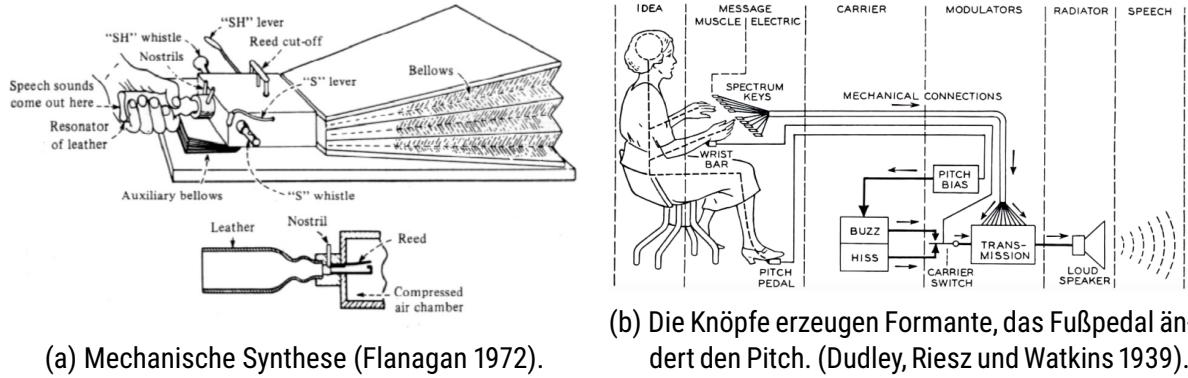


Abb. 2.4: Mechanische und Elektronische Verfahren zur Sprachsynthese.

Synthese bietet das Potential, die Akustik des Menschlichen Körpers sehr genau nachzubilden, die Berechnung erweist sich bisher allerdings als sehr rechenintensiv. Bisherige Ansätze verwenden daher stets lediglich Näherungen des menschlichen Vorbilds.

Neuere Ansätze basieren vorwiegend auf der Modellierung eines Signals durch Zusammenhängen mehrerer bestehender Sprachaufnahmen (Samples). Es können drei Varianten dieser „Konkatenativen Synthese“ unterschieden werden, die sich in erster Linie durch die Länge der einzelnen Sprachsignalstücke unterscheiden. Beim „phrase splicing“ werden üblicherweise ganze Wörter verwendet, was für Systeme mit abgegrenztem Umfang geeignet ist, wie beispielsweise für eine sprechende Uhr, ein Navigationssystem oder die Menüführung eines Anrufbeantworters. Diese Variante hat den Vorteil, dass bei kleinen Anwendungen nur wenige Wörter eingesprochen werden müssen, allerdings klingen die Übergänge zwischen den Verkettungen oft nicht sehr natürlich.

Bei der „unit selection“ werden zahlreiche Halbphone, Phone, Diphone oder Halbsilben als Sprachaufnahmen verwendet. Dies erlaubt das Zusammensetzen zahlreicher neuer Wörter, allerdings stellt die Auswahl der geeigneten Sprachaufnahmen (Segmentierung) und die prosodisch „richtige“ Kombination dieser eine Herausforderung dar.

Bei der „general concatenative synthesis“ werden im Vergleich zur „unit selection“ etwas weniger Sprachaufnahmen verwendet, wobei es sich bei den meisten dieser um Diphone handelt. Der Vorteil im Gegensatz zur „unit selection“ ist hierbei, dass jede Einheit nur einmal vorkommt, weshalb eine einfachere Segmentierung möglich ist.

Die Auswahl geeigneter Sprachaufnahmen zur Konkatenation kann mithilfe von maschinellem Lernen erfolgen, weshalb diese Technik im Folgenden näher erläutert wird.

2.2 Machine Learning (ML)

Hinweis: Teile dieses Abschnitts wurden aus der Bachelorarbeit des Autors (Melde 2018) übernommen, da zum Teil die selben Grundlagen erklärt werden.

Machine Learning (ML) ist ein Teilgebiet des Forschungsgebiet „Künstliche Intelligenz (KI)“, in dem versucht wird, Mechanismen zu entwerfen, mit denen Maschinen oder Computer intelligentes Verhalten entwickeln können (Cyffka 2007, S. 556).

Der Begriff ML beschreibt eine Technik, bei der Computer-Algorithmen aus vorausgegangen Situationen lernen und Rückschlüsse für neue Situationen schließen (Murphy 2012, S. 1). Anwendung findet diese Technik in immer mehr Bereichen und wird beispielsweise zur Prognose von Aktienkursen (Patel et al. 2015), zur Vorhersage von Produktionsausfällen (Susto et al. 2015) oder zur Klassifikation von auf Bildern dargestellten Objekten (Murphy 2012, S. 7; Deng et al. 2009; Krizhevsky, Sutskever und Hinton 2012) eingesetzt.

ML besteht aus einer Lern- und Anwendungsphase. In dem Lern- oder auch Trainingsphase genannten Schritt lernt das System aus annotierten (beschrifteten) Daten („Erfahrung“), die es bei der späteren Anwendung zur Nachbildung von kognitiven Fähigkeiten des Menschen einsetzt. (Rashid 2017, S. 1)

Konkret ermöglicht ML beispielsweise die Klassifizierung von gesprochener Sprache, nur anhand der reinen Audiodaten, ohne dass weitere Metadaten wie beispielsweise Schlagworte in der Ton-Datei enthalten sein müssen. Diese Klassifizierung kann beispielsweise für die in Abschnitt 2.1 gezeigten Speech-To-Text-Anwendungen genutzt werden. (Oord et al. 2016, S. 8; Vasquez und Lewis 2019)

Die Funktionsweise von ML wird im Folgenden näher beschrieben.

2.2.1 Lernmethoden

Der Lernalgorithmus bestimmt die Parameter, mit dem das Modell des neuronalen Netzes aufgebaut wird. Allgemein wird zwischen drei Lernmethoden unterschieden (Lippmann 1987, S. 6–7; Murphy 2012, S. 2).

Beim überwachten Lernen (Englisch: *supervised*) nutzt das Training annotierte Beispieldaten, beispielsweise kategorisierte Bilder oder Audiodateien, um später in der Anwendungsphase andere Bilder oder Audiodateien in die gelernten Kategorien einzufügen (Lippmann 1987, S. 7; Murphy 2012, S. 2–3).

Beim unüberwachten Lernen (Englisch: *unsupervised*) wird für das Training eine nicht-annotierte Menge von Daten genutzt, die durch den ML-Algorithmus selbstständig in verschiedene Gruppen unterteilt werden (Lippmann 1987, S. 7; Murphy 2012, S. 2, 9).

Dieser Ansatz wird auch benutzt, um Reihen fortzusetzen, beispielsweise werden in einem Versuch von Srivastava, Mansimov und Salakhudinov (2015) mehrere auf einen kurzen Video-Ausschnitt folgende Einzelbilder vorhergesagt. Bei dem Anwendungsfall gesprochener Sprache kann so beispielsweise vorhergesagt werden, welche Betonung der Sprecher bei einem gegebenen Textausschnitt für die folgenden Wörtern nutzen wird (Descript 2019b).

Eine letzte Methode ist das sogenannte bestärkende Lernen (Englisch: *reinforcement learning*), bei dem ein System aus Belohnungen und Bestrafungen aufgebaut wird. Während der Anwendungsphase läuft der Trainings-Algorithmus weiter und wird bei guten Entscheidungen belohnt und bei schlechten Entscheidungen bestraft. Der stetige Versuch die Belohnung zu maximieren führt so zu immer besseren Ergebnissen. (Murphy 2012, S. 2)

2.2.2 Klassifikation

Zur Veranschaulichung, wie eine Klassifikation mithilfe von neuronalen Netzen funktioniert, wird zunächst die grundlegende Funktionalität einer Support Vector Machines (SVMs) beschrieben. SVM definieren eine Grenzfunktion, die zwei Klassen voneinander trennt. Liegt ein Wert über der Funktion, so gehört er zur ersten Klasse, liegt er darunter, so gehört er zur zweiten und nicht zur ersten Klasse. Die Achsen des Funktions-Graphen sind mit Merkmalswerten beschriftet, die zur Klassifikation genutzt werden. (Boser, Guyon und Vapnik 1992, S. 148)

In Abb. 2.5 wird zur Veranschaulichung eine beispielhafte SVM gezeigt.

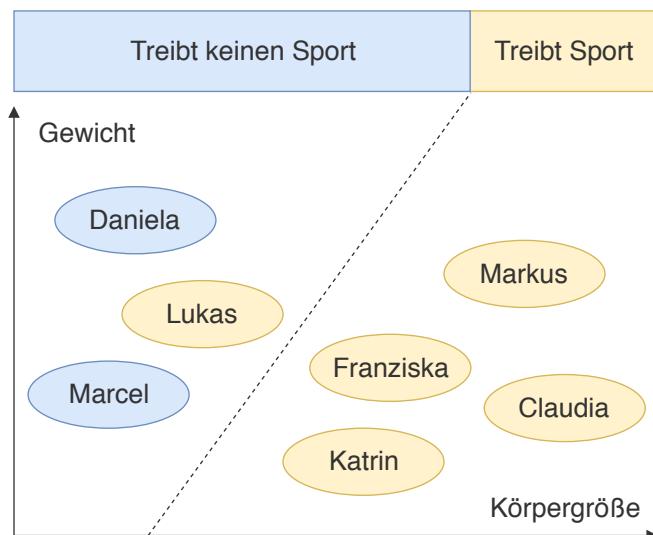


Abb. 2.5: Support Vector Machine (SVM) zur Klassifikation von Personen, die Sport treiben (gelb) und unsportlichen Personen (blau). Hierfür werden die Merkmale Gewicht und Größe verwendet. Durch den Klassifikator wird jeder, dessen Größe um einen bestimmten Wert oder Prozentsatz höher ist, als sein Gewicht, als Sportler klassifiziert. Während der Trainingsphase wird versucht, die Trennlinie (gepunktet) an die richtige Position zu setzen. Die Person Lukas ist Sportler, wird durch die aktuelle Trennlinie aber nicht als solcher klassifiziert. Dies ist ein Zeichen dafür, dass die gewählten Merkmale möglicherweise nicht zur Klassifikation geeignet oder nicht ausreichend sind.

Bei einer klassischen SVM werden die Merkmale zur Unterscheidung der verschiedenen Objekte manuell entworfen, weshalb diese, wie in Abb. 2.5 gezeigt, nicht immer sinnvoll sind.

Zur selbstständigen Gewichtung möglicher Merkmale werden künstliche neuronale Netze eingesetzt.

2.2.3 Künstliche neuronale Netze

Künstliche neuronale Netze sind eine Möglichkeit, Computern zu beschreiben, wie „lernen“ funktioniert.

Die Idee hierbei ist es, die Funktionsweise des menschlichen Gehirns nachzubilden. Neuronale Netze bestehen im Gehirn hauptsächlich aus Neuronen des Nervensystems und Synapsen, die diese miteinander verbinden. Die Neuronen sind in verschiedenen Schichten (*layers*) angeordnet und werden mithilfe von Reizen gesteuert. (Rashid 2017, S. 30, 36; Lippmann 1987, S. 4)

Auch die im Computer nachgebildeten, künstlichen neuronale Netze bestehen aus Neuronen und Synapsen. Während komplexe Netze aus mehreren Millionen Neuronen bestehen, genügen für ein einfaches Beispiel bereits zwei Neuronen (siehe Abb. 2.6). (Rashid 2017, S. 31)

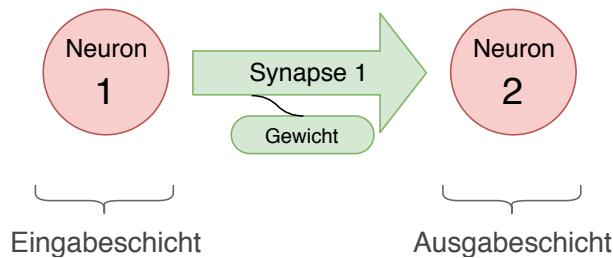


Abb. 2.6: Minimales neuronales Netz aus zwei Neuronen und einer Synapse.

In der in Abb. 2.6 gezeigten Eingabeschicht (*input layer*) liegt ein Neuron, das Eingangsdaten entgegen nimmt und diese an alle Synapsen weiterleitet. Jede Synapse hat ein bestimmtes Gewicht (*weight*), mit dem alle Eingangsdaten multipliziert werden, bevor sie an das Neuron in der Ausgabeschicht (*output layer*) weitergegeben werden. (Lippmann 1987, S. 5)

Die Ausgabe kann also verändert werden, indem die Gewichte der Synapsen geändert werden. Machine Learning Algorithmen versuchen während des Trainings, die Gewichte so oft anzupassen, bis die Ausgabe für alle Eingabewerte zu den jeweils passenden Ausgabewerten führt.

Dieser Vorgang kann anhand des folgenden Beispiels veranschaulicht werden. Es wird angenommen, dass die Elemente der Menge der Eingabewerte $x = \{1, 2, 5\}$ auf die Ausgabewerte $f(x) = \{2, 4, 10\}$ abgebildet werden sollen. Da das in Abb. 2.6 gezeigte neuronale Netz aus nur einer Synapse besteht, gibt es nur ein Gewicht, das zum Erreichen der Ausgabewerte angepasst werden kann: $f(x) = S_1 * x$. Diese Formel lässt sich bereits mit mathematischen Grundkenntnissen auflösen, die Funktion ist für die gegebenen Beispielwerte genau dann wahr, wenn für das Synapsen-Gewicht $S_1 = 2$ gilt. (Melde und Schneider 2018, S. 22)

Der ML-Algorithmus versucht, das Gewicht ohne diese mathematischen Kenntnisse zu bestimmen. Während der Trainings-Phase werden die Gewichte solange schrittweise um einen gewissen Wert (Lernrate, *step size*) erhöht oder verringert, bis die gewünschten Ausgabewerte erreicht werden.

In der Praxis bestehen neuronale Netze häufig aus mehreren Ein- und Ausgabe-Neuronen. Eine Audiodatei, die beispielsweise gesprochene Sprache enthält, besteht, wie in Abschnitt 2.1 gezeigt für jede Frequenz aus einer bestimmten Anzahl von Amplitudenwerten. Soll eine solche Audiodatei klassifiziert werden, gibt es für jeden Frequenz-Amplitudenwert ein Eingabe-Neuron. Für jede zu unterscheidende Klasse gibt es ein Ausgabe-Neuron, das die Wahrscheinlichkeit für die jeweilige Klasse angibt.

Damit eine sinnvolle Verknüpfung der Ein- und Ausgabeneuronen auch bei umfangreicheren Anwendungsfällen möglich ist, werden fortgeschrittenere Netze mit weiteren Synapsen und Neuronen

zwischen der Eingabe- und der Ausgabeschicht eingesetzt (siehe Abb. 2.7) (Lippmann 1987, S. 10, 16).

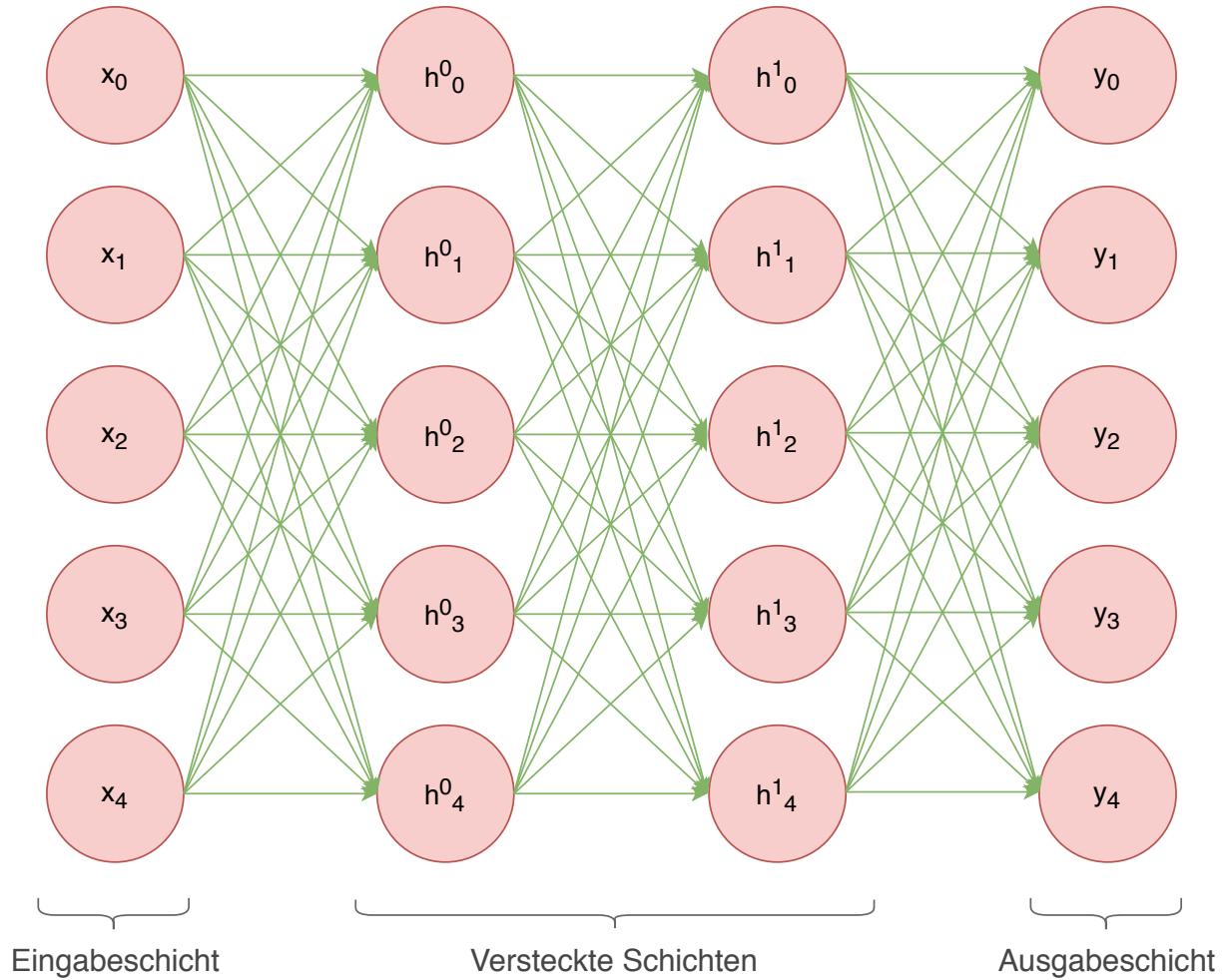


Abb. 2.7: Ein aus drei Schichten bestehendes neuronales Netz (der Eingangsvektor in der Eingabeschicht wird nicht gezählt) mit je fünf Ein- und Ausgabeneuronen. Jede Schicht ist vollständig mit den benachbarten Schichten verbunden (*fully connected layers*). (Lippmann 1987, S. 16)

In solchen größeren Netzen gibt es deutlich mehr Synapsen und damit Gewichte, die berechnet werden können. Anfangs werden die einzelnen Gewichte meist mit Zufallswerten bestückt. Anschließend werden die Gewichte, wie im letzten Beispiel gezeigt, durch „ausprobieren“ so lange verändert, bis ein gutes Ergebnis erzielt wird. (Rashid 2017, S. 4–5)

In größeren neuronalen Netzen erhält ein Neuron also Eingabewerte von mehreren Synapsen. Die verschiedenen Werte werden miteinander verrechnet und anschließend mithilfe einer sogenannten „Aktivierungsfunktion“ normalisiert. Zur Diskretisierung werden die einzelnen Ausgaben der Vorgänger-Neuronen mit dem jeweiligen Synapsen-Gewicht multipliziert und anschließend summiert. Für das Neuron h_1^0 aus Abb. 2.7 wird beispielsweise mit dem Synapsen-Gewicht $w(a, b)$ zwischen zwei Neuronen a und b die folgende Summe berechnet: (ebd., S. 49)

$$h_1^0 = x_0 * w(x_0, h_1^0) + x_1 * w(x_1, h_1^0) + x_2 * w(x_2, h_1^0) + x_3 * w(x_3, h_1^0) + x_4 * w(x_4, h_1^0) \quad (2.1)$$

Um den Eingangswert des Neurons zu erhalten, wird nun diese Summe an die Aktivierungsfunktion übergeben. Je nach Art des neuronalen Netzes wird hierfür eine andere Funktion genutzt. Gängig

ist es, eine lineare Funktion, eine binäre Stufenfunktion oder eine Sigmoid-Funktion zu verwenden (siehe Abb. 2.8). (Rashid 2017, S. 33–34)

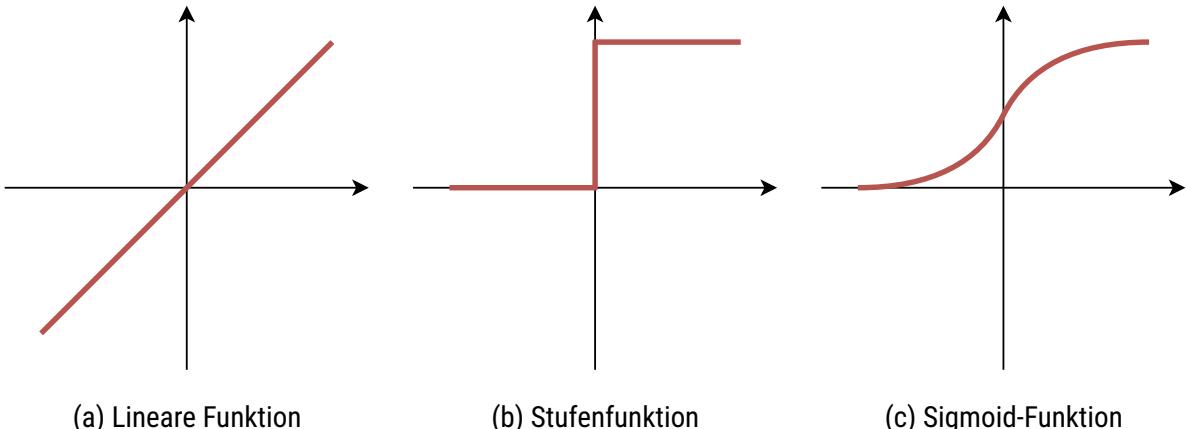


Abb. 2.8: Verschiedene Aktivierungsfunktionen für künstliche neuronale Netze. Die Skalierung der Achsen variiert je nach Implementation, oft wird der Wertebereich $[0,1]$ gewählt. Die x -Achse (horizontal) beschreibt den Eingangswert, die y -Achse (vertikal) die Funktion $f(x)$ (rot) als Ausgangswert. (nach Rashid 2017, S. 33–34; Lippmann 1987, S. 5)

Die Idee hinter den Aktivierungsfunktionen entstammt der Biologie, da auch menschliche Neuronen von einem Schwellwert abhängig aktiviert werden (Rashid 2017, S. 31–32).

Um die eben beschriebenen Rechen-Operationen für ein komplettes neuronales Netz durchzuführen, genügt es, die Matrix aller Eingabewerte mit der Matrix aller Gewichte zu multiplizieren und die Aktivierungsfunktion anschließend auf die sich ergebende Matrix anzuwenden (ebd., S. 49–50).

In tiefen neuronalen Netzen, die aus sehr vielen Schichten bestehen, werden auch Rectified Linear Units (ReLUs) als Aktivierungsfunktion genutzt (Agarap 2018). Eine ReLU-Funktion $f(x)$ steigt für alle $x \geq 0$ linear mit der Funktion $f(x) = x$. Für die Werte $x < 0$ wird je nach ReLU-Implementierung entweder $f(x) = 0$ gesetzt oder es wird eine deutlich schwächer steigende Funktion, wie $f(x) = 0,01 * x$, verwendet. (Kapur 2013)

2.2.4 Convolutional Neural Networks (CNNs)

CNN sind eine spezielle Form der eben vorgestellten neuronalen Netze. Bei CNN sind die Neuronen einer Schicht nicht nur eindimensional in einer Spalte, sondern in einer zweidimensionalen Matrix, auch Faltungsmatrix genannt, dargestellt. Die Synapsen werden in diesem Modell Kernel genannt und berücksichtigen nun mehrere, in der Matrix nebeneinander liegende Neuronen auf einmal (siehe Abb. 2.9). (Karn 2016)

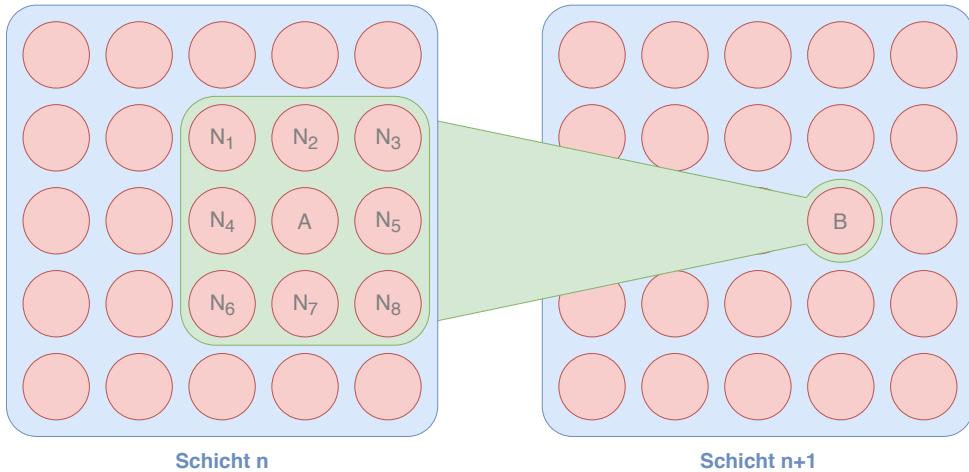


Abb. 2.9: Convolutional Neural Network (CNN). Zur Berechnung der Neuronen der Schicht $n + 1$ (zum Beispiel B) werden das jeweilige Vorgänger-Neuron (A) und seine Nachbar-Neuronen ($N_1 \dots N_8$) berücksichtigt. Diese Matrix (grün) wird Kernel genannt.

Zur Berechnung der Neuronen wurde im vorigen Abschnitt die Formel $f(x) = S_1 * x$ genutzt. Es wurden also das Synapsen-Gewicht S_1 mit dem Vorgängerwert x multipliziert. Bei CNN passt das gleiche, allerdings werden sowohl für das Gewicht als auch für den Vorgängerwert nun Matrizen verwendet, weshalb zur Diskretisierung noch das Ermitteln eines arithmetischen Mittels hinzukommt. Mit den Bezeichnungen aus Abb. 2.9 sowie dem Gewicht des Kernels G_K lässt sich die folgende Formel aufstellen:

$$B = \frac{\text{sum} \left(G_K * \begin{pmatrix} N_1 & N_2 & N_3 \\ N_4 & A & N_5 \\ N_6 & N_7 & N_8 \end{pmatrix} \right)}{9} \quad (2.2)$$

Um diese Formel zu veranschaulichen, werden Beispielwerte eingesetzt:

$$B = \frac{\text{sum} \left(\begin{pmatrix} 1 & 1 & 1 \\ 1 & 90 & 1 \\ 1 & 1 & 1 \end{pmatrix} * \begin{pmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{pmatrix} \right)}{9} = 94,4 \quad (2.3)$$

In diesem Beispiel ist das zentrale Neuron A im Gewicht des Kernels am stärksten gewichtet. Die diskretisierende Funktion, hier das arithmetische Mittel, wird *Pooling-Funktion* genannt und ist je nach Anwendungsfall unterschiedlich.

Die in der Praxis am häufigsten verwendete Pooling-Funktion verwendet statt dem arithmetischen Mittel (Mean-Pooling) das Maximum der Matrix (Max-Pooling). Eine weitere, allerdings überwiegend für die Bildklassifikation verwendete Methode, ist Softmax-Pooling, bei dem die Softmax-Funktion (auch normalisierte Exponentialfunktion genannt) (Bishop 2016, S. 198) eingesetzt wird. (Karn 2016)

CNN können effizient zur Klassifikation von Bildern eingesetzt werden (Karpathy et al. 2014, S. 1; Donahue et al. 2015, S. 1; Krizhevsky, Sutskever und Hinton 2012). Einige Forschungsansätze versuchen daher, Audiofiles grafisch darzustellen, beispielsweise mit einem Spektrogramm (siehe Abschnitt 2.1), und dann diese Repräsentationen zu klassifizieren (Su et al. 2019, S. 7). Ein beispielhaftes CNN zur Klassifikation von Bildern ist in Abb. 2.10 gezeigt.

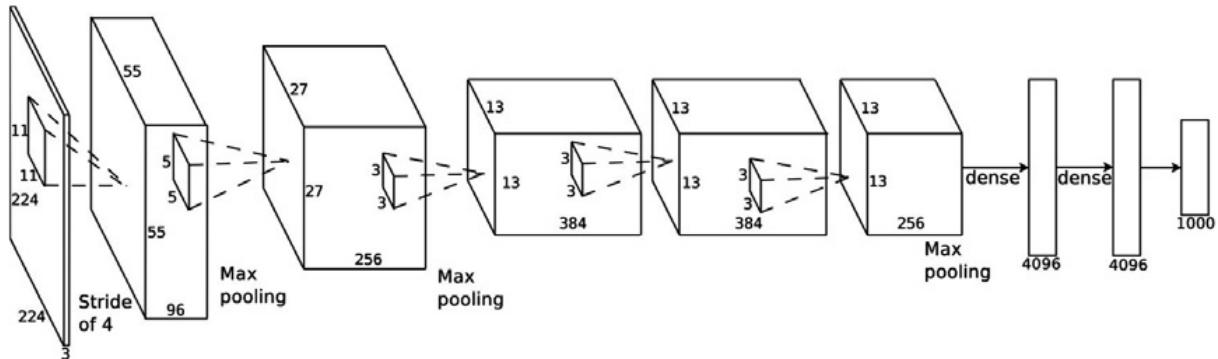


Abb. 2.10: Beispielhafte Architektur eines CNNs zur Bildklassifikation. Das CNN besitzt acht Schichten, zwischen denen mittels verschieden großer Kernel mehrere Neuronen zusammengefasst werden (aus Smirnov, Timoshenko und Andrianov 2014, S. 91).

Bilder wie das Spektrogramm können als Matrix repräsentiert werden, wenn alle Pixel des Bildes und die Intensität einzelner Grundfarben als eine solche angeordnet werden. Um ein ganzes Bild zu verarbeiten, „fährt“ der Kernel nun über die gesamte Matrix, also über alle Neuronen der ersten Schicht und gibt die Ergebnisse an die nächste Schicht weiter. (Karn 2016)

Die versteckten Schichten eines CNN werden genutzt, um mehrere Neuronen zusammenzufassen. Bei einem Bild kann man sich das als Zusammenfassen von Bildausschnitten vorstellen. Die Aufgabe des Trainings ist weiterhin das Bestimmen der Gewichte. (ebd.)

3D-CNN

Die bisher vorgestellten CNN sind gut zur Klassifikation von Daten eines einzelnen Zeitpunkts geeignet, beispielsweise Bilder oder Audiodateien fixer Länge einer Frequenz, können aber nicht gut für Zeitreihen wie beispielsweise Videos eingesetzt werden, da hierfür eine dreidimensionale Kernel-Architektur benötigt würde (Tran et al. 2015, S. 1). CNN, die dreidimensionale Kernel verwenden, werden 3D-CNN genannt (ebd., S. 1). Mithilfe dieser ist es möglich, zeitliche Bezüge innerhalb des neuronalen Netzes zu berücksichtigen (Ji et al. 2013, S. 2; Tran et al. 2015, S. 1).

In Abb. 2.9 wurde gezeigt, wie der Kernel einen Teil der zweidimensional angeordneten Neuronen zusammenfasst. Zur Visualisierung eines 3D-CNNs müssten in Abb. 2.9 für jede Schicht n_t weitere Schichten n_{t-1}, n_{t-2}, \dots hinter die Schicht eingezeichnet werden, wobei die weiteren Schichten für jedes Neuron die jeweiligen Vorgänger-Werte enthalten (Ji et al. 2013, S. 3).

Der 3D-Kernel berücksichtigt die Werte der nahen Vergangenheit, bei der Auswertung von lang andauernden Zusammenhängen stoßen aber auch 3D-CNN an ihre Grenzen (ebd., S. 3).

2.2.5 Recurrent Neural Networks (RNNs)

RNNs berücksichtigen im Gegensatz zu den bisher vorgestellten Netzen nicht nur die aktuelle Eingabe, sondern auch mehrere Eingabewerte, die zuvor durch das Netz geschickt wurden (Elman 1990; Karpathy 2015).

Hierdurch können neue Aufgaben wie das Erstellen von Sätzen, Übersetzungen oder Trendanalysen gelöst werden. RNNs werden darüber hinaus zum Verarbeiten menschlicher Sprache und zur Klassifikation von Videos eingesetzt (Kapur und Khazan 2017).

Die Architektur von RNN ist dynamisch, weshalb im Unterschied zu den bereits vorgestellten neuronalen Netzen Ein- und Ausgaben verschiedener Größen mit dem selben Netz verarbeitet werden können. Angewendet wird dieser Vorteil unter anderem bei der Generierung und Verarbeitung von Sequenzen mit dynamischer Länge. (ebd.)

In Abschnitt 2.2.3 wurde gezeigt, dass ein neuronales Netz aus mehreren Schichten besteht. Jede dieser Schicht besteht aus mehreren Neuronen, die je einen Skalar verarbeiten (siehe Abb. 2.11a). In RNN-Neuronen können Vektoren verarbeitet werden, weshalb ein Neuron in einem RNN die Funktionalität einer ganzen Schicht eines klassischen neuronalen Netzes abbilden kann (siehe Abb. 2.11b) (ebd.).

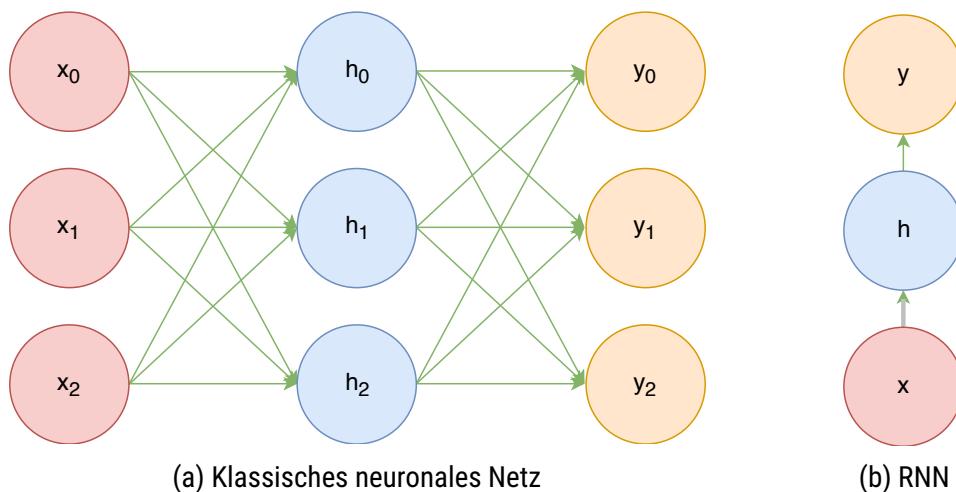


Abb. 2.11: Darstellung eines (a) klassischen neuronalen Netzes als (b) RNN. (nach Kapur und Khazan 2017)

RNN können zur Klassifikation von Sequenzen benutzt werden (*many to one*, siehe Abb. 2.12a), zur Generierung von Sequenzen anhand einer festen Eingabe (*one to many*) oder zur Generierung von Sequenzen anhand einer Sequenz (*many to many*, siehe Abb. 2.12b) (Karpathy 2015; Kapur und Khazan 2017).

Many to many RNN können unter anderem zum Übersetzen von Texten zwischen verschiedenen Sprachen oder zum Generieren einer textuellen Beschreibung für ein Video eingesetzt werden (Karpathy 2015; Kapur und Khazan 2017).

Eine Spezialform von RNNs sind die Bidirectional RNNs (BRNNs), die eine zusätzliche rückwärts laufende Schicht besitzen. Zur Visualisierung eines BRNN müssten in Abb. 2.12b alle von links nach rechts laufenden Pfeile der oberen verborgenen Schicht (blau eingezzeichnet) umgekehrt werden.

Bei RNN tritt das sogenannte „Vanishing Gradient“ Problem auf, das den Zeitraum in den zurückgeschaut wird, einschränkt (Donahue et al. 2015, S. 2).

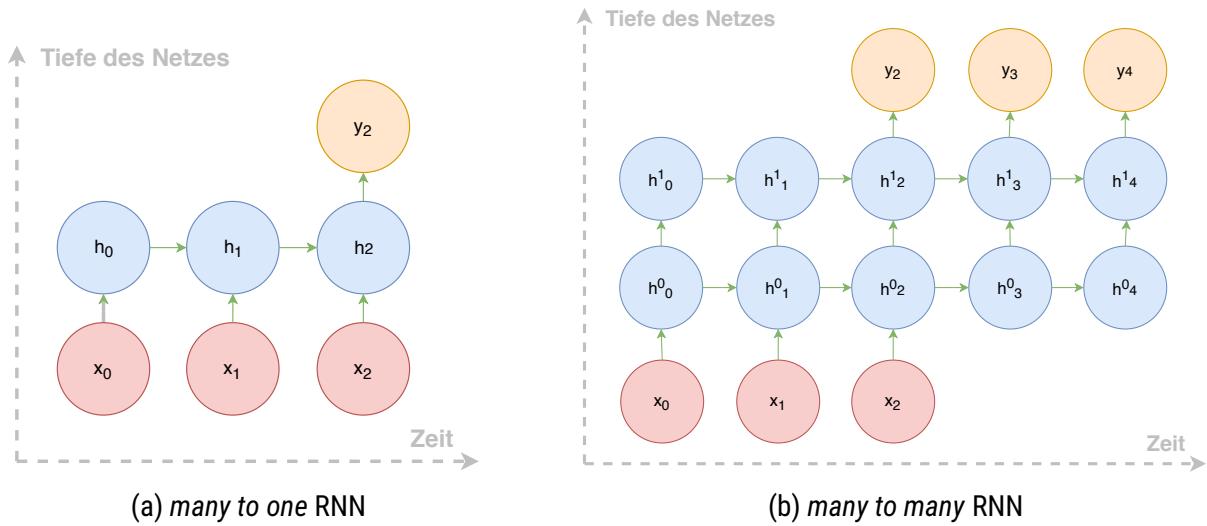


Abb. 2.12: Verschiedene Typen von RNN (nach Karpathy 2015; Kapur und Khazan 2017).

Zur Berechnung der Gewichte eines RNN ist das Bestimmen von Minima einer Funktion in einem mehrdimensionalem Raum notwendig. Hierfür kommt ein „Gradientenverfahren“ zum Einsatz, dass Ableitungen und deren Steigungen berechnet. Mit einer steigenden Anzahl an Neuronen sind die einzelnen Steigungen weniger steil.

Um die kleinen Steigungen darzustellen, sind viele Nachkommastellen notwendig. Da in Computern die Größe von Zahlen sowohl nach oben (*overflow*) als auch nach unten (*underflow*) beschränkt ist, werden Steigungen ab dem Unterschreiten eines gewissen Grenzwertes fälschlicherweise als Null angegeben. Für ein RNN können dann keine Gewichte mehr berechnet werden, was ein Training verhindert. (Donahue et al. 2015; Kapur und Khazan 2017; Karpathy 2015; Kapur 2013)

Long Short-Term Memory (LSTM)

Eine Möglichkeit zur Vermeidung des „Vanishing Gradient“ Problems ist die Verwendung von sogenannten Long Short-Term Memory (LSTM) Zellen (Hochreiter und Schmidhuber 1997, S. 1; Olah 2015). In den LSTM Zellen befinden sich verschiedene Gatter (Hochreiter und Schmidhuber 1997, S. 7; Donahue et al. 2015, S. 2), die es ermöglichen, Zustände weiterzugeben, diese zu modifizieren, zu aktualisieren, zurückzusetzen (Donahue et al. 2015, S. 2) oder mit der Zeit zu summieren (Srivastava, Mansimov und Salakhudinov 2015, S. 3).

LSTMs ermöglichen es daher, Zusammenhänge und Merkmale aus längeren Zeiträumen zu detektieren (ebd., S. 3).

Werden in einem RNN LSTM-Zellen verwendet, so spricht man auch von LSTM-Netzen.

LSTM eignen sich gut für Spracherkennung (Graves und Jaitly 2014) und Sprachübersetzung (Sutskever, Vinyals und Le 2014; Cho et al. 2014) sowie zur Auswertung und Prognose von Werten im zeitlichen Verlauf (Olah 2015), beispielsweise zum Erkennen einer Dokumentstruktur oder Grammatik.

2.3 Voice Cloning

Voice Cloning verfolgt das Ziel, menschliche Stimmen anhand kurzer Beispiele möglichst exakt digital nachzubilden, sodass weitere Sätze durch diese Stimme gesprochen werden können.

Im Unterschied zur bereits beschriebenen Sprachsynthese versucht Voice Cloning nicht nur eine einzelne menschlich klingende Stimme spezialisiert zu erzeugen, sondern es soll ein generelles Modell gelernt werden, anhand dem verschiedene Stimmen von real existierenden Personen auch nach der eigentlichen Entwicklung nachgebildet werden können.

Möglich ist dies durch eine Kombination von digitaler Sprachverarbeitung und Künstlicher Intelligenz. Die Modelle des maschinellen Lernens untersuchen in Wellenformen oder Spektrogrammen Muster und Merkmale und lernen, welche davon für Stimmen unterscheidungswirksam sind, so dass anschließend anhand dieses Modells eine Sprachsynthese (Text-to-Speech) erstellt werden können. Hierbei können dann die Sprachgeschwindigkeit, die Tonhöhe und die Stimmelodie auf die individuelle Stimme und Stimmmelodie angepasst werden.

Auch denkbar ist das Einsetzen und Verwenden von für den Sprecher typischen Wörtern und Füllwörtern.

Für den Anwendungsfall des Voice Clonings scheint sich besonders das überwachte Lernen zu eignen, da die Generierung von Sprache auf uns bereits bekannten Stimmen basieren soll.

Das unüberwachte Lernen eignet sich hierfür nicht, da die durch den Algorithmus neu gruppierten Stimmmerkmale und erstellten Stimmen möglicherweise nicht für Menschen verständlich wären.

Auch Reinforcement Learning scheint für den Anwendungsfall Voice Cloning nicht geeignet zu sein, da zur automatischen Verteilung der Belohnungen und Bestrafungen bereits eine fehlerfreie Spracherzeugung und eine fehlerfreie Spracherkennung existieren müssten.

Für das Voice Cloning könnte ein 3D-CNN eingesetzt werden, da Sprache im Verlauf der Zeit von der höheren Dimensionalität dieser Netze profitiert. Um lange Zusammenhänge gut zu verstehen scheint ein RNN oder sogar LSTM aber noch besser geeignet. Konkret könnte ein *many to many* RNN eingesetzt werden, da dieses anhand mehrerer Beispielwörter ebenfalls mehrere Wörter generieren soll.

Ein einfacher Ansatz wäre ein Vergleich der Beispiel-Audio-Datei mit bestehenden Stimmen und die Auswahl der nächstbesten bekannten Stimme (Nearest Neighbour). Dies würde aber nicht wirklich die gewünschte Stimme klonen. Besser wäre es, wenn mithilfe des Machine Learning Modells die bestehende Stimme anhand des Testsamples angepasst wird (Retraining).

3 Chancen und Risiken des Voice Clonings

Bevor im nächstem Kapitel wissenschaftliche Publikationen und konkrete Implementationen evaluiert werden, sollen in diesem Kapitel ganz bewusst unabhängig vom aktuellen Stand der Technik einige Chancen und Risiken des Voice Clonings anhand möglicher Anwendungsfälle und Beispiele aufgezeigt werden.

Abschließend wird kurz auf die Rechtslage beim Voice Cloning eingegangen, damit keine falschen Erwartungen geschaffen werden.

3.1 Chancen

Digitales Voice Cloning bietet zum einen das Potential, Prozesse in zahlreichen bestehenden Anwendungen zu vereinfachen und ermöglicht andererseits aber auch ganz neue Möglichkeiten.

Beispielsweise könnten Chatbots und digitale Assistenten wie Google Assistant, Amazon Alexa oder Microsoft Cortana mithilfe von Voice Cloning zukünftig eine viel breitere Auswahl an Stimmen anbieten als heute, da es dann möglich wäre, den Assistenten mit der Stimme des Benutzers oder eines Freundes dessen sprechen zu lassen, oder auch mit der Stimme eines Prominenten.

Auch Hörbücher könnten von Prominenten, dem Autor oder von eigenen Verwandten vorgelesen werden.

Telefon-Hotlines haben zurzeit das Problem, dass Roboterstimmen oft direkt als solche erkannt werden. Voice Cloning könnte hier dabei helfen, eine auf das Geschäftsmodell und die jeweilige Zielgruppe abgestimmte Stimme zu verwenden, und dadurch die Conversion-Rate zu erhöhen.

Auch Videospiele-Studios wären ein guter Abnehmer für die Technologie, da viele Rollenspiele beispielsweise Nicht-Spielbare-Charaktere (sogenannte NPCs) verwenden, die mit dem Nutzer sprechen und interagieren. Mit Voice Cloning wäre es möglich, ohne aufwendige Aufnahmen von Synchronsprechern Texte vorzulesen und auch erstmalig möglich, individuelle Stimmen während des Spiels zu generieren, sodass nicht nur die Dateigröße des Spiels sinken kann, sondern auch Dialoge eigene Elemente wie den Namen des Spielers beinhalten können.

Websites und Nachrichtenverlage könnten Voice Cloning nutzen, um geschriebene Artikel und Inhalte mit einer benutzerdefinierten Stimme vorzulesen. So können nicht nur neue Zielgruppen erreicht werden (Seheingeschränkte), sondern auch das Angebot für bestehende Kunden erweitert werden.

Auch Personen, die durch einen Unfall oder eine Krankheit ihre Stimme verloren haben und nicht mehr sprechen können, könnten mit dieser Technik ihre eigene Stimme zurück bekommen (Grossman 2011).

Abschließend wird noch eine Idee aus der fiktiven Fernseh-Serie „Black Mirror“ vorgestellt. In der Folge „Be Right Back“ wird ein menschenähnlicher Roboter geschaffen, der mit der Stimme eines

verstorbenen Freundes spricht. Hierdurch kann die Protagonistin den Tod ihres Freundes besser verarbeiten. Diese Idee scheint gar nicht so abwegig, da es bereits eine Firma gibt, die derartige Pläne verfolgt, und auch Google hat die Option, die Persönlichkeit eines verstorbenen Geliebten digital nachzubilden bereits in einem Patentantrag erwähnt (O'Neill 2016).

3.2 Risiken

Wie viele modernen Technologien kann allerdings auch Voice Cloning für moralisch weniger gut vertretbare oder sogar illegale Zwecke eingesetzt werden.

Erst vor kurzem wurde beispielsweise ein Fall bekannt, bei dem ein Betrüger einen Angestellten einer Firma mit der gefakten Stimme dessen Vorgesetzten anrief, um eine Überweisung zu veranlassen. Da der Mitarbeiter diesem Arbeitsauftrag Folge leistete, wurden über 200.000 Euro an das Bankkonto des Betrügers überwiesen (Brown 2019).

Derartig manipulierte Stimmen können auch für Fake-News oder sogenannte Deep-Fakes eingesetzt werden, also Falschmeldungen oder Videos, die mithilfe tiefer neuronaler Netze erzeugt wurden.

Bei der Erkennung solcher Verbrechen scheint es sich um ein „Katz-und-Maus-Spiel“ zu handeln. Während die Betrüger sicherlich darauf bedacht sind, immer neue und besser täuschendere Ansätze zu verwenden, entwickeln Behörden und Unternehmen mit einem Interesse an der Aufdeckung von Fake-News bereits Machine-Learning-Modelle zur Detektion von gefakten Stimmen (Schroepfer 2019).

3.3 Rechtslage

Auch wenn es sich bei dieser Arbeit keinesfalls um eine Rechtsberatung handeln soll, soll an dieser Stelle auf einen wichtigen Punkt der deutschen Rechtsprechung hingewiesen werden, der so ähnlich vermutlich auch in anderen Ländern existiert.

Jegliche Impersonation einer anderen Person kann eine Verletzung des Artikel 2 Absatz 1 des Grundgesetzes (Recht zur freien Entfaltung der Persönlichkeit) in Verbindung mit Artikel 1 Absatz 1 desselben (Unantastbarkeit der Menschenwürde) darstellen.

Sämtliche Entwicklungen, Veröffentlichungen und Tests im Bereich des Voice Clonings sollten daher ausschließlich mit dem Einverständnis der daran beteiligten Personen geschehen.

4 Untersuchung bestehender Ansätze und Implementationen

In diesem Kapitel werden zunächst einige Ansätze aus der Forschung vorgestellt. Anschließend wird die Praxistauglichkeit dieser Ansätze geprüft, in dem Implementationen der State of the Art Ansätze und kommerzielle Produkte für Voice Cloning getestet, beschrieben und miteinander verglichen werden.

4.1 Forschung

Es wird bereits seit einigen Jahren an der Technik des Voice Clonings geforscht. Nachfolgend sollen die für die Wissenschaft bedeutendsten Ansätze vorgestellt und näher untersucht werden.

4.1.1 WaveNet (2016)

Im Jahr 2016 wurde von Oord et al. (Google DeepMind) *WaveNet* vorgestellt. *WaveNet* ist ein Ansatz zur Generierung von Sprache, der eine beliebige menschliche Sprache imitiert und weitaus natürlicher klingt, als vergleichbare Text-to-Speech Systeme dieser Zeit (Oord et al. 2016).

Mithilfe von WaveNets war es möglich, den messbaren Abstand der Natürlichkeit von Text-to-Speech Systemen und einer Menschlichen Stimme um über 50% zu verringern (ebd., S. 6).

Technisch basiert WaveNet auf der Erstellung einer generativen Wahrscheinlichkeitsverteilung, das bedingte Wahrscheinlichkeiten für verschiedene kurze Audiotücke angibt, die auf eine bestimmte gegebenes Audiotück folgen könnten. WaveNet benutzt hierfür namensgebend die Wellenformen der Audiodateien (siehe Abschnitt 2.1).

Das hierfür basierende Modell basiert auf einem CNN (siehe 2.2.4), das unverarbeitete Audiosignale als Eingabe akzeptiert und ein synthetisiertes Audiosignal als Ausgabe zurückgibt.

WaveNet wird unter anderem bei der Sprachgenerierung für den Google Assistant und den Google Übersetzer eingesetzt (Google Cloud 2019).

4.1.2 Baidu DeepVoice (2017)

Von Entwicklern des chinesischen Suchmaschinen-Unternehmens Baidu wurde 2017 ein Ansatz mit dem Namen DeepVoice veröffentlicht (Arik et al. 2018). Dieser ermöglichte es erstmals, auch mit sehr kurzen Beispielsätzen (Audiodateien mit einer Länge von mindestens drei Sekunden) Stimmen zu klonen (Peng und Sarazen 2018).

Von Arik et al. wurden zwei Ansätze untersucht: Sprecher-Adaption und Sprecher-Encodierung. Beide Ansätze waren bereits mit wenigen Eingabedaten leistungsstark. Während die Sprecher-Adaption natürlichere und ähnlichere Stimmen erzeugen kann, ist dessen Zeitaufwand deutlich höher. Für Einsätze in ressourcenarmen Umgebungen eignet sich daher die Sprecher-Encodierung besser (Arik et al. 2018).

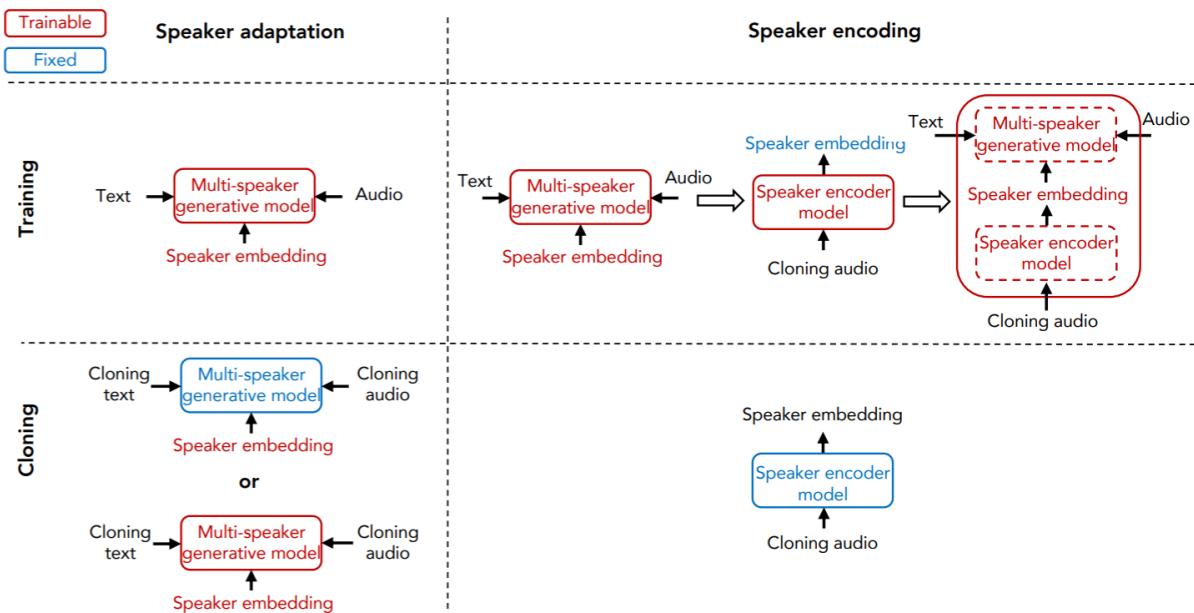


Abb. 4.1: Architekturen der Ansätze von Baidu DeepVoice (Arik et al. 2018, S. 4).

In Abb. 4.1 sind für beide Ansätze die Architekturen und Abläufe für Training und Cloning dargestellt. Die verwendeten Begriffe werden bei der Architektur des Ansatzes im nächsten Abschnitts erklärt, da dieser aufgrund der verfügbaren Implementierung für unsere Untersuchung relevanter ist.

Unter der Adresse <https://audiodemos.github.io/> können Beispiele des Voice Clonings angehört werden. Es ist erkennbar, dass einzelne der Ergebnisse annähernd natürlich klingen, allerdings werden für jede Stimme unterschiedliche Parameter benötigt, um das beste Ergebnis zu erreichen, weshalb es noch keine universelle Möglichkeit gibt, um direkt die natürlichste Stimme zu generieren. Ein Brute-Force Ansatz, der verschiedene Parameter ausprobiert und anschließend die Natürlichkeit des Ergebnisses misst wäre möglich aber zeitaufwändig.

4.1.3 Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis (2018)

Rund zwei Jahre nach *WaveNet* wurde ein neuer Ansatz vorgestellt, der den langen Titel „Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis“ trägt. Dieser ermöglichte erstmals das Klonen einer Stimme mit nur fünf Sekunden Beispiel-Audio-Material (Jia et al. 2018).

Da für diesen Ansatz eine leicht zu installierende Implementation veröffentlicht wurde, wird in diesem Abschnitt nicht nur der theoretische Hintergrund des Ansatzes beschrieben, sondern auch die Implementation getestet (Jemine 2019a).

Erklärung des Ansatzes

Der Ansatz von Jia et al. (Google Inc.) benutzt eine dreiteilige Architektur, die sich in die drei Komponenten Synthesizer, Encoder und Vocoder unterteilen lässt.

Jede dieser Komponenten nutzt ein eigenes neuronales Netz, weshalb auch drei Modelle mithilfe von maschinellem Lernen erlernt werden.

Die Architektur wird in Abb. 4.2 vereinfacht dargestellt und im Folgenden näher beschrieben.

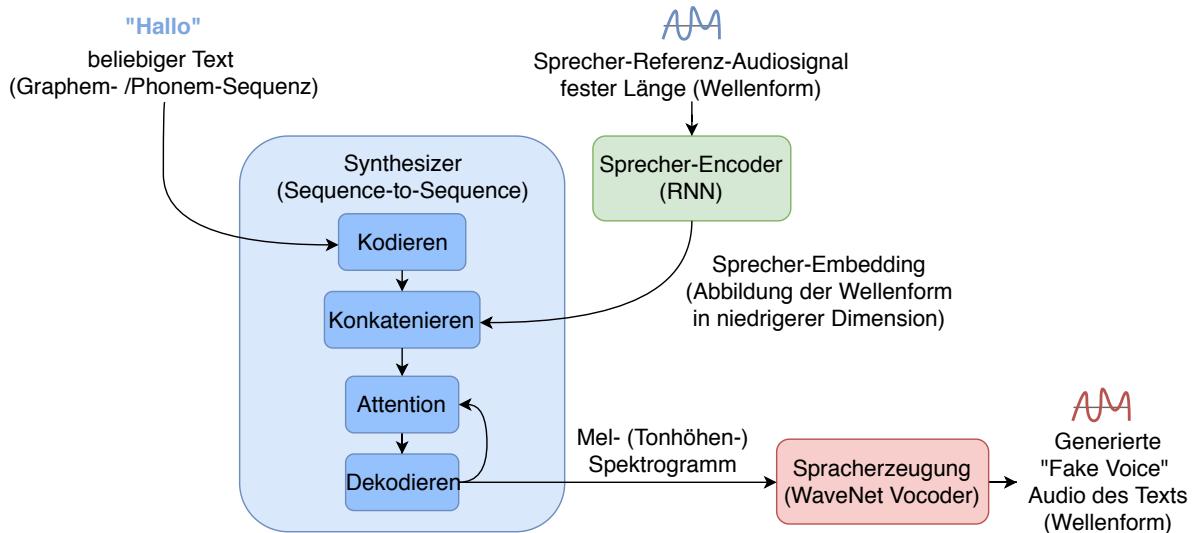


Abb. 4.2: Architektur des Ansatzes von Jia et al. 2018 (nach Jia et al. 2018, S. 3).

Der Sprecher-Encoder basiert auf einem RNN, das eine niedrig-dimensionale Repräsentation des Audiosignals der zu klonenden Stimme erzeugt. Als Metrik wird der „Speaker Verification Loss“ von Wan et al. 2018 genutzt. Dieser basiert darauf, dass Sprachbeispiele von gleichem Sprecher eine ähnliche Cosinus-Ähnlichkeit haben. Das Training erfolgt mit nach Person kategorisierten Sprachsamples ohne ein Transkript. Im Rahmen der Veröffentlichung von Jia et al. wurde mit rund 18.000 eigenen Sprechern trainiert, wobei zusätzlich auch bestehende Datensätze wie LibriSpeech (Panayotov et al. 2015) oder VoxCeleb (Nagrani, Chung und Zisserman 2017) verwendet wurden. (Jia et al. 2018, S. 8)

Der Synthesizer nimmt einen beliebigen geschriebenen Text entgegen und erzeugt für diesen mit Hilfe der Repräsentation des Encoders ein Mel-Tonhöhen-Spektrogramm dieses Textes in der gewünschten Stimme. Er ist trainiert auf Paaren von Text-Transkripte beziehungsweise Phonem-Sequenzen und Audio-Dateien. Für die Repräsentationen (Embeddings) wird ein vortrainiertes Netzwerk benutzt, das mithilfe von Transfer Learning vereinfacht gesagt also zusätzlich neuen Trainingsdaten, verbessert wird. In Abb. 4.3 werden beispielhaft einige der vom Synthesizer erzeugten Mel-Spektrogramme beschrieben

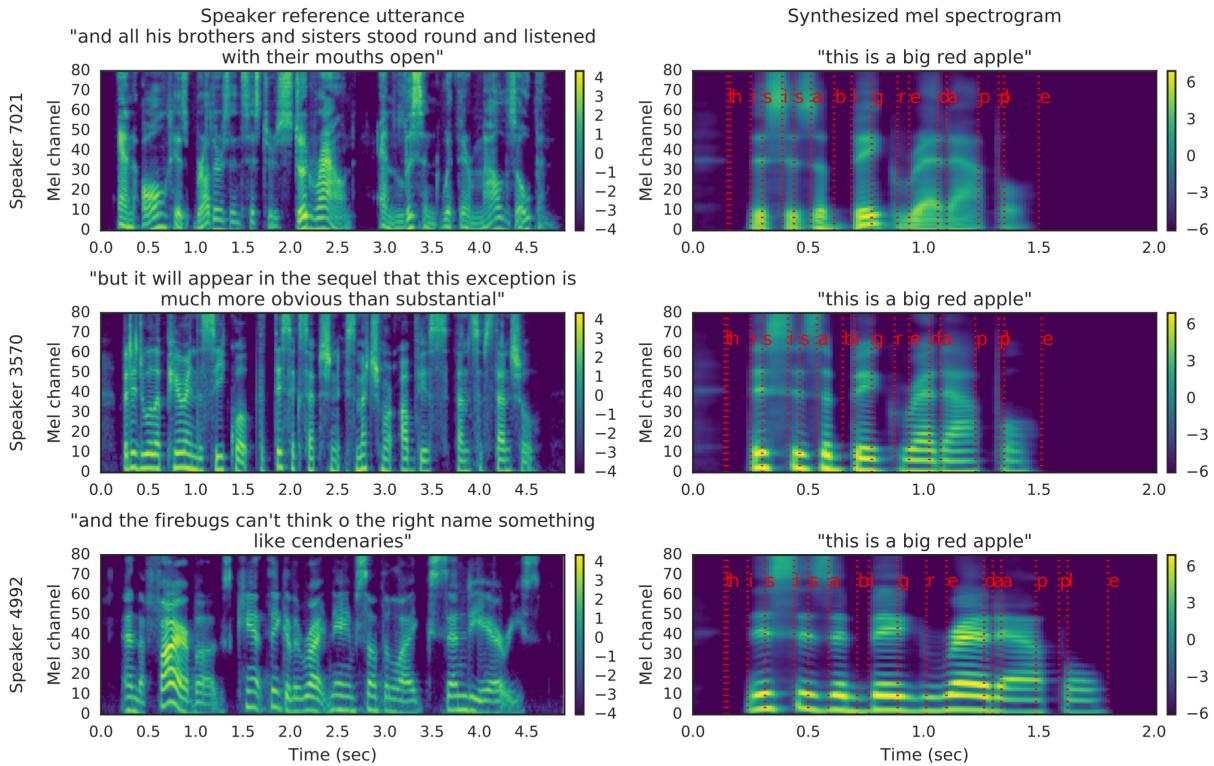


Abb. 4.3: Synthese des gleichen Texts für je 5 Sekunden lange Audiodateien verschiedener Stimmen. Links sind Mel-Spektrogramme die die Input-Stimme repräsentieren (Embeddings) und rechts die Ausgabe-Spektrogramme des Synthesizers. In rot ist die Ausrichtung des Texts zur Sprache dargestellt. Es werden drei Stimmen des Trainingssets dargestellt: Eine männliche (Oben) und zwei weibliche (Mitte und Unten). (Jia et al. 2018, S. 4)

Der Spracherzeuger (Neural Vocoder) wandelt die synthetisierten Mel-Spektrogramme abschließend in hörbare Zeit-Wellenformen beziehungsweise Audio-Dateien um.

Die Qualität der synthetisierten Audiodateien wird zur Evaluation gemessen an der Ähnlichkeit (Similarity) der Ausgabedatei zur Eingabedatei und der Natürlichkeit (Naturalness) der generierten Stimme. Diese Ergebnisse hängen stark von den verwendeten Trainingsdatensätzen ab (Jia et al. 2018, S. 9).

Die Autoren haben in ihrer Veröffentlichung demonstriert, dass die entwickelte Architektur realistische Sprache mit Stimmen außerhalb des Trainingsdatensatzes generieren kann, weshalb geschlussfolgert werden kann, dass das Modell eine realistische Repräsentation des Raums der Stimmvarianten gelernt hat (ebd., S. 9).

Dennoch kann das Modell keine Stimmen auf „menschlichem Niveau“ generieren, da diese nicht natürlich genug klingen. Die Forscher begründen dies darauf, dass zwar mit sehr vielen Stimmen trainiert wurde, für jede einzelne Stimme aber nur ein kleiner Beispielsatz gesprochen wurde. Außerdem

habe das Modell Probleme dabei, einzelne wortspezifische Betonungen von Sprechertypischen Nuancen zu trennen. (Jia et al. 2018, S. 9)

Implementierung des Ansatzes: Voice Cloning Toolbox

Der vorangehend vorgestellte Ansatz wurde von Corentin Jemine im Rahmen seiner Thesis (Jemine 2019a) in Form der *Real-Time Voice Cloning Toolbox* entwickelt und auf GitHub veröffentlicht (Jemine 2019b).

Im Rahmen dieser Arbeit wurde das Programm installiert und getestet, um einen ersten Eindruck zu bekommen, was praktisch bereits mit einfach zur Verfügung stehendem Code möglich ist.

Zur Installation der *Real-Time Voice Cloning Toolbox* wurde zunächst eine virtuelle Python Umgebung (Virtual Environment, venv) eingerichtet. In diese wurden alle in der mitgelieferten Datei requirements.txt genannten Module installiert. Die Verwendung einer virtuellen Umgebung hat den Vorteil, dass bei mehreren Projekten mit verschiedenen Anforderungen auf dem Computer Rechner keine Konflikte hinsichtlich Modulversionen oder sich widersprechenden Modulen entstehen.

Nach der Einrichtung der Python-Umgebung müssen drei vortrainierte Modelle heruntergeladen werden, jeweils eines für den Enkoder, den Synthesizer und den Vocoder. Es besteht grundsätzlich auch die Möglichkeit, diese Modelle selbst zu trainieren, auf Grund der Größe der Datensätze und der daraus entstehenden langen Trainingszeit ist das praktisch aber nicht sinnvoll.

Zum Überprüfen der Installation kann im Installationsordner der Befehl `python demo_cli.py` ausgeführt werden, durch den verschiedene Tests gestartet werden, die die Vollständigkeit und Funktionalität aller Komponenten prüft.

Zum Start der Toolbox genügt ein einfacher Befehl: `python demo_toolbox.py`.

Möchte man mit verschiedenen Stimmen experimentieren, so empfiehlt es sich, einen oder mehrere Datensätze mit Beispielstimmen, wie beispielsweise LibriSpeech (Panayotov et al. 2015), herunterzuladen. Diese Datensätze können beim Start der Toolbox über einen Parameter referenziert werden, um eine Auswahl dieser in der grafischen Benutzeroberfläche zu ermöglichen. (Jemine 2019b)

In Abb. 4.4 ist ein Screenshot der Toolbox nach einiger Interaktion zu sehen. Im Folgenden werden die einzelnen Bereiche der Oberfläche näher erläutert.

Wie bereits in Abb. 4.2 gezeigt, benötigt der Ansatz, und damit auch diese Implementation, als Eingabewerte eine Referenz-Audiodatei der zu klonenden Stimme sowie einen beliebigen geschriebenen, vorzulesenden, Text.

Die Audiodatei kann bei der Toolbox entweder aus einem Datensatz wie LibriSpeech (Panayotov et al. 2015) geladen werden, von einem beliebigen Ort auf der Festplatte importiert werden oder direkt innerhalb der Toolbox mithilfe eines Mikrofons aufgezeichnet werden. Die Konfiguration hierfür befindet sich in Abb. 4.4 in der linken oberen Ecke. Das aus der Stimme und dem Beispiel-Audio-Sample generierte Mel-Spekrogramm wird an zweitunterster Stelle angezeigt, allerdings nicht für die folgenden Berechnungen verwendet (Jemine 2019a, S. 31).

In der rechten Hälfte der Oberfläche befindet sich oben die Möglichkeit, einen beliebigen Text einzugeben, der in der ausgewählten Stimme vorgelesen werden soll.

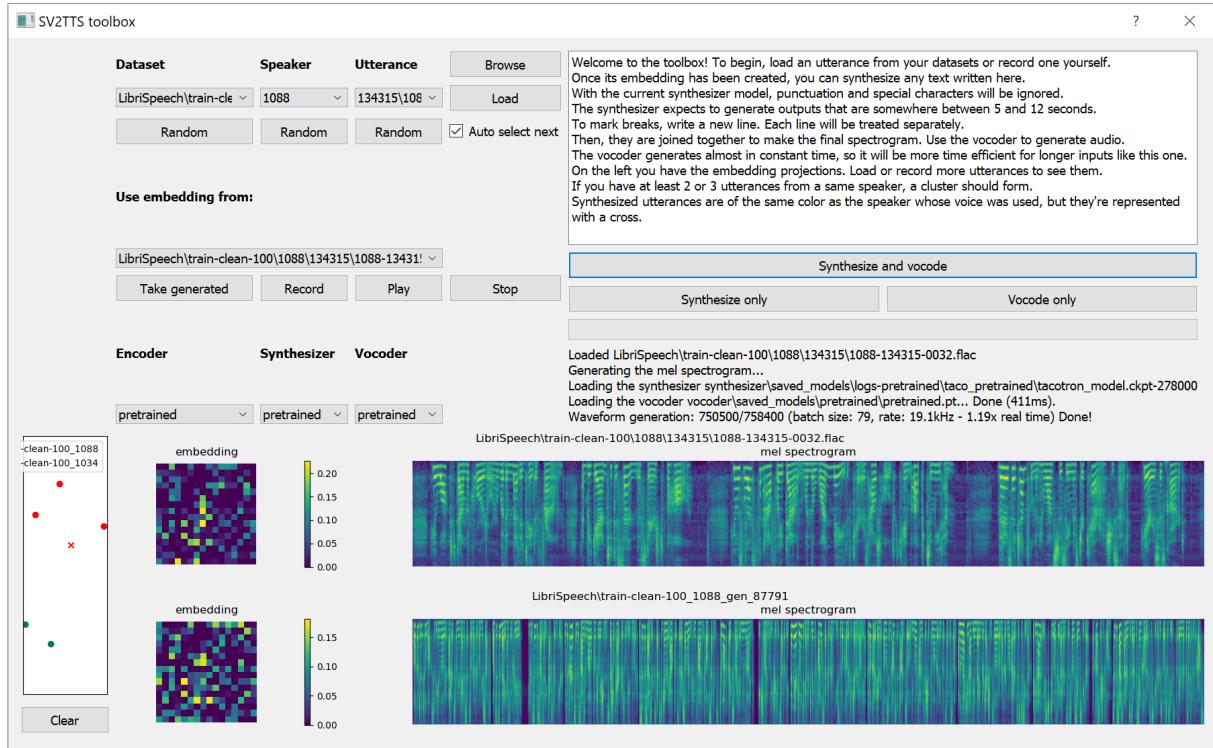


Abb. 4.4: Screenshot der Voice Cloning Toolbox von Jemine.

Wie im vorherigen Abschnitt beschrieben, werden nach dem Generieren eines Embeddings durch den Encoder noch zwei Neuronale Netze durchlaufen. Zunächst den Synthesizer, der für die gegebene Stimme und den Text ein Mel-Diagramm generiert, und den Vocoder, der aus diesem schlussendlich Sprache erzeugt. Dieses wird an unterster Stelle angezeigt.

Nach Durchlauf des Vocoders wird unten links eine Pixelgrafik angezeigt, die das Embedding der generierten Wellenform repräsentiert, und die Wiedergabe des generierten Audiofiles startet.

Noch weiter links wird eine Projektion der verschiedenen Embeddings in einen zweidimensionalen Raum generiert, um die Ähnlichkeit verschiedener Stimmen bzw. Audiosamples miteinander zu vergleichen.

Für die Dimensionalitätsreduktion wird das UMAP Verfahren (McInnes, Healy und Melville 2018) verwendet (Jemine 2019a, S. 31).

Durch diese Grafiken bietet die Toolbox eine Methode zum Nachvollziehen der generierten Ergebnisse und der einzelnen Arbeitsschritte.

Die *Real-Time Voice Cloning Toolbox* bietet dank der einfachen Installation und der grafischen Oberfläche eine gute und schnelle Möglichkeit Voice Cloning selber auszuprobieren, ohne auf kommerzielle Lösungen zurückgreifen zu müssen.

Im Rahmen dieser Arbeit wurde die Synthese mehrerer Beispielsätze anhand verschiedener Eingabedateien und Stimmen getestet. Es ist aufgefallen, dass jede der generierten Dateien ohne weiteres direkt als Fake identifiziert werden konnte, weshalb hier kein Test mithilfe einer wissenschaftlichen Methode durchgeführt wurde (wie beispielsweise die Ermittlung eines „Mean Opinion Score“).

Beispiele der generierten Dateien befinden sich im digitalen Anhang dieser Seminararbeit, in der begleitenden Präsentation und unter <https://github.com/AlexanderMelde/VoiceCloning>.

4.1.4 MelNet (2019)

Die neueste wissenschaftliche Veröffentlichung, die in dieser Arbeit untersucht wird, stammt vom Juni 2019. Vasquez und Lewis (Facebook AI Research) bauen mit ihrem Ansatz *MelNet* grundsätzlich auf der Idee von *WaveNet* auf, verbessert dieses aber durch einige grundlegende Verbesserungen.

MelNet lernt Audiostrukturen auf verschiedenen Zeitskalen, weshalb es sowohl lokale Wellenformstrukturen erfassen kann (Zoomed-In), die für eine hohe Wiedergabetreue (Fidelity) sorgen, als auch Abhängigkeiten über lange Zeiträume hinweg erfassen kann (Zoomed-Out), wodurch komponistische Strukturen und Betonungen ganzer Sätze erfasst werden.

Statt einem eindimensionalen Wellenform in der Zeitdimension wird bei *MelNet* eine zweidimensionale Zeit-Frequenz-Repräsentation, das Spektrogramm, ausgewertet. Dies hat den Vorteil, dass Strukturen auf verschiedenen Zeitskalen erkannt und ausgewertet werden können (siehe Abb. 4.5).

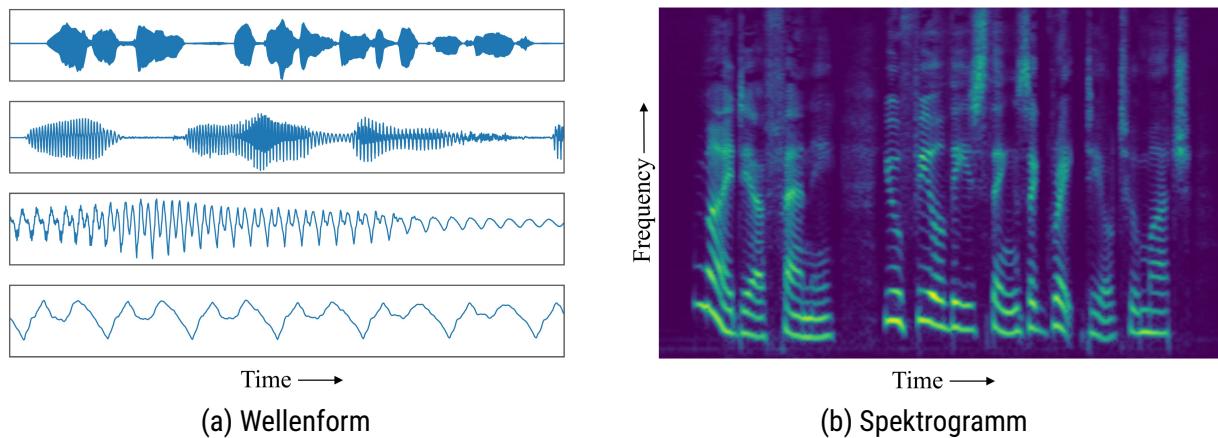


Abb. 4.5: Darstellung eines vier-sekündigen Audiosignals bei *MelNet* als Wellenform und Spektrogramm. Es ist erkennbar, dass bei einer Wellenform Strukturen auf verschiedenen Zeitskalen ganz unterschiedlich aussehen, während das Spektrogramm eine Struktur hat die über die gesamte Fläche fließend verteilt ist. (Vasquez und Lewis 2019, S. 2)

Das maschinelle Lernen erfolgt bei *MelNet* mithilfe eines RNN, also eines Netzes, das besonders gut für Zeitserien geeignet ist. Die optimalen Parameter werden wie üblich mittels Backpropagation und Gradient Descent optimiert (siehe 2.2.5).

Nach Aussage der Forscher generiert *WaveNet* zwar Audiodateien mit einer höheren Wiedergabetreue (Fidelity), *MelNet* ist aber überlegen beim Erfassen von Strukturen höherer Schichten, wie beispielsweise die subtilen Konsistenzen einer Stimme über mehrere Wörter hinweg.

Trotzdem kann das Modell eine menschliche Stimme immer noch nicht zuverlässig über einen längeren Zeitraum hinweg imitieren. Hierfür fehlen vor allem Betonungen von Wörtern oder Sätzen und das Aufbauen von Spannung über den Text einer ganzen Seite hinweg.

Dies sind die gleichen Nachteile, die auch die Generierung von natürlich klingendem Text mittels KI aktuell hat, da auch dort nur oberflächliche kurze Abhängigkeiten, und wenige Langzeitstrukturen erfasst werden.

4.2 Kommerzielle Angebote

Es gibt bisher nur wenige kommerzielle Verwendungen der vorgestellten wissenschaftlichen Ansätze. Im Rahmen dieser Arbeit konnten zwei Produkte identifiziert werden, die Voice Cloning als zentrales Element einsetzen und Endnutzern zur Verfügung stellen.

4.2.1 Adobe Project VoCo

Adobe hat im Jahr 2016 auf der firmeneigenen Konferenz „Adobe MAX 2016“ das Projekt VoCo vorgestellt (Adobe Creative Cloud 2016). Dabei handelte es sich um einen frühen Prototypen einer Software, die das „Photoshop für Audio-Dateien“ werden sollte (Gault 2016).

Im darauf folgenden Jahr wurde das zugehörige Paper von veröffentlicht (Jin, Mysore et al. 2017). In diesem lässt sich erkennen, dass der Ansatz zunächst ein klassisches Text-To-Speech Verfahren verwendet und anschließend die Stimme mittels einer „Voice Conversion“ zur gewünschten Stimme konvertiert wird (ebd., S. 3–4). Für die Konvertierung kommt CUTE zum Einsatz, das auf einer konkatenativen Synthese mit *unit selection* basiert (Jin, Mysore et al. 2017, S. 4–5; Jin, Finkelstein et al. 2016, S. 1).

Der Vorgang der *unit selection* ist in Abb. 4.6 dargestellt.

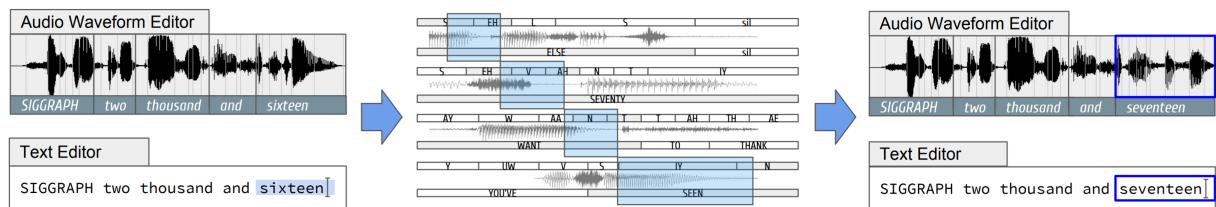


Abb. 4.6: Ersetzen von Wörtern mit Adobe VoCo. Im Beispiel wird das Wort *sixteen* durch das Wort *seventeen* ersetzt. Letzteres wird durch die Konkatenation von Teilen anderer Wörter (*else*, *seventy*, *want* und *seen*) generiert (Jin, Mysore et al. 2017, S. 1).

Wie bereits in Abb. 4.6 angedeutet, basiert VoCo auf dem Editieren von Text zur Generierung geklonter Sprache. Die grafische Oberfläche von VoCo ist in Abb. 4.7 dargestellt.

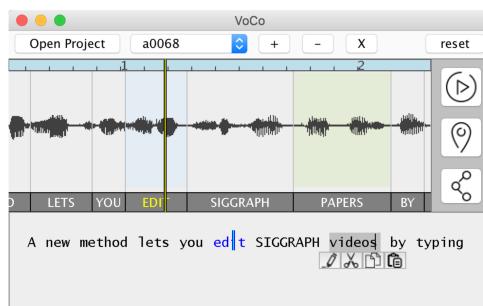


Abb. 4.7: Grafische Oberfläche von Adobe VoCo. Es können nicht nur einzelne Teile des Textes und der zugehörigen Sprache verschoben und ausgeschnitten werden, sondern auch neue Wörter eingefügt werden, deren Audio mit Voice Cloning generiert wird (Jin, Mysore et al. 2017, S. 3).

Bis heute wurde VoCo als Software allerdings nicht veröffentlicht. Um Täuschungen und Manipulationen zu verhindern, scheint Adobe damals eine Art Wasserzeichen vorgesehen zu haben, wobei

im gleichen Atemzug auch erwähnt wurde, dass sie daran arbeiten, die geklonten Stimmen erkennbar zu machen (Gault 2016). Ob es nun noch zu keiner Veröffentlichung kam, weil die Ethikfrage nicht geklärt wurde, oder weil das Klonen als solches keine ausreichende Qualität erreicht hat lässt sich nur mutmaßen. Auch eine stille Einstellung des Projekts aus vertrieblichen Gründen könnte eine Erklärung sein.

4.2.2 Descript Overdub

Die Schnitt-Software für Podcast-Autoren Descript bietet mit dem Feature *Overdub* die Möglichkeit, im Nachhinein einzelne in der Audioaufnahme gesprochene Wörter gegen andere auszutauschen, ohne dass diese eingesprochen werden müssen. Sie bildet damit die komplette Funktionalität des im Vorangegangenen Abschnitt beschriebenen Projekts VoCo ebenfalls ab (Mason 2019).

Die grafische Benutzeroberfläche der Podcasting-Software Descript ist in Abb. 4.8 dargestellt.

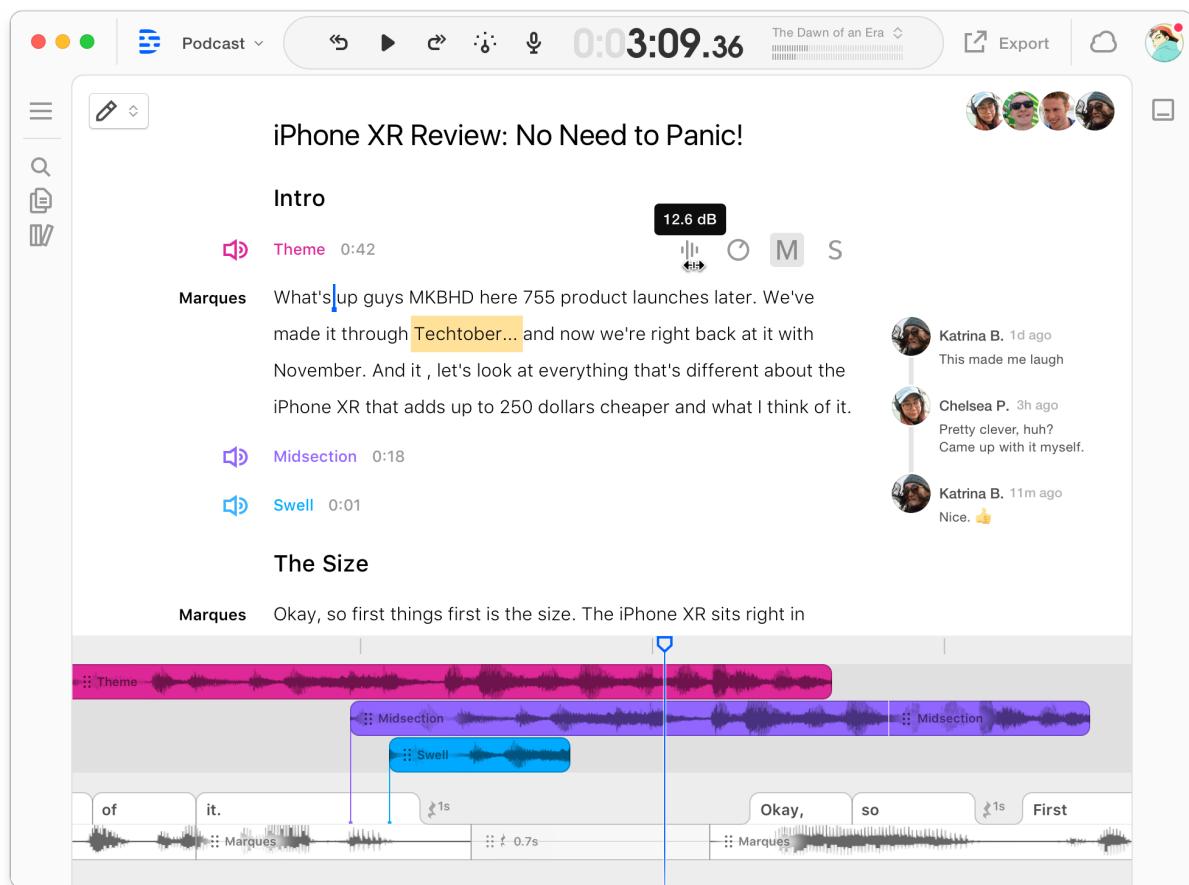


Abb. 4.8: Grafische Oberfläche von Descript. Für die Verknüpfung von Sprache und Text wird ein Speech-To-Text Verfahren (siehe Abschnitt 2.1) eingesetzt (Mason 2019).

Die Funktion *Overdub* ist durch den Kauf des Startups *Lyrebird* entstanden. Lyrebird stellte einen Webservice bereit, bei dem Nutzer kostenlos Ihre eigene Stimme aufnehmen konnten, um anschließend Sätze mit dieser Stimme generieren zu lassen. Dieser Dienst wurde im Rahmen der Integration in Descript eingestellt. Während die meisten Funktionalitäten von Descript mit einem kostenlosen oder kostenpflichtigen Abonnement genutzt werden können, befindet sich das *Overdub* Feature

derzeit in einer geschlossenen Beta-Phase. Im Rahmen dieser Arbeit wurde eine Anfrage zur Aufnahme in das Beta-Programm gestellt, allerdings gab es im Zeitraum dieser Arbeit noch zu keiner Rückmeldung des Herstellers.

Hinweis: Der Autor hat Lyrebird bereits 2018 privat getestet, und die Ergebnisse schienen besser, als was in dieser Arbeit mit der Voice Cloning Toolbox generiert werden konnte. Da aufgrund der Einstellung des Webservices das Experiment nicht mehr wiederholt werden kann, wird Lyrebird im Rahmen dieser Arbeit nicht weiter berücksichtigt.

Descript *Overdub* generiert für jedes zu ersetzenende Wort drei Audiodateien: Den Präfix, das Wort selber und den Suffix. Durch dieses Verfahren wird nicht nur die Audiodatei des jeweiligen Wortes, sondern auch die Audiodateien der Wörter direkt vor und nach dem zu ersetzenen Wörter angepasst (siehe Abb. 4.9, Brébisson 2019).

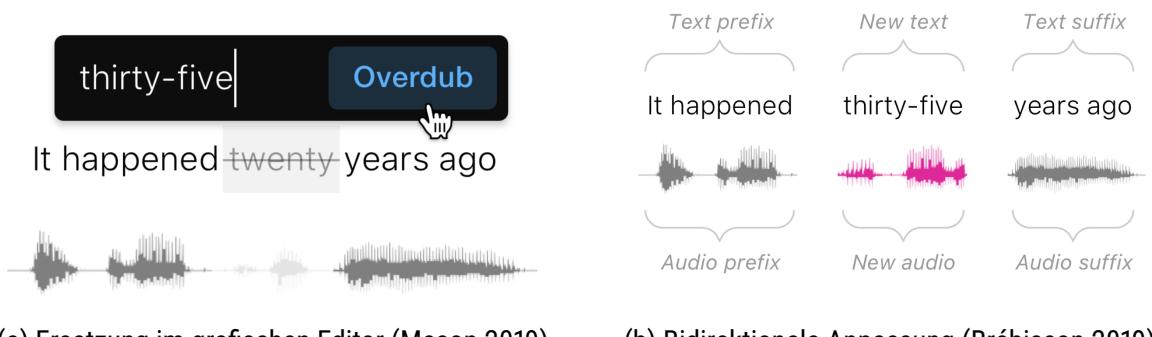


Abb. 4.9: Ersetzen von Wörtern mit Descript *Overdub*.

Für die Generierung der Audiosequenzen werden je zwei klassische neuronale Netze (*feed forward*) eingesetzt, von denen eines die Länge von Phonemen abschätzt und das andere die Audiosequenz unter Zuhilfenahme eines der bekannten Text-To-Speech Ansätze generiert. Anschließend wird mithilfe eines Attention-Netzwerks ermittelt, an welcher Stelle die jeweiligen Audiosequenzen geschnitten und konateniert werden sollen, um möglichst natürlich zu klingen (Brébisson 2019).

Da Voice Cloning wie bereits in Abschnitt 3.2 beschrieben auch missbräuchlich verwendet werden könnte, hat Descript zu *Overdub* eine Stellungnahme zur Ethik veröffentlicht (Descript 2019a). In dieser wird unter anderem betont, dass das Einverständnis jedes Sprechers explizit eingeholt werde. Es wird auch darauf hingewiesen, dass die dahinterliegende Grundlagenforschung öffentlich verfügbar ist, weshalb davon auszugehen sei, dass sich andere zukünftige Produkte keine oder weniger derartige ethische Restriktionen auferlegen. Mit einer Referenz auf den Deepfake-Detektor von Facebook (Schroepfer 2019) wird abschließend betont, dass man mit führenden Wissenschaftlern im Bereich des maschinellen Lernens, Ethik-Professoren und Personen der Öffentlichkeit im Gespräch sei, um die besten Methoden zur Weiterentwicklung und Veröffentlichung der Technologie sicherzustellen (Descript 2019a).

Auf der Website von Descript kann *Overdub* für eine voreingestellte Auswahl von Sprechern und Sätzen ausprobiert werden (Descript 2019b). Im Rahmen dieser Arbeit wurden verschiedene Beispielsätze generiert, die wie die Ergebnisse der Voice Cloning Toolbox (siehe Abschnitt 4.1.3) im digitalen Anhang dieser Seminararbeit, in der begleitenden Präsentation und unter <https://github.com/AlexanderMelde/VoiceCloning> abgerufen werden können.

4.3 Vergleich

Anhand der wissenschaftlichen Veröffentlichungen kann erkannt werden, dass schnelle Fortschritte gemacht werden. An den kommerziellen Produkten und den in Abschnitt 3.1 vorgestellten Chancen ist erkennbar, dass auch aus wirtschaftlicher Sicht eine Nachfrage nach dieser Technologie existiert.

Alle untersuchten Ansätze haben gemein, dass sie eine Kombination aus Techniken der digitalen Sprachverarbeitung und des maschinellen Lernen verwenden.

Die kommerziellen Produkte erzeugten in den Tests natürlich klingendere Ergebnisse als die Voice Cloning Toolbox, die stellvertretend für die wissenschaftlichen Ansätze mit Open-Source-Implementationen getestet wurde. Hierbei muss allerdings beachtet werden, dass es sich bei den vorbereiteten Beispiele der kommerziellen Anwendungen möglicherweise um besonders gute Beispiele handelt, und auch durch die serverseitige Generierung stehen diesen Ansätzen vermutlich größere Ressourcen zur Verfügung.

Aufgrund des Budgets dieser Arbeit, geschlossenen Quelltexten und dem mit einem Test verbundenem Zeitbedarf konnten nicht alle genannten Ansätze und Produkte getestet werden. Darüber hinaus gibt es die Schwierigkeit, das Einzelne der Funktionen sich bei den kommerziellen Produkten noch in einem geschlossenem Beta-Test befinden, weshalb ein Test nur auf Einladung der Anbieter möglich wäre.

Bei der Untersuchung der kommerziellen Angebote ist aufgefallen, dass auch für diese die Ethik-Frage eine wichtige Rolle zu spielen scheint, da in den begleitenden Schreiben stets auch diese Thematik berücksichtigt wurde.

5 Fazit

Die Modelle zur Generierung von Sprache können bereits gute Ergebnisse erzielen und es ist abzusehen, dass mit zukünftigen Produkten und Veröffentlichungen realistische Imitationen menschlicher Sprache möglich sind.

In der praktischen Implementierung hat sich gezeigt, dass es bei der Toolbox aktuell bei einigen Stimmen noch deutlich hörbare Unterschiede zwischen der digital erzeugten Stimme und dem echten Vorbild gibt, in der kommerziellen Lösung von Descript ist man dem Anbieter und seinen Demos zufolge aber schon deutlich weiter.

Die meisten Veröffentlichungen basieren auf englischen Datensätzen, nur in einem Fall wurde auch einer in Mandarin verwendet (Oord et al. 2016, S. 1). Für weitere Sprachen wurden noch nicht ausreichend genügend große Datensätze gesammelt, weshalb das Eingeben eines fremdsprachigen Textes in ein auf englische Stimmen trainiertes Modell meist keine zufriedenstellenden realitätsnahen Ergebnisse erzeugt.

Die Anfangs aufgestellte These, „Voice Cloning ist möglich und bereits heute eine Gefahr“, kann daher bestätigt werden, da mit genügend Aufwand und bei Verwendung der richtigen Techniken bereits Täuschungen möglich sind (Brown 2019). Es scheint nur noch eine Frage der Zeit zu sein, bis auch die natürlicher klingenden Implementationen leichter zugänglich gemacht und bestehende Modelle noch weiter verbessert werden.

Literaturverzeichnis

- Adobe Creative Cloud (Nov. 2016). #VoCo. *Adobe Audio Manipulator Sneak Peak with Jordan Peele*. URL: <https://youtu.be/I3l4XLZ59iw> (besucht am 12. 08. 2019) (ref. auf S. 25).
- Agarap, Abien Fred (2018). „Deep Learning using Rectified Linear Units (ReLU)“. In: *arXiv preprint arXiv:1803.08375* (ref. auf S. 10).
- Arik, Sercan et al. (2018). „Neural Voice Cloning with a Few Samples“. In: *Advances in Neural Information Processing Systems*, S. 10019–10029 (ref. auf S. 19).
- Bishop, C.M. (2016). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer New York. ISBN: 9781493938438 (ref. auf S. 12).
- Boser, Bernhard E, Isabelle M Guyon und Vladimir N Vapnik (1992). „A training algorithm for optimal margin classifiers“. In: *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, S. 144–152 (ref. auf S. 7).
- Brébisson, Alexandre de (Sep. 2019). „How Imputations Work: The Research Behind Overdub“. In: URL: <https://www.descript.com/post/how-imputations-work-the-research-behind-overdub> (besucht am 14. 11. 2019) (ref. auf S. 27).
- Brown, Jennings (Sep. 2019). „Scammer Successfully Deepfaked CEO's Voice To Fool Underling Into Transferring \$243,000“. In: *Gizmodo*. URL: <https://gizmodo.com/scammer-successfully-deepfaked-ceos-voice-to-fool-under-1837835066> (besucht am 03. 11. 2019) (ref. auf S. 1, 17, 29).
- Cho, Kyunghyun et al. (2014). „On the properties of neural machine translation: Encoder-decoder approaches“. In: *arXiv preprint arXiv:1409.1259* (ref. auf S. 14).
- Cyffka, Andreas (2007). *PONS Kompaktwörterbuch: Deutsch als Fremdsprache*. Pons (ref. auf S. 6).
- Deng, Jia et al. (2009). „ImageNet: A large-scale hierarchical image database“. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, S. 248–255 (ref. auf S. 6).
- Descript (Okt. 2019a). „Descript Ethics Statement“. In: URL: <https://www.descript.com/ethics> (besucht am 30. 10. 2019) (ref. auf S. 27).
- (Okt. 2019b). „Lyrebird AI“. In: URL: <https://www.descript.com/lyrebird-ai> (besucht am 30. 10. 2019) (ref. auf S. 6, 27).
- Donahue, Jeffrey et al. (2015). „Long-term recurrent convolutional networks for visual recognition and description“. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634 (ref. auf S. 12–14).
- Dudley, Homer, R.R. Riesz und S.S.A. Watkins (1939). „A synthetic speaker“. In: *Journal of the Franklin Institute* 227.6, S. 739–764. URL: <http://www.sciencedirect.com/science/article/pii/S0016003239908161> (besucht am 03. 12. 2019) (ref. auf S. 4, 5).
- Elman, Jeffrey L (1990). „Finding structure in time“. In: *Cognitive science* 14.2, S. 179–211 (ref. auf S. 13).
- Flanagan, James L. (1972). *Speech Analysis Synthesis and Perception*. Springer Berlin Heidelberg. ISBN: 978-3-662-01562-9 (ref. auf S. 5).
- Gault, Matthew (Nov. 2016). „After 20 Minutes of Listening, New Adobe Tool Can Make You Say Anything“. In: *VICE*. URL: https://www.vice.com/en_us/article/jpgkxp/after-20-minutes-of-listening-new-adobe-tool-can-make-you-sayanything (besucht am 08. 12. 2019) (ref. auf S. 25, 26).
- Google Cloud (Dez. 2019). „WaveNet und andere synthetische Stimmen“. In: URL: <https://cloud.google.com/text-to-speech/docs/wavenet?hl=de> (besucht am 06. 12. 2019) (ref. auf S. 18).

- Graves, Alex und Navdeep Jaitly (2014). „Towards end-to-end speech recognition with recurrent neural networks“. In: *International Conference on Machine Learning*, S. 1764–1772 (ref. auf S. 14).
- Grossman, Wendy M. (Sep. 2011). „Getting Voice: New Speech Synthesis Could Make Roger Ebert Sound More Like Himself“. In: *Scientific American*. URL: <https://www.scientificamerican.com/article/new-speech-synthesis-could-make-robert-ebert-sound-more-like-himself/> (besucht am 04.12.2019) (ref. auf S. 16).
- Hochreiter, Sepp und Jürgen Schmidhuber (1997). „Long short-term memory“. In: *Neural computation* 9.8, S. 1735–1780 (ref. auf S. 14).
- IMDb (2019). *Terminator 2: Tag der Abrechnung (1991)* - Plot Summary Synopsis. URL: <https://www.imdb.com/title/tt0103064/plotsummary> (besucht am 12.11.2019) (ref. auf S. 1).
- Jemine, Corentin (Juli 2019a). „Automatic Multispeaker Voice Cloning. (Unpublished master's thesis)“. In: *Université de Liège, Belgique*. URL: <https://matheo.uliege.be/handle/2268.2/6801> (besucht am 22.10.2019) (ref. auf S. 1, 20, 22, 23).
- (Okt. 2019b). *Real-Time Voice Cloning*. Version 3e72a7d. URL: <https://github.com/CorentinJ/Real-Time-Voice-Cloning> (besucht am 22.10.2019) (ref. auf S. 1, 22, 23).
- Ji, Shuiwang et al. (2013). „3D convolutional neural networks for human action recognition“. In: *IEEE transactions on pattern analysis and machine intelligence* 35.1, S. 221–231 (ref. auf S. 12).
- Jia, Ye et al. (2018). „Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis“. In: *Advances in neural information processing systems*, S. 4480–4490 (ref. auf S. 1, 20–22).
- Jin, Zeyu, Adam Finkelstein et al. (2016). „CUTE: A concatenative method for voice conversion using exemplar-based unit selection“. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, S. 5660–5664 (ref. auf S. 25).
- Jin, Zeyu, Gautham J. Mysore et al. (Juli 2017). „VoCo: Text-based Insertion and Replacement in Audio Narration“. In: *ACM Transactions on Graphics* 36.4 (ref. auf S. 25).
- Kapur, Rohan (März 2013). „Rohan #4: The vanishing gradient problem“. In: *A Year of Artificial Intelligence*. URL: <https://ayearofai.com/ec68f76ffb9b> (besucht am 23.08.2018) (ref. auf S. 10, 14).
- Kapur, Rohan und Lenny Khazan (Apr. 2017). „Rohan & Lenny #3: Recurrent Neural Networks & LSTMs“. In: *A Year of Artificial Intelligence*. URL: <https://ayearofai.com/10300100899b> (besucht am 22.08.2018) (ref. auf S. 13, 14).
- Karn, Ujjwal (Aug. 2016). „An Intuitive Explanation of Convolutional Neural Networks“. In: *the data science blog*. URL: <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/> (besucht am 19.06.2018) (ref. auf S. 11, 12).
- Karpathy, Andrej (Mai 2015). „The Unreasonable Effectiveness of Recurrent Neural Networks“. In: *Andrej Karpathy blog*. URL: <http://karpathy.github.io/2015/05/21/rnn-effectiveness> (besucht am 19.06.2018) (ref. auf S. 13, 14).
- Karpathy, Andrej et al. (2014). „Large-scale video classification with convolutional neural networks“. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, S. 1725–1732 (ref. auf S. 12).
- Kempelen, Wolfgang von (1791). *Mechanismus der menschlichen Sprache*. Wien: Deutsches Textarchiv. URL: http://www.deutsches-textarchiv.de/kempelen-maschine_1791 (besucht am 03.12.2019) (ref. auf S. 4).
- Krizhevsky, Alex, Ilya Sutskever und Geoffrey E. Hinton (2012). „ImageNet Classification with Deep Convolutional Neural Networks“. In: *Advances in Neural Information Processing Systems* 25. Hrsg. von F. Pereira et al. Curran Associates, Inc., S. 1097–1105 (ref. auf S. 6, 12).
- Lippmann, Richard (1987). „An introduction to computing with neural nets“. In: *IEEE Assp magazine* 4.2, S. 4–22 (ref. auf S. 6, 8–10).
- Mason, Andrew (Sep. 2019). „Introducing Descript Podcast Studio & Overdub“. In: URL: <https://medium.com/descript/introd>

- ucing-descript-podcast-studio-13 b8da53311b (besucht am 08.12.2019) (ref. auf S. 26, 27).
- McInnes, Leland, John Healy und James Melville (2018). „UMAP: Uniform manifold approximation and projection for dimension reduction“. In: *arXiv preprint arXiv:1802.03426* (ref. auf S. 23).
- Melde, Alexander (Sep. 2018). *Erkennung menschlicher Handlungen durch Auswertung der Körperhaltungen von Personen in einem Video mithilfe von Machine Learning und neuronalen Netzen*. URL: <https://melde.net/portfolio/handlungserkennung> (besucht am 07.11.2019) (ref. auf S. 6).
- Melde, Alexander und Stefan Schneider (Juni 2018). *Verkehrszeichenerkennung*. Version 77e029f. URL: <https://github.com/AlexanderMelde/Verkehrszeichenerkennung> (besucht am 14.07.2018) (ref. auf S. 8).
- Mihkla, Meelis (März 2008). *Kõne ajalise struktuuri modelleerimine eestikeelsele tekst-kõne sünteesile / Modelling the Temporal Structure of Speech for the Estonian Text-To-Speech Synthesis*. ISBN: 978-9949-11-798-7 (ref. auf S. 4).
- Murphy, Kevin (2012). *Machine learning: a probabilistic perspective*. Cambridge: MIT Press (ref. auf S. 6, 7).
- Nagrani, Arsha, Joon Son Chung und Andrew Zisserman (2017). „VoxCeleb: a large-scale speaker identification dataset“. In: *arXiv preprint arXiv:1706.08612* (ref. auf S. 20).
- O'Neill, Natalie (März 2016). „Companies Want to Replicate Your Dead Loved Ones With Robot Clones“. In: *VICE*. URL: https://www.vice.com/en_us/article/pgkgby/companies-want-to-replicate-your-dead-loved-ones-with-robot-clones (besucht am 21.11.2019) (ref. auf S. 17).
- Olah, Christopher (Aug. 2015). „Understanding LSTM Networks“. In: *colah's blog*. URL: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/> (besucht am 19.06.2018) (ref. auf S. 14).
- Oord, Aäron van den et al. (2016). „WaveNet: A Generative Model for Raw Audio“. In: *arXiv preprint arXiv:1609.03499* (ref. auf S. 6, 18, 29).
- Panayotov, Vassil et al. (2015). „LibriSpeech: an ASR corpus based on public domain audio books“. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, S. 5206–5210 (ref. auf S. 20, 22).
- Patel, Jigar et al. (2015). „Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques“. In: *Expert Systems with Applications* 42.1, S. 259–268 (ref. auf S. 6).
- Peng, Tony und Michael Sarazen (Aug. 2018). „Baidu AI Can Clone Your Voice in Seconds“. In: URL: <https://medium.com-synced-review/baidu-ai-can-clone-your-voice-in-seconds-93558a7b984f> (besucht am 08.12.2019) (ref. auf S. 19).
- Pfister, Beat und Tobias Kaufmann (2017). *Sprachverarbeitung. Grundlagen und Methoden der Sprachsynthese und Spracherkennung*. Springer. ISBN: 9783662528372 (ref. auf S. 3).
- Rabiner, Lawrence R und Ronald W Schafer (2011). *Theory and applications of digital speech processing*. Bd. 64. Pearson Upper Saddle River, NJ (ref. auf S. 4).
- Rashid, Tariq (2017). *Neuronale Netze selbst programmieren: Ein verständlicher Einstieg mit Python*. O'Reilly (ref. auf S. 6, 8–10).
- Schroepfer, Mike (Sep. 2019). „Creating a data set and a challenge for deepfakes“. In: *facebook AI*. URL: <https://ai.facebook.com/blog/deepfake-detection-challenge/> (besucht am 03.11.2019) (ref. auf S. 17, 27).
- Smirnov, Evgeny A, Denis M Timoshenko und Sergei N Andrianov (2014). „Comparison of regularization methods for imagenet classification with deep convolutional neural networks“. In: *Aasri Procedia* 6, S. 89–94 (ref. auf S. 12).
- Spektrum Akademischer Verlag (1998). „Formant“. In: *Lexikon der Physik*. URL: <https://www.spektrum.de/lexikon/physik/formant/5219> (besucht am 04.12.2019) (ref. auf S. 4).
- Srivastava, Nitish, Elman Mansimov und Ruslan Salakhudinov (2015). „Unsupervised learning of video representations using LSTMs“. In:

- International conference on machine learning*, S. 843–852 (ref. auf S. 6, 14).
- Su, Yu et al. (2019). „Environment sound classification using a two-stream cnn based on decision-level fusion“. In: *Sensors* 19.7, S. 1733 (ref. auf S. 12).
- Susto, Gian Antonio et al. (2015). „Machine learning for predictive maintenance: A multiple classifier approach“. In: *IEEE Transactions on Industrial Informatics* 11.3, S. 812–820 (ref. auf S. 6).
- Sutskever, Ilya, Oriol Vinyals und Quoc V Le (2014). „Sequence to sequence learning with neural networks“. In: *Advances in neural information processing systems*, 3104–3112 (ref. auf S. 14).
- Tran, Du et al. (2015). „Learning spatiotemporal features with 3D convolutional networks“. In: *Proceedings of the IEEE international conference on computer vision*, 4489–4497 (ref. auf S. 12).
- Vasquez, Sean und Mike Lewis (2019). „MelNet: A Generative Model for Audio in the Frequency Domain“. In: *arXiv preprint arXiv:1906.01083* (ref. auf S. 1, 6, 24).
- Velasquez, Jonathan (Nov. 2016). *macro photography of silver and black studio microphone condenser*. URL: <https://unsplash.com/photos/c1ZN57GfDB0> (besucht am 30.10.2010) (ref. auf S. II).
- Wan, Li et al. (2018). „Generalized end-to-end loss for speaker verification“. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, S. 4879–4883 (ref. auf S. 20).
- Wölfel, Matthias (Nov. 2019). *Profile of Prof. Dr. Matthias Wölfel*. URL: https://www.researchgate.net/profile/Matthias_Woelfel (besucht am 03.11.2019) (ref. auf S. 1).