RUHR
UNIVERSITÄT
BOCHUM
RUB

Adversarially Robust Speech and Speaker Recognition

Dissertation

zur Erlangung des Grades eines Doktor-Ingenieurs
der Fakultät für Elektrotechnik und Informationstechnik
an der Ruhr-Universität Bochum

vorgelegt von

Lea Schönherr
geboren in Würzburg

Bochum, April 2021

Tag der mündlichen Prüfung: 19. Mai 2021

Gutachter:	Prof. Dr.-Ing. Dorothea Kolossa Ruhr-Universität Bochum
Zweigutachter:	Prof. Dr. Thorsten Holz Ruhr-Universität Bochum

Abstract

Speech and speaker recognition systems are integrated into our everyday life; voice assistants answer questions, set timers, or play music, but also send personal messages, control smart homes, and place orders online. They constantly capture and analyze their surrounding environment, making them a gateway for potential attacks. Many security and privacy concerns arise from their built-in *Automatic Speech Recognition* (ASR) system, including *adversarial examples* and *spoofing attacks.* Adversarial examples, which sound like benign audio for human listeners, are interpreted by the ASR system as an attacker-chosen, malicious speech command. Spoofing attacks that imitate a victim's voice can fool speaker recognition systems used to recognize a user, compromising security guarantees.

This thesis explores adversarially robust speech and speaker recognition in three parts. The first part focuses on increasing the robustness of speaker recognition by augmenting it with a facial recognition. We show that a weighted recognition based on environmental conditions, such as noise and lighting, is more accurate than relying on a single modality or a multi-modal system with static weights. Additionally, we propose a method to detect spoofing attacks against audio-visual speaker recognition; to assess authenticity, we simultaneously verify the synchronicity and transcription of the captured streams.

The second part analyzes adversarial examples for hybrid speech recognition systems that exploit *psychoacoustic hearing thresholds.* We show that it is possible to calculate inconspicuous adversarial examples by leveraging psychoacoustic principles to limit the audibility of adversarial perturbations. The attack is further extended to be viable when the adversarial example is played via a loudspeaker. For this purpose, we also consider varying room characteristics for the optimization of adversarial examples. The resulting adversarial examples remain viable across different rooms and recording setups. In addition to demonstrating this attack, we developed a detection mechanism for adversarial examples. We use a one-class classifier trained on uncertainty measures to detect potential adversarial examples as outliers.

In the final chapter, we perform a systematic analysis of the sensitivity of popular smart speakers to *accidental triggers*. We investigate the prevalence of accidental triggers by measuring triggers for a diverse set of 11 smart speakers. Finally, we propose an approach to artificially craft accidental triggers based on a Levenshtein distance that can be used to benchmark the robustness of smart speakers.

Overall, this thesis shows real threats against hands-free audio interfaces that need to be considered. In addition, we have developed methods to quantify weaknesses of speech assistants and countermeasures to improve the robustness of speech and speaker recognition.

Kurzfassung

Sprach- und Sprechererkennungssysteme sind in unseren Alltag integriert; Sprachassistenten beantworten Fragen, stellen Wecker oder spielen Musik ab, verschicken aber auch persönliche Nachrichten, steuern Smart Homes und geben Online-Bestellungen auf. Sie erfassen und analysieren ständig ihre Umgebung, was sie zu einem Einfallstor für potenzielle Angriffe macht. Viele Sicherheits- und Datenschutzbedenken ergeben sich aus ihrer eingebauten automatischen Spracherkennung, einschließlich der damit einhergehenden Risiken von *Adversarial Examples* und *Spoofing-Angriffen.* Adversarial Examples, die für Menschen wie harmlose Audiosignale klingen, werden vom automatischen Spracherkennungssystem als ein vom Angreifer gewählter, potentiell bösartiger Sprachbefehl interpretiert. Spoofing-Angriffe, die die Stimme eines Opfers imitieren, können Sprechererkennungssysteme, die zur Erkennung eines Benutzers eingesetzt werden, täuschen und so Sicherheitsgarantien beeinträchtigen.

In dieser Arbeit wird die Robustheit der Sprach- und Sprechererkennung in drei Teilen untersucht. Der erste Teil konzentriert sich darauf, die Robustheit von Sprechererkennung zu erhöhen, indem sie durch Gesichtserkennung ergänzt wird. Wir zeigen, dass eine gewichtete Erkennung auf der Basis von Umgebungsbedingungen, wie z. B. Lärm und Beleuchtung, genauer ist, als sich auf eine einzelne Modalität oder ein bimodales Erkennungssystem mit konstanter Gewichtung zu verlassen. Zusätzlich schlagen wir eine Methode zur Erkennung von Spoofing-Angriffen gegen eine audiovisuelle Sprechererkennung vor; um die Authentizität zu beurteilen, überprüfen wir gleichzeitig die Synchronität und Transkription der erfassten Streams.

Im zweiten Teil analysieren wir Adversarial Examples für hybride Spracherkennungssysteme, die *psychoakustische Hörschwellen* ausnutzen. Wir zeigen, dass es möglich ist, unauffällige Adversarial Examples zu berechnen, indem psychoakustische Prinzipien genutzt werden, um die hörbaren Signalveränderungen zu minimieren. Der Angriff wird so erweitert, dass er auch durchführbar ist, wenn die Adversarial Examples über Lautsprecher abgespielt werden. Hierfür berücksichtigen wir unterschiedliche Raumeigenschaften bei der Optimierung von Adversarial Examples. Die resultierenden Adversarial Examples bleiben über verschiedene Räume und Aufnahmekonfigurationen hinweg robust. Neben der Demonstration dieses Angriffs haben wir auch einen Mechanismus zur Erkennung von Adversarial Examples entwickelt. Wir verwenden einen Ein-Klassen-Klassifikator, der auf Unsicherheitsmaße trainiert wurde, um potenzielle Adversarial Examples als Ausreißer zu erkennen.

Im letzten Kapitel führen wir eine systematische Analyse der Empfindlichkeit beliebter Smart Speaker gegenüber versehentlichen Aktivierungen durch. Wir untersuchen die Häufigkeit von versehentlichen Aktivierungen, indem wir deren Anzahl bei 11 Smart Speakern messen. Schließlich schlagen wir einen Ansatz zur künstlichen Herstellung von solchen Aktivierungen vor, der zum Benchmarking der Robustheit von Smart Speakern verwendet werden kann.

Insgesamt zeigt diese Arbeit reale Bedrohungen gegen Sprach- und Sprechererkennungssysteme auf, die es zu berücksichtigen gilt. Darüber hinaus haben wir Methoden zur Quantifizierung dieser Schwächen von Sprachassistenten und Maßnahmen zur Verbesserung der Robustheit von Sprach- und Sprechererkennung entwickelt.

Acknowledgement

First, I would like to thank my advisor Dorothea for her continuous support and valuable feedback. I really enjoyed being part of the *Cognitive Signal Processing Group*.

Moreover, I would like to thank Thorsten, who always welcomed me in his group and provided me with advice and constructive discussions.

Many thanks to my colleagues and friends who have supported me through countless ups and downs. Without you, this thesis would not have been possible: Thorsten, Sina, Joel, Nils, Moe, Merlin, Alexandru, Katharina, David, Kai, Lena, Jan, Hendrik, Doreen, Dennis, Jan, and many, many more.

I also want to thank my family: Jürgen, Christine, Sylvia, and Lorenz, for their endless support and for never asking questions like "When will you be finished?".

Finally, I would like to thank Max for being a part of this journey. I am incredibly lucky to have you by my side all the time.

Contents

List of Figures

List of Tables

Abbreviations

aKLD	*averaged Kullback-Leibler Divergence*
ASR	*Automatic Speech Recognition*
AUROC	*Area Under the Receiving Operator Curve*
BNN	*Bayesian Neural Network*
CCA	*Canonical Correlation Analysis*
CHMM	*Coupled Hidden Markov Model*
CHiME	*Computational Hearing in Multisource Environment*
CNN	*Convolutional Neural Network*
CoIA	*Co-Inertia Analysis*
CQT	*Constant Q Transform*
CTC	*Connectionist Temporal Classification*
DCT	*Discrete Cosine Transformation*
DFT	*Discrete Fourier Transform*
DNN	*Deep Neural Network*
DNN-HMM	*Deep Neural Network Hidden Markov Model*
DTW	*Dynamic Time Warping*
EER	*Equal Error Rate*
ELBO	*Evidence Lower Bound*

EM	*Expectation Maximization*
FGSM	*Fast Gradient Sign Method*
fNN	*feed-forward Neural Network*
GMM	*Gaussian Mixture Model*
GMM-HMM	*Gaussian Mixture Model Hidden Markov Model*
GMM-UBM	*Gaussian-Mixture-Model-based Universal Background Model*
HMM	*Hidden Markov Model*
JFA	*Joint Factor Analysis*
KLD	*Kullback-Leibler Divergence*
LBP	*Local Binary Pattern*
LDA	*Linear Discriminant Analysis*
LSTM	*Long Short-Term Memory*
MCE	*Minimum Classification Error*
MFCC	*Mel Frequency Cepstral Coefficients*
MI	*Mutual Information*
MUSHRA	*Multiple Stimuli with Hidden Reference and Anchor*
NLP	*Natural Language Processing*
NN	*Neural Network*
PGD	*Projected Gradient Descent*
PLDA	*Probabilistic Linear Discriminant Analysis*
ReLU	*Rectified linear*
RIR	*Room Impulse Response*
RNN	*Recurrent Neural Network*
ROC	*Receiver Operating Characteristic*
SNR	*Signal-to-Noise Ratio*
SNRseg	*segmental Signal-to-Noise Ratio*
STFT	*Short-Time Fourier Transform*
TTS	*Text-To-Speech*
UBM	*Universal Background Model*
WER	*Word Error Rate*

Symbols

Functions and Operators

$\arg\max(\cdot)$	Argument of the maximum
$\arg\min(\cdot)$	Argument of the minimum
$\max(\cdot)$	Maximum
$\min(\cdot)$	Minimum
$\mathrm{corr}(\cdot)$	Correlation
$\mathcal{D}(\cdot)$	Dispersion
$\mathbb{E}(\cdot)$	Expected value
$\mathcal{F}(\cdot)$	Neural network
$\mathcal{H}(\cdot)$	Entropy
$I(\cdot)$	Mutual information
$D(\cdot)$	Kullback-Leibler divergence
$\mathcal{L}(\cdot)$	Loss/objective function
$\mathcal{N}(\cdot)$	Normal distribution
$\mathrm{P}(\cdot)$	Probability
$\mathcal{R}(\cdot)$	Empirical risk
$\mathrm{sgn}(\cdot)$	Sign of argument

$\mathrm{Re}(\cdot)$	Real part
$\mathrm{Im}(\cdot)$	Imaginary part
$\lvert\cdot\rvert$	Absolute value
$\lvert\lvert\cdot\rvert\rvert_\infty$	Infinity norm
$*$	Convolution operator

Greek symbols

α	Learning rate
Γ	Posteriors
δ	Viterbi probability for all time steps/states
ϵ	Perturbation of an adversarial example
ξ	RIR distribution
κ	Uncertainty measure
λ	Stream weights
μ	Mean
η	Margin to psychoacoustic hearing thresholds
ϕ	GMM mixture component weight
Π	HMM initial probabilities
$\boldsymbol{\Sigma}$	Covariance matrix
σ	Standard deviation
$\boldsymbol{\theta}$	Model parameters
τ	Transcription
ψ	HMM states with highest probability for all time steps
χ	STFT features
Ψ	Perceivable noise
ω	Window function

Roman symbols

A	Audio modality
$\mathcal{A}$	Audio stream
a	HMM state transition probability

$\boldsymbol{b}$	Room dimensions
b	Observation Likelihood
c	MFCC coefficient
C	Speaker likelihood
D	Deletions
d	Deletion weight
$\boldsymbol{D}$	Diagonal residual matrix
$\mathcal{D}$	Data set distribution
D	Train set
f	Frequency
f_{mel}	Mel frequency
$\boldsymbol{H}$	Hearing thresholds
h	Room impulse response
$\mathcal{H}$	RIR distribution
$\mathcal{I}$	Image
I	Insertions
i	Insertion weight
$\mathcal{L}$	Levenshtein distance
$\boldsymbol{m}$	Speaker- and session-independent super vector
$\boldsymbol{M}$	Modified spectrogram
$\boldsymbol{\mathcal{M}}$	Super vector
$\boldsymbol{o}$	Observation vector
$\boldsymbol{P}$	Spectral perturbations
P	Probability
q	HMM state
$\boldsymbol{r}$	Receiver position
R	Recognition rate
$\boldsymbol{s}$	source position
$\int$	Mel-filtered signal
$\boldsymbol{S}$	Spectrogram

S	Substitutions
s	Substitution weight
T_{60}	Reverberation time
$\boldsymbol{T}$	Total variability space
$\boldsymbol{U}$	Eigenchannelmatrix
$\boldsymbol{V}$	Eigenvoicematrix
$\mathcal{V}$	Video stream
V	Video modality
$\boldsymbol{w}$	I-vector
w	Neural network weight
$\mathcal{X}$	Set of all possible values of the distributions
x	Input signal
X	Frequency domain signal
$\boldsymbol{x}$	Feature vector
y	Data label
y'	Adversarial target label
$\hat{y}$	Model output

1 | Introduction

The popularity of hands-free audio interfaces like voice assistants has grown substantially in the last decade. They are integrated into our everyday lives, answer questions, set timers, play music, but are also used to send personal messages, control smart homes, and place orders online. To enable a frictionless user experience, these speech-based interfaces constantly capture and analyze their surrounding environment. However, the integration of voice assistants into security- and privacy-sensitive environments makes them potential targets for attacks that leverage the built-in *Automatic Speech Recognition* (ASR) system [1, 2, 3].

Smart speakers are on their way to become a pervasive technology but have several security and privacy implications due to the way these devices operate; attacks over radio or TV could affect a large number of devices. Unintended online shopping orders have already happened because of commands uttered in TV commercials, where Amazon Echos have reacted to an unintended purchase command [1]. In another case, Burger King produced a TV commercial that intentionally activated Google Home speakers [2]. The integration of voice assistants into smart homes can lead to significant vulnerabilities. In a worst-case scenario, an attacker may take over systems such as security cameras, alarm systems, or smart locks [3].

State-of-the-art ASR systems that map speech input into its text representation are based on neural networks, which are vulnerable to *adversarial examples*: maliciously changed inputs that force a machine learning model to make a wrong prediction [4].

Acoustic adversarial examples sound like inconspicuous audio for human listeners but are interpreted by the ASR system as an attacker-chosen speech command. An attacker can often bypass any countermeasure or detection mechanism once they are aware of the defense principles [5] and it is hard to design robust countermeasures or detection mechanisms that follow Kerckhoffs' principle [6], where an attacker is assumed to have full knowledge about the applied security measures.

Voice profiles are used in many modern voice assistants to recognize users and guard against misuse [7]. Additional voice training is used to recognize the user and to build context around questions, e. g., the contact list of a person is used to better understand uttered names [8]. In addition, these systems can deliver personalized results that make use of private information.

Spoofing attacks that imitate a victim's voice can fool speaker recognition systems and compromise security and privacy guarantees. This may happen via replay attacks, where an attacker plays a recording of the victim's voice, or via *Text-To-Speech* (TTS) or voice conversion, where the victim's voice is synthesized or converted from another speaker [9].

This thesis focuses on adversarially robust speech and speaker recognition in three parts. The first part investigates the robustness of speaker recognition. We augment speaker recognition with a facial recognition and show that a weighted recognition, with weights that are chosen according to environmental conditions, like noise and lighting, is more accurate than relying on a single modality or a multi-modal system with static weights. The results are presented for an i-vector-based speaker recognition system and face recognition based on *Local Binary Pattern* (LBP). We train a neural network that takes uncertainty measures of both modalities as its input and returns the optimal weight for the score-based recognition.

Adding a modality will not prevent all kinds of attacks. Therefore, in Chapter 2, additionally an audio-visual spoofing detection is presented. For this purpose, we use *Coupled Hidden Markov Models* (CHMMs) to simultaneously verify the synchronicity and transcription of an audio-visual stream. Our results show that the system successfully recognizes various attack scenarios: where the wrong text has been uttered, one modality is missing, or the audio and the video streams do not match.

Chapter 3 focuses on adversarial examples for hybrid speech recognition systems. We analyze an attack based on *psychoacoustics*, where the principles of dynamic hearing thresholds are exploited to limit the adversarial perturbations to those parts of the audio sample, where the perturbations are inconspicuous. Our experimental results show that it is possible to significantly decrease the perceptible noise. This is confirmed in two user studies with human listeners.

To investigate the practical implications and the real-world impact of audio adversarial examples, the attack is extended to situations where the audio signal is played via a loudspeaker. We consider the room characteristics as a random variable utilizing *room impulse responds* (RIRs) during the optimization of adversarial examples. RIRs describe the transmission of audio signals over the air. We show that no prior knowledge about the attack environment is required. The resulting adversarial examples remain robust if played in various rooms and do not need to be tailored to a specific environment.

In the remainder of Chapter 3, we propose a detection mechanism for audio adversarial examples. Instead of the *Deep Neural Network* (DNN) acting as the acoustic model of a hybrid ASR system, we substitute it via different neural networks capable of uncertainty quantification. Additionally, a one-class classifier is trained on different uncertainty measures to detect adversarial examples as outliers of a one-class classifier trained on benign examples. This has the advantage that the detection is not tailored to own attack.

In Chapter 4, we perform a systematic analysis of the sensitivity of popular smart speakers to so-called *accidental triggers*. Accidental triggers are triggers of a smart speaker that confuses the wake work with similar-sounding words or sentences. We conduct a study on the prevalence of accidental triggers by measuring triggers for 11 different smart speakers and wake words for different audio content and languages. Additionally, we introduce a method to craft accidental triggers with the help of a Levenshtein distance metric. We demonstrate that this method enables us to systematically find new accidental triggers and argue that this method can benchmark the robustness of smart speakers.

1.1 Acoustic Signal Processing

Acoustic signal processing is needed to, e. g., extract features that are used as input for a machine learning model like ASR and speaker recognition.

In the following, two typical audio features are presented: *Short-Time Fourier Transform* (STFT) as frequency representation of audio signals and *Mel Frequency*

Cepstral Coefficients (MFCCs) that utilizes the non-linear frequency perception of listeners. Additionally, we describe psychoacoustic hearing thresholds and the dependencies between frequencies that lead to masking effects in the human perception.

1.1.1 Acoustic Features

Acoustic features, e. g., for speech signals, should preserve all relevant information while removing redundant information and reducing the amount of irrelevant data used as input for statistical models or neural networks. Audio features are generally represented in the frequency domain, either directly, via an STFT, or with additional processing steps taken to consider human frequency perception, e. g., in MFCCs.

Short-Time Fourier Transform. To calculate the STFT, the input waveform is divided into *frames* (e. g., 20 ms) with an overlap (e. g., 10 ms) between two adjacent frames, followed by a window function to avoid spectral leakage[1]. Window functions (i. e., Hamming, Hann, Blackman window) normally are a trade-off between frequency resolution and spectral leakage and their choice depend on the task and the signal properties. The frames are transformed individually using a *Discrete Fourier Transform* (DFT) to obtain a representation in the frequency domain

$$X(k) = \sum_{n=1}^{N} x_\omega(n) e^{-i2\pi \frac{kn}{N}}, \quad k = 1, \dots, K, \tag{1.1}$$

where x_ω is the framed and windowed input signal, N the frame length in samples, and K the DFT length. For the final feature χ, the logarithm of the magnitude spectrum is calculated in the last step.

$$\chi(k) = \log(|X(k)|), \tag{1.2}$$

Mel Frequency Cepstral Coefficients. STFT features serve as a starting point to calculate MFCC features. For this purpose, a Mel filter bank is applied to the STFT features. The Mel filter bank is an ensemble of triangular filters that imitates the human non-linear frequency perception (cf. Section 1.1.2). To map frequency into the Mel scale, the following relation can be used

$$f_{\text{mel}} = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right), \tag{1.3}$$

[1]Energies from one frequency leak into other frequency bins.

where f if the frequency in Hz and f_{mel} the Mel-scaled frequency.

To compute the MFCCs, a *Discrete Cosine Transformation* (DCT) is applied to the logarithm of the Mel-filtered signal $\mathfrak{f}$

$$c(m) = \sum_{k=0}^{K-1} \log[\mathfrak{f}(k)] \cdot \cos\left(\frac{\pi \cdot m(k-0.5)}{K}\right), \quad m = 1, \ldots, M, \tag{1.4}$$

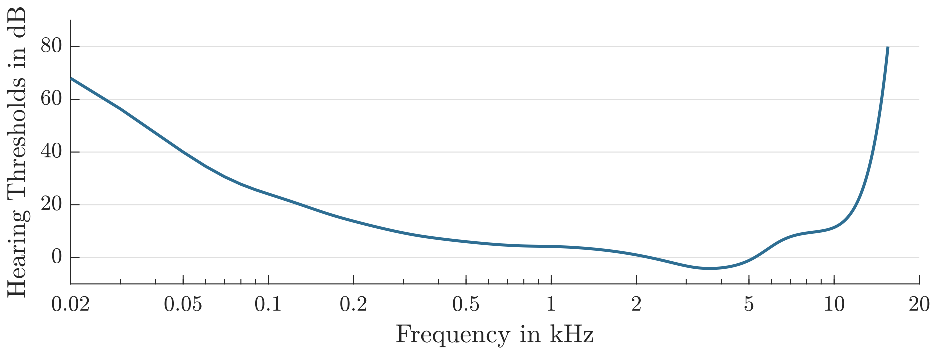
where M is the number of DCT coefficients. The features are normally augmented by their first and second derivatives to represent temporal structure.

1.1.2 Psychoacoustics

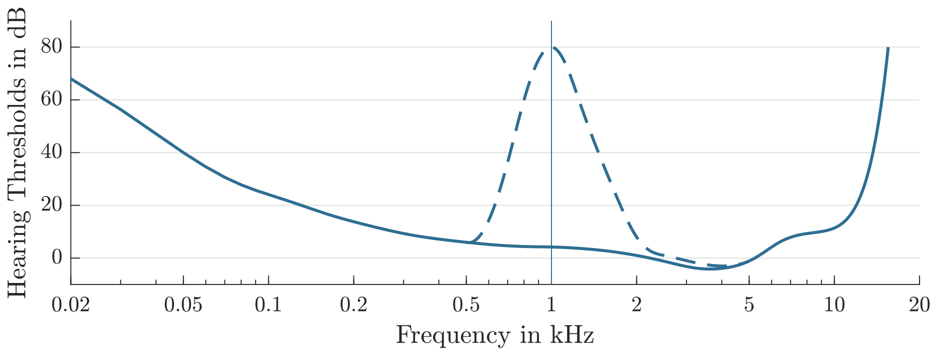
Psychoacoustics yields an effective measure of (in-)audibility and describes how the dependencies between frequencies and across time lead to masking effects in human perception [10]. Probably the best-known example for applying these effects is MP3 compression [11], where the compression algorithm uses empirical hearing thresholds to minimize bandwidth or storage requirements. Psychoacoustic masking can be understood as a combination of signal-independent and signal-dependent thresholds.

Signal-Independent Thresholds. Figure 1.1a shows the *absolute hearing thresholds*, plotted over the entire perceptible frequency range. They describe the thresholds of the human perception as a function of the frequency, namely the energy necessary for a tone of a specific frequency to be perceptible to humans. Sounds with levels below the curve are generally not perceptible for a human listener.

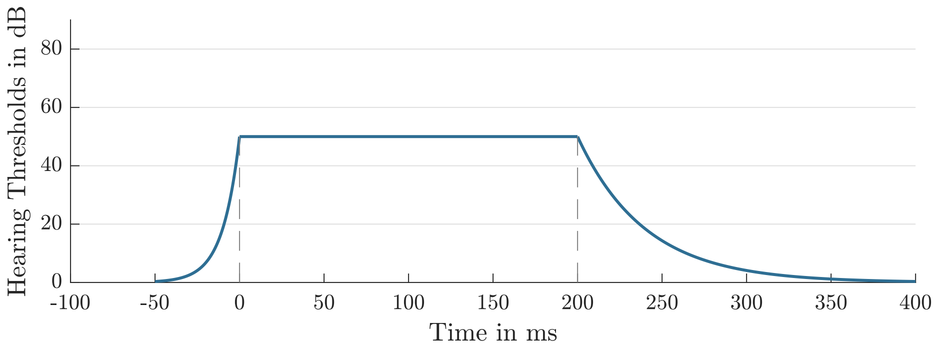
Signal-Dependent Thresholds. Signal-dependent thresholds are based on a human-perception-based representation of the audio signal, the so-called *critical bands*. Critical bands describe humans' non-linear frequency perception and are also utilized for the non-linear frequency mapping of MFCCs. Critical bands have been found in a wide range of experiments. Consistently, these reveal that lower frequencies have a higher resolution than higher frequencies [12]. Within these bands, a second tone can strongly interfere with the first tone's perception. An example of such a signal-dependent frequency masking is shown in Figure 1.1b; here, a tone at 1 kHz shifts the absolute hearing thresholds (solid line) to a higher threshold in the neighboring frequencies (dashed line).



(a) Absolute Hearing Thresholds



(b) Frequency Masking



(c) Temporal Masking

Figure 1.1: Psychoacoustic thresholds describe the limitations of the human auditory system. Figure 1.1a shows the average human hearing threshold in quiet. Figure 1.1b shows an example of masking, illustrating how a loud tone at 1kHz shifts the hearing thresholds of nearby frequencies and Figure 1.1c shows how the recovery time of the auditory system after processing a loud signal leads to temporal masking.

Another class of signal-dependent thresholds is temporal masking. Temporal masking can be explained by the fact that the auditory system needs a certain amount of time, in the range of a few hundreds of milliseconds, to recover after processing a higher-energy sound event to be able to perceive a new, less energetic sound. Interestingly, this effect does not only occur at the end of a sound but also, although much less distinct, at the beginning of a sound. This seeming causal contradiction can be explained by the processing of the sound in the human auditory system. An example of temporal masking is shown in Figure 1.1c.

The combination of all these masking effects builds the psychoacoustic hearing thresholds of an audio signal. They describe the amount of energy that can be added to the signal in each time-frequency bin without being noticeable as a change to the original audio signal for a human listener.

1.2 Statistical Modeling

Audio features are often summarized by a statistical model, e. g., *Gaussian Mixture Models* (GMMs). Even if state-of-the-art DNNs often replace these models, they remain the foundation of many speech applications, including speaker and speech recognition.

This section formally describes GMMs and their training via the *Expectation Maximization* (EM) algorithm. Additionally, it introduces *Hidden Markov Models* (HMMs), a core component of state-of-the-art hybrid ASR systems, and the Viterbi algorithm used to decode HMMs.

1.2.1 Gaussian Mixture Models

GMMs are probabilistic mixture models that can approximate generic distributions as a linear combination of weighted Gaussian distributions. Their training is unsupervised and do therefore require no information about the label of the underlying training data. A multivariate GMM with K mixture components for input data $\boldsymbol{x} \in \mathbb{R}^{D_x}$ is described via

$$p(\boldsymbol{x}) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{1.5}$$

with $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ defining a multivariate normal distribution with mean vector $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$ of mixture component i

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{\sqrt{(2\pi)^K |\boldsymbol{\Sigma}_i|}} \exp\Big(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)\Big). \tag{1.6}$$

The weights of the mixture components are constrained such that

$$\sum_{i=1}^{K} \phi_i = 1. \tag{1.7}$$

Multivariate GMMs with a defined number of components K are trained via the EM algorithm, which approximates the mixture model iteratively by applying two steps per iteration; (i) the *expectation* step that calculates the expectation of the mixture components given the input data $\boldsymbol{x}$ and the model parameters ϕ_i, $\boldsymbol{\mu}_i$, and $\boldsymbol{\Sigma}_i$ for $i = 1 \ldots K$ and (ii) the *maximization* step which maximizes the calculated expectation with respect to the model parameters. The two steps are repeated until it converges or a maximum number of iterations is reached.

1.2.2 Hidden Markov Models

HMMs are used to estimate the probability of state sequences as a statistical process. In contrast to Markov chains, the states of HMMs can not be observed directly. Instead, only observations that depend on the states are available. Additionally, following the first-order Markov property, HMMs assume succeeding states to depend only on previous states. For hybrid ASR systems, in general, first-order Markov models are used such that the current state only depends on one predecessor state.

HMMs can be described as a set of states $Q = \{q_i\}_{i=1}^N$ with N states. Additionally, for each state, conditional observation likelihoods

$$b_i(\boldsymbol{o}(t)) = \mathrm{P}(\boldsymbol{o}(t)|q_i), \tag{1.8}$$

need to be modeled, where $\boldsymbol{o}$ are the observed features.

State transition probabilities that describe the probability of moving from state $q(t) = i$ to state $q(t+1) = j$ in a discrete time step $t \rightarrow t+1$ are defined as

$$a_{ij} = \mathrm{P}\big(q(t+1) = j | q(t) = i\big). \tag{1.9}$$

For the first state of a sequence, initial state probabilities $\boldsymbol{\Pi} = [\pi_1, \ldots, \pi_N]$ are

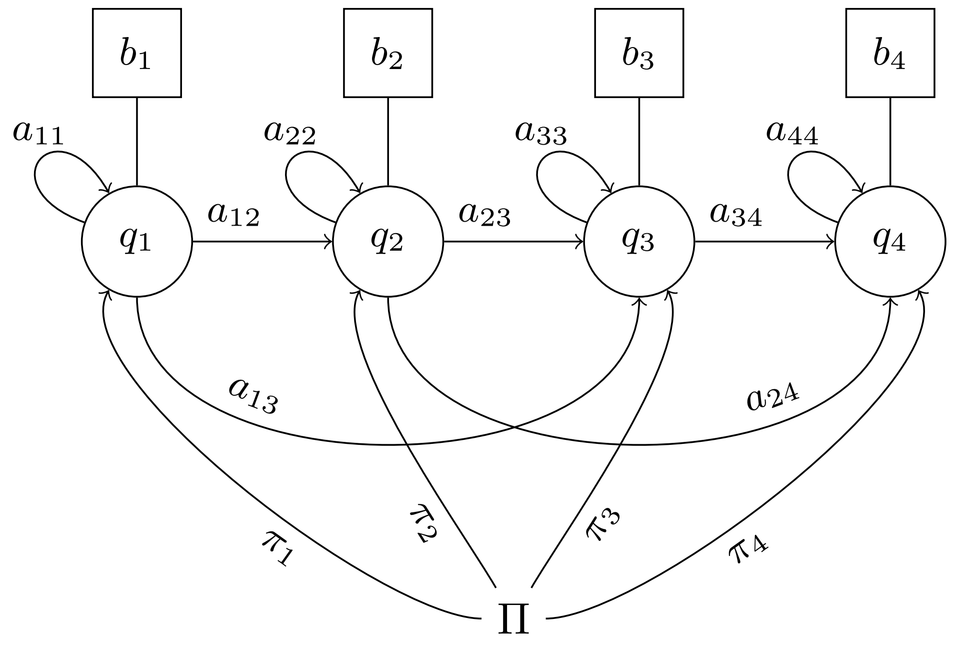


Figure 1.2: HMM with 4 states and its conditional observation likelihoods, transition probabilities, and initial state probabilities

required, describing the probabilities to start in any state. An example of an HMM with 4 states and its conditional observation likelihoods, transition probabilities, and initial state probabilities is shown in Figure 1.2.

Viterbi Algorithm. A naive way to find the state sequence with the highest probability given an HMM and an observation sequence is to test all possible state sequences. This is clearly infeasible, as the number of possible state sequences grows exponentially with an increasing length of the sequence T.

The Viterbi algorithm provides a far more efficient way to find the best state sequence of an HMM given an observation sequence. For this purpose, it utilizes dynamic programming techniques to conduct a graph search and to detect the path—or state sequence, in case of an HMM—with the lowest cost.

The Viterbi algorithm maximizes the the probability $\delta(i,T)$ w.r.t i along a path $q(t)$ for $t = 1, \ldots, T$ time steps. It is a four-step algorithm; the first two steps iterate through the observations $\boldsymbol{o}(t)$ keeping track of the best predecessor states $\psi(i,t)$ that maximize the probability $\delta(i,T)$. The last two steps return the state sequence with the highest probability:

1. Initialization:

$$\delta(i,1) = \Pi_i b_i\big(\boldsymbol{o}(1)\big), \qquad \forall\, i = 1 \dots N, \tag{1.10}$$

$$\psi(i,1) = 0, \qquad \forall\, i = 1 \dots N. \tag{1.11}$$

2. Recursion for all $t = 2 \dots T-1$:

$$\delta(j,t) = \max_{i=1\dots N} \big[\delta(i,t-1)a_{ij}\big] b_j\big(\boldsymbol{o}(t)\big), \quad \forall\, j = 1 \dots N, \tag{1.12}$$

$$\psi(j,t) = \operatorname*{arg\,max}_{i=1\dots N} \delta(i,t-1)a_{ij}, \quad \forall\, j = 1 \dots N. \tag{1.13}$$

3. Termination:

$$P^* = \max_{i=1\dots N} \delta(i,T), \tag{1.14}$$

$$q(T) = \operatorname*{arg\,max}_{i=1\dots N} \delta(i,T). \tag{1.15}$$

4. Backtracking:

$$q(t) = \psi\big(q(t+1), t+1\big), \quad \forall\, t = T-1, \dots 1. \tag{1.16}$$

The Viterbi algorithm can be extended with a beam search, where only the best paths are stored during the recursion step to increase efficiency. This variation of the algorithm does not guarantee to find the path with the highest probability but makes the calculation feasible for large HMMs.

1.3 Deep Learning

DNNs have evolved to state-of-the-art models for many applications. Their ability to learn and represent real-world data makes them a favorable choice for many tasks that are hard to model with statistical approaches, such as image classification, *Natural Language Processing* (NLP), and speech-based recognition systems.

The following section introduces neural networks, including their gradient-based optimization and the calculation of adversarial examples, a major weakness of deep-learning-based systems.

1.3.1 Deep Neural Networks

Deep neural networks are biologically inspired machine learning models for representation learning. Depending on the task, neural networks can, for example, be utilized for supervised learning, e. g., in classification and regression, or unsupervised learning like clustering.

For this purpose, so-called neurons are arranged in layers that are stacked and connected via weighted edges to form a neural network. The neural network's architecture and parameters $\boldsymbol{\theta} = \{w_{ij}^{(k)}\}$ describe a function

$$\hat{\boldsymbol{y}} = \mathcal{F}_{\boldsymbol{\theta}}(\boldsymbol{x}), \tag{1.17}$$

that maps input data $\boldsymbol{x}$ to an output $\hat{\boldsymbol{y}}$.

For each layer $k = 1, \dots, K$, a set of weights and biases is defined. These are optimized during the training phase. A weight $w_{ij}^{(k)}$ describes the connection strength between neuron i of the predecessor layer and neuron j of the current layer k. The biases $w_{0j}^{(k)}$ are offsets for each neuron $j = 1, \dots, J$ of layer k.

This type of a neural network is called *fully-connected neural network*. An example of such a network with three layers is shown in Figure 1.3. Neural networks with more than three layers are referred to as DNN.

For the mapping from the input vector $\boldsymbol{x}$ to the output $\hat{\boldsymbol{y}}$, *activation functions* $h^{(k)}(\cdot)$ are used to combine all outputs from the predecessor layer. The activation functions are non-linear functions, which enables the neural network to learn complex relations. The mapping for neuron j in layer k can therefore be described via

$$h^{(k)}\Big(w_{0j}^{(k)} + \sum_{i=1}^{N} w_{ij}^{(k)} \cdot \hat{x}_i\Big), \tag{1.18}$$

where $\hat{x}_i$ describes the output of neuron i of the predecessor layer. For the training of the neural network, activation functions need to be differentiable. Activation functions are a critical parameter that significantly impacts a model's accuracy and depends on the network's type and task. The *Rectified linear* (ReLU) activation function, for example, is a simple and effective choice that overcomes problems like *vanishing gradients* of the sigmoid activation function or the hyperbolic tangent activation function. The output layer of a neural network typically uses different functions, e. g., the softmax function for classification tasks.

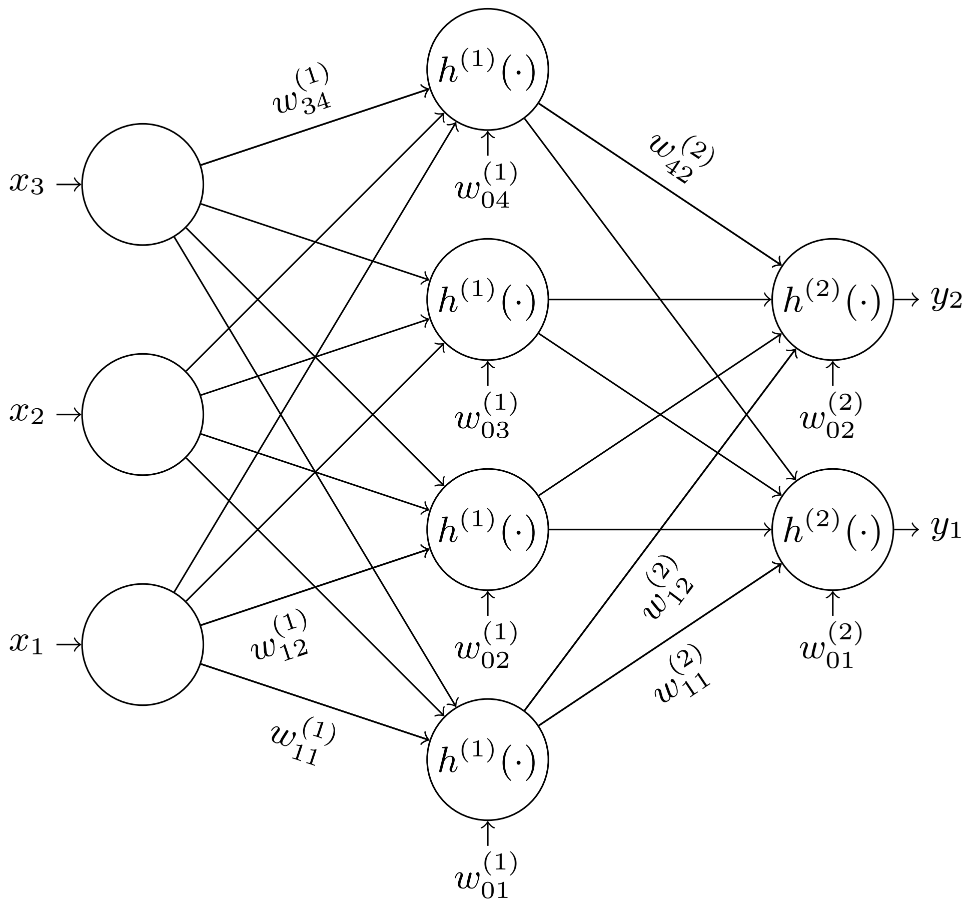


Figure 1.3: Fully-connected neural network with three layers.

Depending on the problem, different and more complex variants of neural networks are deployed, such as *Convolutional Neural Networks* (CNNs) for image data or *Long Short-Term Memory* (LSTM) networks [13]. Attention networks [14] like transformers [15] are useful, if a temporal representation or context should be learned, e. g., like for ASR or translation tasks.

1.3.2 Optimization

Neural networks describe a non-linear and non-convex problem space. A formal description of the training of neural network is the minimization of the model's *empirical risk* $\mathcal{R}$ w.r.t to the model parameters $\boldsymbol{\theta}$

$$\min_{\boldsymbol{\theta}} \mathcal{R}(\mathcal{F}_{\boldsymbol{\theta}}, D), \tag{1.19}$$

where the labeled training data $D = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}) \sim \mathcal{D}\}_{i=1}^{M}$ with M training samples and input data $\boldsymbol{x}^{(i)} \in \mathbb{R}^{D_x}$ and label $\boldsymbol{y}^{(i)} \in \mathbb{R}^{D_y}$ approximates the true distribution $\mathcal{D}$ of the data [16]. In practice, the empirical risk is described via

$$\mathcal{R}(\mathcal{F}_{\boldsymbol{\theta}}, D) = \frac{1}{M} \sum_{i=1}^{M} \mathcal{L}(\hat{\boldsymbol{y}}^{(i)}, \boldsymbol{y}^{(i)}), \tag{1.20}$$

where $\mathcal{L}(\hat{\boldsymbol{y}}^{(i)}, \boldsymbol{y}^{(i)})$ describes the *objective function* to measure the difference between the output of the neural network $\hat{\boldsymbol{y}}$ and the actual label $\boldsymbol{y}$.

To minimize the empirical risk, in general, gradient-based methods like *backpropagation* are typically used. Backpropagation is an optimization algorithm for computational graphs like those of neural networks that are too complex to be optimized analytically. For this purpose, the gradients $\nabla_{\boldsymbol{\theta}}$ of parameters $\boldsymbol{\theta}$ are calculated using the objective function $\mathcal{L}(\hat{\boldsymbol{y}}^{(i)}, \boldsymbol{y}^{(i)})$

$$\nabla_{\boldsymbol{\theta}} = \frac{\partial \mathcal{L}(\hat{\boldsymbol{y}}^{(i)}, \boldsymbol{y}^{(i)})}{\partial \boldsymbol{\theta}}. \tag{1.21}$$

The objective function therefore needs to be differentiable. Depending on the tasks different objective functions are used, e. g., mean square error for regression problems or the cross-entropy for classification tasks. The parameters are updated iteratively

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \cdot \nabla_{\boldsymbol{\theta}}, \tag{1.22}$$

with a learning rate α.

With *stochastic gradient descent*, the gradient can be approximated via the average of a batch of training samples to accelerate the training process. Hence, stochastic gradient descent is not as precise as gradient descent but much faster than the classical version.

Additionally, backpropagation based on gradient descent is guaranteed to find a minimum, but not necessarily the global minimum. Variations like the *Adam* optimizer further improve the parameter optimization [17] by using different learning rates for each parameter and additionally adapting these learning rates, utilizing the second moment of the objective function.

1.3.3 Adversarial Examples

Adversarial examples are an attack against machine learning models where an input sample is changed in such a way that the model is forced to make a wrong prediction.

In general, adversarial examples can be calculated for any machine learning system [18, 19, 20], but they are particularly successful for neural networks [21, 22].

The efficiency of adversarial examples in neural networks is due to the high number of parameters, which leads to a very complex function $\mathcal{F}(\boldsymbol{x})$. Additionally, the training data set is only an approximation of the distribution of real data and neural networks do therefore not capable to model real world data exactly.

The insufficient generalization of $\mathcal{F}(\boldsymbol{x})$ can lead to blind spots, which are usually not obvious to humans. Adversarial examples exploit this weakness by using a manipulated input $\boldsymbol{x}'$ that closely resembles the original input $\boldsymbol{x}$, but leads to a different mapping:

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\delta}, \quad \text{such that } \mathcal{F}(\boldsymbol{x}) \neq \mathcal{F}(\boldsymbol{x}'). \tag{1.23}$$

The added distortions δ can also be restricted, whereby the restriction depends on the data. In Chapter 3, we show an approach where psychoacoustic hearing thresholds are used to minimize the perceptible distortions of audio adversarial examples.

White-Box vs. Black-Box Attack. In a white-box attack, the attacker knows the model structure and all model parameters. This information can be used to specifically design perturbations that are added to the original input. A black-box attack does not have information about the system it attacks. Therefore, these kinds of attacks are more challenging but not impossible. It has been shown that gradient masking—obfuscating the gradient to avoid optimization-based attacks—is no effective countermeasure against adversarial examples [23].

Untargeted Attacks. Untargeted attacks, also called evasion attacks, aim to force the system to output a wrong prediction. For this purpose, the attacker tries to maximize the loss between the model output and the true label

$$\boldsymbol{x}' = \arg\max_{\boldsymbol{\delta}} \mathcal{L}\Big(\mathcal{F}(\boldsymbol{x} + \boldsymbol{\delta}), \boldsymbol{y}\Big). \tag{1.24}$$

Targeted Attack. In the more challenging version of targeted attacks, the malicious input $\boldsymbol{x}'$ should be predicted as an attacker chosen target label $\boldsymbol{y}'$

$$\boldsymbol{x}' = \arg\min_{\boldsymbol{\delta}} \mathcal{L}\Big(\mathcal{F}(\boldsymbol{x} + \boldsymbol{\delta}), \boldsymbol{y}'\Big). \tag{1.25}$$

Both versions, the untargeted and the targeted attack, can be calculated via gradient descent. Note that in case of an untargeted attack it is actually *gradient ascent*, because the objective function is maximized. However, it is still common to refer to it as gradient descent. Similar to the optimization-based training presented in Section 1.3.2, the input is updated with a learning rate but instead of updating the model parameters, the gradient $\nabla_{\boldsymbol{x}}$ is used to modify the input signal.

In practice, different approaches for the calculation of adversarial examples have been shown to be successful. For example, with the *Fast Gradient Sign Method* (FGSM), the gradient is not used directly but preprocessed via $\operatorname{sgn}(\nabla_{\boldsymbol{x}})$ such that only the sign of the gradients are relevant. In combination with a large learning rate and if the distortions are restricted to $||\boldsymbol{\delta}||_\infty < \epsilon$, one step is often enough to successfully craft adversarial examples.

An iterative version of crafting adversarial examples is *Projected Gradient Descent* (PGD). In contrast to FGSM, a smaller learning rate is used, and the optimization runs iteratively. This approach requires more computation time, but the added perturbations are therefore less distinct.

1.4 Speech Applications

Various systems utilize speech as input for different kinds of tasks. Speaker recognition is used for biometric authentication, and ASR is built into hands-free interfaces like smart speakers to answer questions or to control smart homes.

In the following, we present a broad overview of speaker recognition and ASR. This includes a description of standard terms used in the context of speaker recognition and the calculation of i-vector-based speaker recognition and its underlying principles. Also, existing approaches for end-to-end ASR are outlined, and the details of hybrid speech recognition systems based on DNNs, are described.

1.4.1 Speaker Recognition

Speaker recognition is a method for a voice-based biometric identification and verification method. For this purpose, a voice profile representing a person's voice characteristics is extracted from an utterance. The voice profile reflects biological properties (e. g., the length of the vocal tract) and learned pronunciation (e. g., language-dependent characteristics) that are unique for that person and can be used to verify or identify a person, either text-dependently or text-independently.

Identification. In general, identification is a 1-to-N matching, where the identity of a person is matched with a set of possible identities, e. g., all persons enrolled in a database. In this manner, the question “Who is this person?” is answered. Identification is, for example, applied in use-cases such as personalization, where an application is adapted to the preferred settings of the user or if personal information of the current speaker is considered for a request.

Verification. In contrast to identification, verification matches 1-to-1. The system already has information about a person’s potential identity and verifies whether the person under consideration is the person they claim to be. In this way, the question “Is this person who they say they are?” is answered. Verification is generally used in use cases such as authentication and maps a voice sample with a voice profile stored during the speaker’s enrollment.

Text-dependent. Text-dependent recognition requires the speaker to utter a specific password or passphrase. It has the advantage that it needs less training data and is easier to train.

Text-independent. Text-independent speaker recognition works for any spoken content of a speaker and is not restricted to, e. g., a password or passphrase. The recognition is, therefore, more flexible, but the model requires more training data. In this work, we focus on text-independent speaker recognition.

Universal Background Model

State-of-the-art speaker recognition is based on *Universal Background Model*s (UBMs). A UBM represents a speaker-independent model, which is trained by fitting a GMM on the training set’s feature representation, e. g., MFCCs. For this purpose, the training data should contain utterances representing the deployment domain of the recognizer in as many respects as possible, e. g., importantly the language [24].

In a second step, the UBM is used in a *Joint Factor Analysis* (JFA). This key component models inter-speaker variability and removes other components, e. g., channel variability [25]. For this purpose, a speaker’s utterance is represented as a supervector $\boldsymbol{\mathcal{M}}$ defined as

$$\boldsymbol{\mathcal{M}} = \boldsymbol{m} + \boldsymbol{V}\boldsymbol{y} + \boldsymbol{U}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{z}, \tag{1.26}$$

where $\boldsymbol{m}$ is a speaker- and session-independent supervector, retrieved from the UBM. The matrices $\boldsymbol{V}$ and $\boldsymbol{D}$ describe the speaker subspace, namely the eigenvoice matrix and the diagonal residual matrix. The eigenchannel matrix $\boldsymbol{U}$ defines the session subspace and is, therefore, speaker-independent. The vectors $\boldsymbol{y}$, $\boldsymbol{x}$, and $\boldsymbol{z}$ are factors for the respective matrices and assumed to be a random variable that can be described with a normal distribution $\mathcal{N}(0, \boldsymbol{I})$.

I-Vectors

A robust representation for text-independent speaker recognition are *i-vectors*. Dehak et al. [25] has first presented the principles of i-vectors, and, in contrast to the classical JFA, i-vectors describe speaker- and channel-dependent components in a single space. This space is referred to as the *total variability space* $\boldsymbol{T}$

$$\boldsymbol{\mathcal{M}} = \boldsymbol{m} + \boldsymbol{T}\boldsymbol{w}, \tag{1.27}$$

where $\boldsymbol{w}$ is defined as the i-vector that is used as input for speaker recognition. These are extracted via Baum-Welch statistics and from the UBM [25].

In recent years, neural-network-based versions of i-vectors have also been proposed [26]. These so-called *x-vectors* replace the JFA with a DNN and directly map the feature representation of an utterance into a voice profile.

1.4.2 Automatic Speech Recognition

ASR transcribes spoken content into its text representation, and with the rise of neural networks, it has become widely used for hands-free interfaces.

In general, state-of-the-art ASR systems can be divided into two types: End-to-end systems and hybrid systems. While end-to-end systems have the advantage that they are easier to construct, hybrid systems do still achieve better performance, especially in noisy environments [27].

End-to-End ASR Systems

End-to-end ASR systems directly map raw audio into its text representation and are less complex than hybrid systems [28] but need longer to converge during their training.

CTC-based end-to-end systems rely on the *Connectionist Temporal Classification* (CTC) loss for a sequence-to-sequence classification. In contrast to classical one-to-one mappings, it can be applied for mappings where the input and the output sequences are not aligned [29]. The CTC loss assumes the outputs of a sequence to be independent of each other. Consequently, the output of one time step does not affect the output of other times steps. ASR systems using CTC loss are therefore not capable of modeling language directly. To solve that problem, an external language model, e. g., n-grams, is used. For the practical implementation, a language model score is added to the calculation of CTC-based ASR models [30].

In contrast to CTC-based systems, Recurrent Neural Network *(RNN)-transducer systems* [31] also include a *prediction network* that acts as a language model. Additionally, these systems have an acoustic model (*encoder network*) and a *joint network* to combine both components. The three models are generally trained jointly.

Attention-based End-to-End Systems utilize attention mechanisms [14] to map the audio input into a transcription. The attention mechanism is borrowed from language translation tasks and has the advantage that it can emphasize the segments of a sequence that are more important for the recognition. Recent versions also use transformer networks, which are easier to train, since they do not require the recurrent structure of classical attention networks [32].

Hybrid ASR Systems

In contrast to end-to-end systems, HMM-based hybrid ASR systems use acoustic and language models that are optimized separately. Hybrid ASR systems show more robust performances in noisy environments [33, 34, 35] and require less data but are harder to construct. The formerly state-of-the-art GMM-based acoustic model has been replaced by a DNN, resulting in so-called DNN-HMM ASR systems.

Figure 1.4 shows a high-level overview of a DNN-HMM system. The DNN-HMM-based ASR system can be divided into three parts: the feature extraction, which transforms the raw input data into representative features, a DNN-based acoustic model of the system, and the decoding step, which returns the recognized transcription.

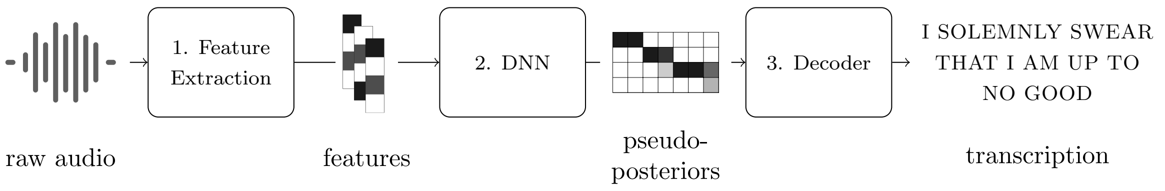


Figure 1.4: Overview of a state-of-the-art ASR system with the three main components: (1) feature extraction from the raw audio data, (2) calculating pseudo-posteriors with a DNN, and (3) the decoding stage, which returns the transcription.

Feature Extraction. The feature extraction transforms the raw input data into features that should ideally preserve all relevant information (e. g., phonetic class information, formant structure) while discarding the unnecessary remainder (e. g., properties of the room impulse response, residual noise, or voice properties like pitch information). Common acoustic feature representations are STFT or MFCC (cf. Section 1.1.1).

Acoustic Model DNN. Hybrid systems use an HMM representation in the decoding stage and utilize the DNN to estimate all HMM state probabilities (modeling context-dependent phonetic units) given the acoustic input signal. The extracted features are used as the input for the acoustic model DNN. Based on these, the DNN calculates a matrix of so-called pseudo-posteriors, which describe the probabilities for each of the language's phones being present in each frame of the feature representation. The pseudo-posteriors, which are used during the decoding step to find the most likely word sequence.

Decoding. The decoder of an ASR system utilizes some form of graph search to find the most probable word sequence given the posterior probabilities and the transition structure, e. g., represented in an HMM or *Weighted Finite-State Transducers*. This graph search can be perform in a static decoding graph e. g., constructed as a composition of individual transducers (i. e., graphs with input/output symbol mappings attached to the edges). These individual transducers describe, for example, the grammar, the lexicon, context-dependent phonetic units, and the transition and output probability functions of these phonetic units. The transducers and the pseudo-posteriors (i. e., the output of the DNN) are then used to find an optimal path through the HMM via Viterbi decoding or any other dynamic programming approach that is suited to the typically very large problem sizes.

2 | Audio-Visual Speaker Identification

Speaker recognition systems are used to build voice profiles to recognize a user based on their voice. It is used for voice assistants to build context around questions and also for telephone and mobile banking to access bank accounts [36]. Here it is used to replace passwords via a speech-based biometric authentication. For this purpose, the customer's voice profile that needs to be enrolled is compared with an incoming authentication request.

The performance of biometric systems using only one modality is poor in many real-world situations. In case of speaker and face recognition, both modalities, audio and images, can be affected by different kinds of noise: During the visual recognition process, bad lighting conditions, blurring, and rotations of the head can decrease the probability of a successful recognition. In contrast, an acoustic speaker recognition system is not influenced by these specific problems. However, the latter can be compromised by background noise.

Additionally, audio-only speaker recognition, like other biometric methods, is vulnerable to spoofing attacks, where the attacker claims the identity of another person [37]. Spoofing attacks aim to fake input data, and in the case of speaker recognition, different types exist [9]. In a replay attack, a recording of the victim's voice is used. This can also be specific utterances for text-specific speaker recognition, created by concatenating samples of the target speaker. Speech synthesis, or TTS, creates artificial samples, and in the case of voice conversion, the utterance of another speaker is converted into a target speaker's voice.

Multimodal recognition systems, like audio-visual speaker recognition, can be used to overcome these limitations by extending acoustic speaker recognition with a visual biometric method, e. g., face recognition. It would be advantageous to combine these two systems to compensate each other's weaknesses such that they can truly benefit from each other.

In the following chapter, we show thatw eighting multiple recognition results based on environmental conditions, such as noise and lighting, is more accurate than relying on a single modality or a multi-modal system with static weights. The results are presented for an i-vector-based speaker recognition system and face recognition based on LBP. We train a neural model that takes uncertainty measures of both modalities as its input and returns the optimal weight for the score-based combined recognition.

Additionally an audio-visual spoofing detection is presented, where we utilize CHMMs to simultaneously verify the synchronicity and transcription of an audio-visual stream. The results show that the system successfully recognizes different attack scenarios, where either the wrong text has been uttered, one modality is missing completely, or the audio and the video streams do not match.

2.1 Robust Audio-Visual Speaker Identification

In general, the fusion of multiple classifiers can be characterized by the stage where the fusion is applied. An early fusion is applied at feature level. Alternatively, the fusion can be conducted at the decision level, where the features are calculated separately and combined for the final decision. A late fusion, which we use here, is on the score level. For this purpose, for each modality, a score is calculated and a final decision is obtained by considering the combined scores. Atrey et al. [38] described an overview of fusion strategies for combining multiple modalities. In our work we will consider the basic product rule. The rule has been considered by other works [39], and it will serve as our baseline.

Previous works on audio-visual speech recognition [40, 41] have shown that noise-adaptive stream weights can significantly improve the performance of multimodal speech recognition. The idea is to create a side-channel for each modality, which outputs a measure of confidence. This helps to decide how much each modality should contribute to the overall decision.

In contrast to Abdelaziz et al. [40], we use the stream weight estimation for an identification task. Thus, we propose a new approach using a discriminative cost function to calculate target stream weights, which increase the score of the genuine speaker relative to the most competitive speaker. This approach will be described in Section 2.1.2 in more detail.

Similar to the *Minimum Classification Error* (MCE), used for ASR [42], our approach tries to minimize the classification error. While the MCE criterion in the context of ASR is used to reduce the number of classification errors of sequences, specifically of word sequences, our cost function maximizes the score of the genuine speaker relative to the next best candidate, which leads to a more robust speaker recognition.

Further, we introduce confidence measures, which can be used together with the target stream weights in order to learn a mapping function for an optimal weight estimation. We use a commonly used GMM-based speaker recognition model [43, 44]. More specifically, the i-vector approach it utilized, which is designed to decouple channel-related and speaker-based signal variabilities using JFA and offers greater speaker identification robustness [25].

For a state-of-the-art face recognition, we chose LBP, due to its ability to distinguish faces effectively [45]. Additionally, LBP can be implemented with low computational cost and shows high accuracy for changing lighting conditions [46].

2.1.1 Speaker and Face Recognition

In order to train and test the combination of the acoustic and visual recognition on the score level, the scores for both modalities need to be calculated separately.

Face Recognition

For face recognition, we chose LBP, which is essentially based on comparing the value of a pixel with the values of the surrounding pixels. These adjacent pixels are translated into a binary pattern. If the pixel value is smaller, the corresponding position in the binary pattern is set to 0; if the value is larger, the position is set to 1. The resulting binary pattern can be interpreted as an integer value, which is saved for each pixel and is used for the classification. To optimize the recognition, a radius can be defined, describing which neighbors should be considered for each pixel. Additionally, to decrease the dimension, the image is divided into cells and, for each cell, one element for the feature vector $\boldsymbol{x}_V$ is computed.

Furthermore, depth images are considered as well. This can later act as a countermeasure against spoofing attacks, using a captured image of the victim. Additionally, and importantly for our application, the depth image is illumination-independent, which leads to a more robust recognition. Therefore, both feature vectors computed with LBP are concatenated.

After a training phase, the resulting feature vector of a new image can be used for comparison with all enrolled speakers. At this point, the Euclidean distance is computed between the test image and the mean training image of all speakers C_k with $k = 1, ..., N_S$ to obtain a confusion matrix, where N_S is the number of enrolled speakers.

In addition, for the later audio-visual fusion, a Rayleigh probability density function is fitted onto the distances of the *true positives* to obtain class posterior probabilities $\mathrm{P}(C_k|\boldsymbol{x}_V)$.

Speaker Recognition

The i-vector recognition method is more complex than the LBP and involves several steps. We used an implementation for MATLAB by Microsoft Research [47]. In the following, we give a brief outline of this approach:

1. In the training phase, MFCCs are extracted from audio files, which contain sentences of the enrolled speakers. This data is used to fit a *Gaussian-Mixture-Model-based Universal Background Model* (GMM-UBM).

2. The i-vectors for the training and test vectors are computed as described in Section 1.4.1.

3. The trained i-vectors of all classes are processed using a *Linear Discriminant Analysis* (LDA) with Fisher's criterion for further dimensionality reduction and to improve the classification in the scoring function.

4. A Gaussian *Probabilistic Linear Discriminant Analysis* (PLDA) is applied to the previously computed training i-vectors. This corresponds to learning a fac-

tor analysis model of the i-vectors, which will be used in the actual speaker identification stage.

5. For the actual identification, a log-likelihood ratio is computed, acting as feature vector $\boldsymbol{x}_A$, and used for a pairwise scoring of the test i-vectors against all enrolled i-vectors. This is done for every possible combination so that a confusion matrix is obtained as the result considering all speakers C_k.

In addition to this confusion matrix, posterior probabilities are needed for the later fusion stages. Thus, an appropriate distribution needs to be fitted onto the likelihood ratios of the *true positives* to obtain $\mathrm{P}(C_k|\boldsymbol{x}_A)$. Here, Gaussian distributions provided a good fit, and were thus learned on the training data.

2.1.2 Classifier Combination

Once the modality-dependent feature vectors $\boldsymbol{x}_A$ and $\boldsymbol{x}_V$ have been extracted, both identification systems can compute their respective scores. However, an unweighted combination of the previously introduced scores $\mathrm{P}(C_k|\boldsymbol{x}_A)$ and $\mathrm{P}(C_k|\boldsymbol{x}_V)$ is not ideal, because under certain conditions, one of the two systems might be presented with reliable data, whereas the other might only have distorted features available, e. g., due to acoustic noise or low-quality video data.

Therefore, in this work, we suggest using confidence information, which informs a fusion stage about the reliability of each of the two sub-systems. Thus, the confidence information is utilized to reach a more environmentally robust classification based on noise-dependent weighting of the two subsystems. This approach is explained in further detail in the following.

Baseline

As mentioned above, the intention is to compute the probabilities of seeing any of the possible classes, respectively speakers C_k, given the two feature vectors $\boldsymbol{x}_i$ from the two modalities $i \in \{V, A\}$, where $k = 1 \dots N_S$. Since the probability density $\mathrm{P}(\boldsymbol{x}_i)$ is unknown, but the likelihood $\mathrm{P}(\boldsymbol{x}_i|C_j)$ has been learned, we can marginalize over all classes in the denominator:

$$\mathrm{P}(C_k|\boldsymbol{x}_i) = \frac{\mathrm{P}(\boldsymbol{x}_i|C_k)\mathrm{P}(C_k)}{\mathrm{P}(\boldsymbol{x}_i)} = \frac{\mathrm{P}(\boldsymbol{x}_i|C_k)\mathrm{P}(C_k)}{\sum_{j=1}^{N_S} \mathrm{P}(\boldsymbol{x}_i|C_j)\mathrm{P}(C_j)}. \tag{2.1}$$

For each unimodal classifier, the class $C_{\hat{K}}$ is assigned to the input feature vector $\boldsymbol{x}_i$ if

$$\hat{K} = \arg\max_{k=1 \dots N_S} \mathrm{P}(C_k|\boldsymbol{x}_i). \tag{2.2}$$

Applying the same decision rule to the audio-visual fusion task leads to

$$\hat{K} = \arg\max_{k=1 \dots N_S} \mathrm{P}(C_k|\boldsymbol{x}_V, \boldsymbol{x}_A). \tag{2.3}$$

Since the probability $\mathrm{P}(C_k|\boldsymbol{x}_V, \boldsymbol{x}_A)$ in Equation (2.3) is not known, Bayes' theorem and marginalization are applied to rewrite the conditional joint probability, leading to

$$\mathrm{P}(C_k|\boldsymbol{x}_V, \boldsymbol{x}_A) = \frac{\mathrm{P}(\boldsymbol{x}_V, \boldsymbol{x}_A|C_k)\mathrm{P}(C_k)}{\sum_{j=1}^{N_S} \mathrm{P}(\boldsymbol{x}_V, \boldsymbol{x}_A|C_j)\mathrm{P}(C_j)}. \tag{2.4}$$

Assuming that the feature vectors $\boldsymbol{x}_V, \boldsymbol{x}_A$ are statistically independent given the class C_k, the joint distribution can be factorized into

$$\mathrm{P}(\boldsymbol{x}_V, \boldsymbol{x}_A|C_k) = \mathrm{P}(\boldsymbol{x}_V|C_k)\mathrm{P}(\boldsymbol{x}_A|C_k). \tag{2.5}$$

Substituting from Equation (2.5) into Equation (2.4) and cancelling the priors by considering all speakers as equally likely, the decision rule can be rewritten as

$$\hat{K} = \arg \max_{k=1 \dots N_S} \mathrm{P}(C_k|\boldsymbol{x}_V)\mathrm{P}(C_k|\boldsymbol{x}_A). \tag{2.6}$$

Weighting of Classifiers

A stream weight λ is defined and incorporated in the following decision rule:

$$\hat{K} = \arg \max_{k=1 \dots N_S} \mathrm{P}(C_k|\boldsymbol{x}_V)^{(1-\lambda)}\mathrm{P}(C_k|\boldsymbol{x}_A)^{\lambda}, \tag{2.7}$$

such that $0 \leq \lambda \leq 1$. If $\lambda = 0.5$, the decision rule is equivalent to that of unweighted classification. This type of stream weighting has previously led to great accuracy improvements for audio-visual speech recognition, e. g., in [40].

To achieve optimal stream weights for audio-visual identification, we will propose a cost function in the following. This cost function can provide optimal stream weights based on the true speaker's identity. Hence, the resulting stream weights can be used as training targets for learning a mapping function f, which uses confidence measures as its input and outputs estimated stream weights.

Thus, we also need appropriate confidence measures. For this purpose, we have considered a range of metrics. Among those, the dispersion $\mathcal{D}$ and different estimators of the distortion or noise level of the video and audio files have shown to provide reliable confidence measures. The dispersion is computed over the posterior probabilities obtained for one test file:

$$\mathcal{D}_i = \frac{2}{K(K-1)} \sum_{l=1}^{K-1} \sum_{m=l+1}^{K} \log \frac{\mathrm{P}(C_1^*|\boldsymbol{x}_i)}{\mathrm{P}(C_m^*|\boldsymbol{x}_i)}, \tag{2.8}$$

where the K classes $C_1^*, \dots, C_K^*$ with the largest probabilities are used, sorted in descending order of likelihood. The value of K can be lower than or equal to the number of enrolled speakers.

The noise level of the audio signals, denoted by κ_A, is estimated by a minimum mean-square error log-spectral amplitude estimator [48], for which we have used the MATLAB implementation provided by [49].

To estimate the image distortion (denoted by $\boldsymbol{\kappa}_V$) three different values are considered, i. e., for each image, the lighting condition, the degree of blurring, and the rotation are estimated and used as confidence measures in a vector

$$\boldsymbol{\kappa}_V = [\kappa_{V,L}, \kappa_{V,B}, \kappa_{V,R}]. \tag{2.9}$$

As the feature $\kappa_{V,L}$ for the lighting conditions (providing information of whether an image is overexposed or underexposed), the mean pixel value over all pixels is calculated. A potential blurring, e. g., due to the speaker's movements during image capture, is estimated by applying a Laplacian filter kernel for edge detection. To obtain one feature value $\kappa_{V,B}$, we calculate

$$\kappa_{V,B} = \sigma^2(\mathcal{I}_L), \tag{2.10}$$

where σ represents the variance and $\mathcal{I}_L$ is the image after edge detection.

The last confidence measure $\kappa_{V,R}$, representing a potential rotation of the speaker's head, is obtained by horizontally mirroring the image and calculating the cross-correlation between the original and the mirrored image. Moreover, to obtain blurring and rotation measures independently of lighting conditions, $\kappa_{V,B}$ and $\kappa_{V,R}$ are computed from the depth image.

Furthermore, we suggest using a function f, which maps all confidence measures to an optimal weight in the sense of our decision rule in Equation (2.7):

$$\hat{\lambda} = f(\mathcal{D}_A, \mathcal{D}_V, \kappa_A, \boldsymbol{\kappa}_V). \tag{2.11}$$

Before the mapping function f can be used as given in Equation (2.11), it has to be learned. For this purpose, we are utilizing supervised machine learning approaches.

To obtain a large number of training targets for learning the mapping function f, both speaker identification systems—audio and video—first need to be trained and in the second step, predictions with various data sets have to be calculated. The development set contains N_{DS} files under each of the different acoustic and visual conditions. In our case, $N_C = 10$ conditions are utilized for each single-modality recognition system. For these data sets, the corresponding dispersion and noise levels are computed. After that, $N = N_{DS} \cdot N_C^2$ input cases for the function f can be formed. In order to learn the mapping function f with these N tuples, we will need training targets, i. e., optimal stream weights for the entire range of the development set.

Optimal Stream Weights. In the following, we suggest an approach to find the optimal stream weights λ_ψ for all cases in the development set. The approach leads to ideally discriminative stream weights insofar as it maximizes the ratio of the likelihoods of the true speaker C_{true} and the most likely competing speaker C_{conf}.

For this, we assume C_{true} is the true class of the input feature vector $\boldsymbol{x}_i$, while C_{conf} is the class that is most likely to be confused with the true identity, i. e.,

$$C_{\text{conf}} = \underset{\forall C_k \setminus C_{\text{true}}}{\arg\max}\, \mathrm{P}(C_k|\boldsymbol{x}). \tag{2.12}$$

Therefore, to find the optimal value of λ_ψ, we suggest to maximize the following discriminative cost function for every file in the development set:

$$\begin{aligned} \lambda_\psi &= \arg\max_{\lambda} \left[\frac{\mathrm{P}(C_{\text{true}}|\boldsymbol{x}_V, \boldsymbol{x}_A)}{\mathrm{P}(C_{\text{conf}}|\boldsymbol{x}_V, \boldsymbol{x}_A)} \right] \mathrm{P}(\lambda) \\ &= \arg\max_{\lambda} \left[\log \mathrm{P}(C_{\text{true}}|\boldsymbol{x}_V, \boldsymbol{x}_A) - \log \mathrm{P}(C_{\text{conf}}|\boldsymbol{x}_V, \boldsymbol{x}_A) \right] \mathrm{P}(\lambda), \end{aligned} \tag{2.13}$$

such that $0 \leq \lambda \leq 1$, where the joint distribution probabilities are defined according to (2.7):

$$\mathrm{P}(C_k|\boldsymbol{x}_V, \boldsymbol{x}_A) = \mathrm{P}(C_k|\boldsymbol{x}_V)^{(1-\lambda)} \mathrm{P}(C_k|\boldsymbol{x}_A)^{\lambda}. \tag{2.14}$$

In this way, the distance between the posterior probability of the true and the second class is maximized. If one of the two recognition systems makes a wrong prediction, $\mathrm{P}(C_{\text{conf}}|\boldsymbol{x})$ will be higher than $\mathrm{P}(C_{\text{true}}|\boldsymbol{x})$. This effect can typically be mitigated through the choice of better weighting, and an optimal λ is obtained through maximizing Equation (2.13).

As in [40], we assume that the optimal λ should follow a prior distribution $\mathrm{P}(\lambda) \sim \mathcal{N}(\mu, \sigma)$. In our work, μ is obtained during a search, testing different values for μ between 0 and 1 with a step size of 0.01 and using that value which leads to the highest recognition rate. To refine the result, the variance σ^2 is increased iteratively until all λ_ψ become 0 or 1, or a maximum number of iteration steps is reached. Considering all possible λ_ψ of all iterations steps, the λ_ψ that lead to the best recognition rate are used as *oracle weights*.

Since the true class identity is used for this computation, these stream weights λ_ψ are referred to as *oracle weights*. They can thus only serve as training targets for learning f in Equation (2.11), but are not applicable in practice for speaker identification.

Models for estimating λ. Based on the above considerations, a method for generating training targets λ_ψ for the function f is available, but an appropriate model for the function still needs to be chosen. Since the weights λ can be in the range of $[0, 1]$, finding optimum weights becomes a regression problem rather than a classification problem. Experiments have been carried out with *feed-forward neural networks*, either shallow, or DNNs, for the mapping function.

2.1.3 Experimental Results

For the experiments, a data set was employed, which we had recorded with a Kinect sensor from Microsoft. We therefore considered the following data, provided by the Kinect sensor: the four-channel microphone array, the Full HD video, and the captured depth images. The data set contains 30 speakers (15 female and 15 male), with recorded utterances of English digits from 0 to 9. For this work, 4 digits were concatenated randomly for one training or test unit, in order to obtain a longer utterance.

During our experiments, we used all $N_S = 30$ speakers in each phase. This includes the enrollment during the training, the development set to learn the mapping function, and the test set to verify the obtained mapping function. Such a so-called closed-set identification does not consider non-enrolled speakers, like impostors. However, in this work we focus only on the optimal combination of different modalities for speaker identification among enrolled speakers and do not consider impostors.

In order to train the speaker and face recognition as described in Section 2.1.1, a total number of $N_F = 30$ utterances per speaker of the introduced data set were used.

The images were used as gray scale images and processed with the LBP as described in Section 2.1.1. For this, one image of each utterance was chosen for the recognition. The best results for the face recognition could be achieved with a radius of 2 pixels for the video image and a radius of 3 pixels for the depth image. We observed that it is possible to implement a robust face recognition solely using the depth images. Thus, the depth image seems to be a valuable contribution, regarding the robustness and security of face recognition. For our experiments, we used both, the video and the depth images.

For the speaker recognition, the best results were achieved with 24 MFCCs, augmented with their first- and second-order derivatives. The speaker recognition with i-vectors was applied to one channel of the recorded microphone array. To this end, we chose 128 mixture components for the GMM-UBM.

After the training phase, numerous data sets need to be created in order to learn the mapping function. For this purpose, $N_F = 20$ new utterances per speaker were considered for the development set, which had not been used to train the sub-systems. In order to simulate adverse environmental conditions, we added varying amounts of noise to the audio test files and introduced distortions to the video data. For acoustic speaker recognition, white Gaussian noise was added to the utterances. In total, 9 different noisy audio test cases with *Signal-to-Noise Ratios* (SNRs) between 4 dB and 20 dB were created and the original utterances were also included in the test cases.

For the images, different kinds of distortions were considered, i. e., different lighting, blurring, and rotations. To the depth images, which are independent of the lighting, only the blurring and rotations were applied. For the lighting distortions, we manipulated the pixel values, in order to simulate different conditions. To blur the images, we convolved the image with a filter kernel, which describes the direction and amount of motion for each test case. For this, the kernel is chosen such that the smoothing values are either concentrated in the focus of the kernel (less blurring) or are more distributed in the defined direction (more blurring). In Figure 2.1 all $N_F = 10$ conditions are shown with the original image located on the bottom right.

Overall, $N_F^2 = 100$ different combinations of acoustic and visual conditions were formed to create the development set. Each condition contains $N_S \cdot N_F = 30 \cdot 20$ recordings. For those combinations, the *oracle weights* and the confidence measure values were computed according to Section 2.1.2.

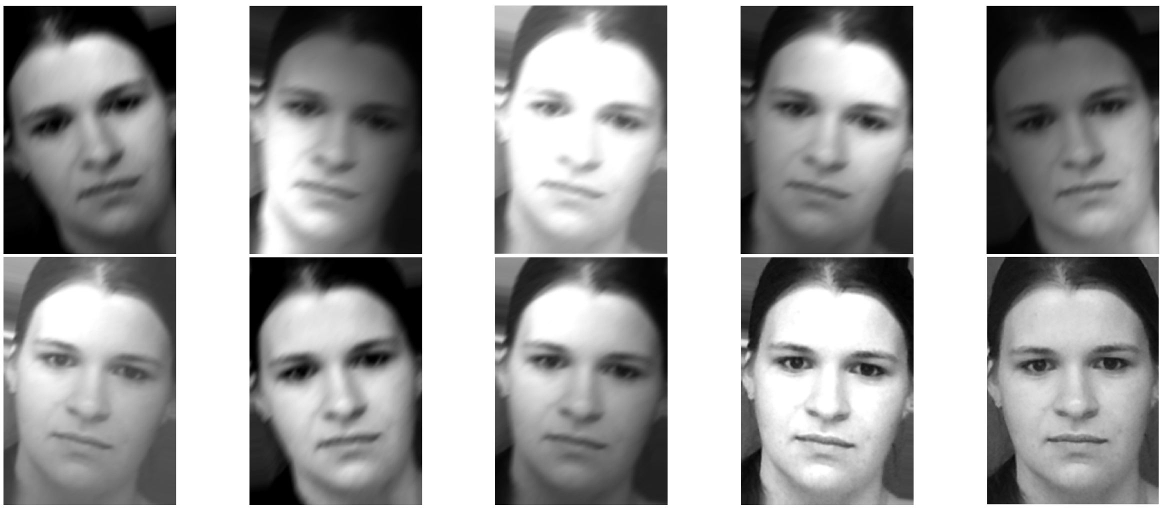


Figure 2.1: For the face recognition, different kinds of distortion were considered, i. e., adverse lighting conditions, blurring, and rotations.

Weight Estimation

After the calculation, the values of the *oracle weights* and the confidence measures from the complete development set are used to train a mapping function. For this purpose, we used *feed-forward neural networks* with different numbers of hidden layers. Additionally, we tested different combinations of confidence measures. For the dispersion calculated by Equation 2.8 we chose $K = 7$.

To assess the performance, the recognition rate R is calculated by

$$R = \frac{N_{\mathrm{rec}}}{N_S \cdot N_F}, \tag{2.15}$$

where N_{rec} denotes the number of correctly classified files. In order to verify the mapping function computed by the *Neural Network* (NN), a test set, considering $N_F = 20$ new utterances per speaker, was used.

In Table 2.1, an overview is presented of the different numbers of hidden layers (with 10 neurons per hidden layer) combined with different sets of confidence measures as inputs for the mapping function. Here, the first value in each cell is calculated with the development set (dev) and the second value with the test set. For this, we always used the output $\hat{\lambda}$ of the mapping function, trained using the confidence measures as denoted in the first column.

Table 2.1: Recognition rates for different settings. Best values are shown in bold.

Number of Hidden Layers:		1	2	3	4
$f(\mathcal{D}_A, \mathcal{D}_V, \kappa_A, \kappa_V)$	dev	0.939	0.940	**0.941**	0.941
	test	0.932	0.932	**0.932**	0.932
$f(\kappa_A, \kappa_V)$	dev	0.936	0.940	0.940	0.940
	test	0.927	0.930	0.929	0.931
$f(\mathcal{D}_A, \mathcal{D}_V)$	dev	0.928	0.928	0.928	0.928
	test	0.914	0.914	0.914	0.914

By comparing the results of the development set and the test set, one can see that the mapping function remains robust for unseen data. Further, using only the dispersion values or the estimated noise/distortion levels leads to almost equally high recognition rates as using all confidence measures. Moreover, changing the number of layers does not substantially affect the recognition rate. However, using 3 hidden layers and all confidence measures led to the highest accuracy for combinations using clean audio and video data and was therefore chosen for the following experiments.

In Table 2.2 the results for this setting (bold values in Table 2.1) are shown in detail. Here, all $N_F^2 = 10 \cdot 10$ possible combinations of the test conditions are presented. In the gray cells, the recognition rate of the single-modality systems are displayed. In the remaining cells, the combined recognition is shown. Here, the top value represents the recognition rate obtained with the baseline approach ($\lambda = 0.5$). The second value is the recognition rate achieved with the weights $\hat{\lambda}$, estimated by the DNN.

As one can see, in the results for the baseline system with $\lambda = 0.5$, the audio recognition shows high recognition rates for test cases with high SNR and low recognition rates for test cases with low SNR. On the other hand, the combined recognition rate barely seems influenced by the face recognition.

When the noise level increases, one would expect a classifier to yield to a flat posterior probability distribution, as it will not be able to reliably differentiate between the classes. Yet, classifiers applied outside of their training domain, i. e., trained on clean data and applied on noisy data, have a tendency to yield peaked distributions,

Table 2.2: Recognition rates achieved on the test set for every combination of audio noise and video distortions. The rows present the different image distortions (ID1 – ID9) and the original image (OI). The different audio test cases are presented in the columns. The bold values show the recognition rates for the single-modality systems.

			4 dB	6 dB	8 dB	10 dB	12 dB	14 dB	16 dB	18 dB	20 dB	clean
λ			**0.30**	**0.40**	**0.50**	**0.60**	**0.71**	**0.78**	**0.89**	**0.94**	**0.98**	**1.00**
ID1	0.5	**0.39**	0.35	0.45	0.55	0.66	0.73	0.81	0.91	0.95	0.98	1.00
	$\hat{\lambda}$		0.57	0.68	0.74	0.79	0.83	0.88	0.92	0.96	0.98	1.00
ID2	0.5	**0.45**	0.35	0.43	0.54	0.64	0.74	0.83	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.66	0.71	0.77	0.82	0.89	0.93	0.96	0.98	0.99	0.99
ID3	0.5	**0.56**	0.36	0.44	0.54	0.66	0.75	0.84	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.74	0.78	0.84	0.89	0.93	0.98	0.99	0.99	0.99	0.99
ID4	0.5	**0.64**	0.35	0.45	0.54	0.65	0.75	0.84	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.79	0.83	0.87	0.91	0.94	0.96	0.98	0.98	0.99	0.99
ID5	0.5	**0.68**	0.36	0.45	0.55	0.66	0.74	0.82	0.91	0.95	0.98	1.00
	$\hat{\lambda}$		0.74	0.79	0.83	0.86	0.90	0.94	0.97	0.98	0.99	1.00
ID6	0.5	**0.76**	0.35	0.45	0.55	0.66	0.74	0.84	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.85	0.89	0.93	0.96	0.98	0.98	0.99	0.99	1.00	1.00
ID7	0.5	**0.85**	0.36	0.46	0.56	0.67	0.74	0.83	0.91	0.96	0.98	1.00
	$\hat{\lambda}$		0.88	0.90	0.93	0.96	0.98	0.99	1.00	1.00	1.00	1.00
ID8	0.5	**0.87**	0.36	0.46	0.55	0.66	0.75	0.84	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.92	0.96	0.97	0.98	0.99	1.00	1.00	1.00	1.00	1.00
ID9	0.5	**0.98**	0.36	0.44	0.54	0.66	0.74	0.83	0.92	0.96	0.98	1.00
	$\hat{\lambda}$		0.97	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
OI	0.5	**1.00**	0.35	0.44	0.53	0.64	0.73	0.81	0.91	0.95	0.98	1.00
	$\hat{\lambda}$		1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

preferring certain classes [41]. This contradicts the assumption of the purely Bayesian fusion for $\lambda = 0.5$ and yields inferior results for the unweighted fusion with increasing noise levels (cf. Table 2.2). The weighting of the streams counteracts this effect by introducing an external signal for classifier confidence.

In contrast, the mapping function, obtained by the DNN, performs just as expected: It increases all recognition rates up to at least very close to the highest recognition rate of the single-modality systems. For a few combinations at high SNRs, there are slight decreases in comparison to the baseline result. However, the recognition rate in these cases remains very high and the changes in accuracy never exceed 1 %.

On the whole, for the complete test set, the average improvement was from 74.17 % to 93.23 %. This shows the applicability of the presented approach also in those situations where stream weights are not based on oracle information, but rather estimated from the newly suggested approach, based on easily estimated confidence values.

2.2 Audio-Visual Spoofing Detection

In a spoofing attack against biometrics, a malicious party tries to imitate another person's biometrics. One approach to increase the robustness against spoofing attacks is to use multimodal biometrics, e. g., audio-visual speaker verification [50]. However, if no countermeasure is implemented, multimodal systems have shown to be vulnerable against spoofing attacks imitating only one trait successfully, since they often focus on the trait with the least distortions [51, 52].

In general, different kinds of spoofing attacks against audio-visual authentication need to be considered. This encompasses replay attacks, where an impostor uses a previously recorded utterance of the victim, e. g., a video of the entire identification process or a recording of the audio channel and an additional visual input, like an arbitrary image or video of the victim.

Playing back a synthesized version of an utterance constitutes another spoofing attack. Synthesizing has the advantage that it is more flexible in a challenge-response system, where the user is asked to utter different and previously unknown sentences in each session for liveness detection [53, 54]. However, in contrast to recordings, synthesized videos are more difficult to access and the movements of the lips often appear artificial [55].

To prevent attacks, the task is to distinguish between a spoofing attack and a genuine speaker such that a sophisticated attack can be detected, but a genuine speaker is not rejected.

We use CHMMs to simultaneously verify the audio-visual synchronicity and transcription in a challenge-response setup, where the person-to-identify has to utter a predefined sentence. CHMMs have proven successful in audio-visual speech recognition, increasing the robustness of speech recognition in adverse conditions [56, 40]. For audio-visual speech recognition, CHMMs are more appropriate than HMMs with early feature fusion, as they allow slight asynchronicities between feature streams, which gives them a significant advantage regarding the recognition performance.

We will use and expand their capability to handle and detect asynchronicity in the following, which will allow us to employ them for simultaneous verification of audio-visual synchronicity and spoken content of the utterance. The proposed spoofing is applicable in a deep-learning-based approach for the speech recognition in an equivalent manner. However, this work focuses on spoofing detection and due to the limited data available here, a *Gaussian Mixture Model Hidden Markov Model* (GMM-HMM)-based CHMM system is a good starting point that already allows us to explore the applicability of different feature sets for verification purposes.

2.2.1 Spoofing Detection

In order to recognize a synchronization mismatch between the audio and the video data, we use CHMMs, so that we can simultaneously recognize both the audio and the video transcription, and any time difference between the audio and the video stream.

Coupled HMMs

CHMMs are an extension of HMMs that is particularly useful for combining different streams in a multimodal system without the necessity of fusion on the feature level. For the construction of the CHMMs, it is necessary initially to represent each single word as a uni-modal HMM. In general, for audio-visual speech recognition with CHMMs, the two marginal HMMs, one for each stream, are trained separately for each word. During the training of the HMMs, the conditional observation likelihoods

$$b_i(\boldsymbol{o}(t)) = \mathrm{P}(\boldsymbol{o}(t)|q(t) = i), \tag{2.16}$$

for state $q(t) = i$ are calculated based on the observations $\boldsymbol{o}(t)$ of the stream for the frame at time step t. Additionally, the state transition probabilities $a(i, j)$ are obtained during training. The probability of going from state $q(t) = i$ to state $q(t+1) = j$ in a discrete time step $t \rightarrow t + 1$ is

$$a_{ij} = \mathrm{P}(q(t+1) = j|q(t) = i). \tag{2.17}$$

As in a speech recognition application, the state transitions are defined such that the model can not step back into a previous state:

$$a_{ij} = 0, \quad \forall i > j. \tag{2.18}$$

For a CHMM, all states $Q^A = \{q_i^A\}_{i=1}^{N^A}$ of the audio HMM are combined with all states $Q^V = \{q_1^V\}_{i=1}^{N^V}$ of the visual HMM such that the resulting CHMM has $N = N^A \cdot N^V$ states. The new conditional observation likelihoods $b_{\boldsymbol{i}}(\boldsymbol{o}(t))$ for each coupled state are combinations of the corresponding conditional observation likelihoods of the audio A and the video V stream

$$b_{\boldsymbol{i}}(\boldsymbol{o}(t)) = \mathrm{P}(\boldsymbol{o}^A(t)|q^A(t) = i^A)\mathrm{P}(\boldsymbol{o}^V(t)|q^V(t) = i^V), \tag{2.19}$$

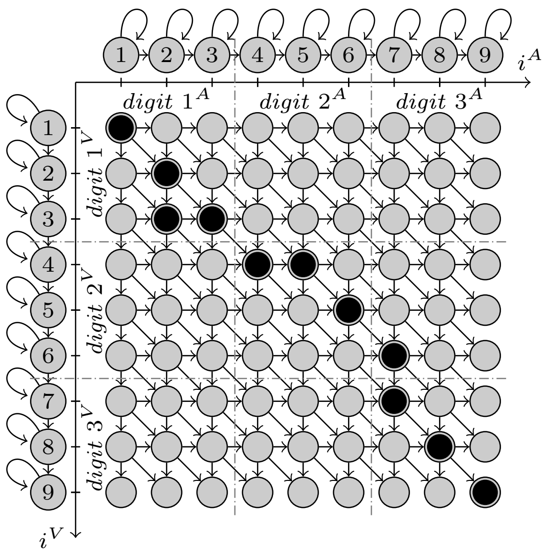
with $\boldsymbol{i} = [i^A, i^V]$ describing the coupled state as a combination of the single-modality states i^A and i^V. The feature vectors $\boldsymbol{o}^A(t)$ and $\boldsymbol{o}^V(t)$ are obtained from the audio and the video stream, respectively. The state transitions for the CHMM are calculated by:

$$a_{\boldsymbol{ij}} = a_{i^A j^A} \cdot a_{i^V j^V}, \tag{2.20}$$

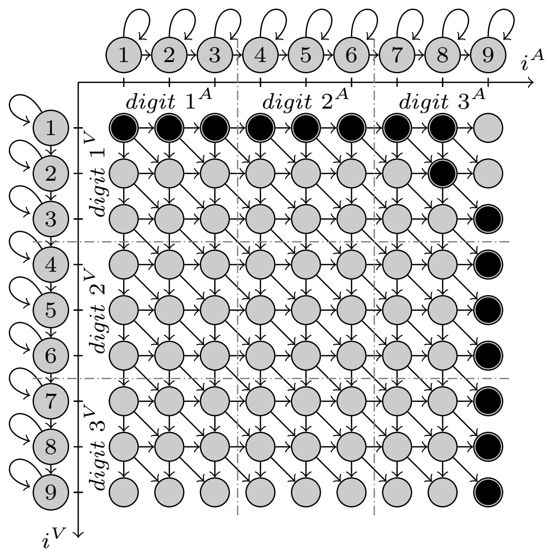
with the coupled states $\boldsymbol{i} = [i^A, i^V]$ and $\boldsymbol{j} = [j^A, j^V]$.

CHMMs for Spoofing Detection

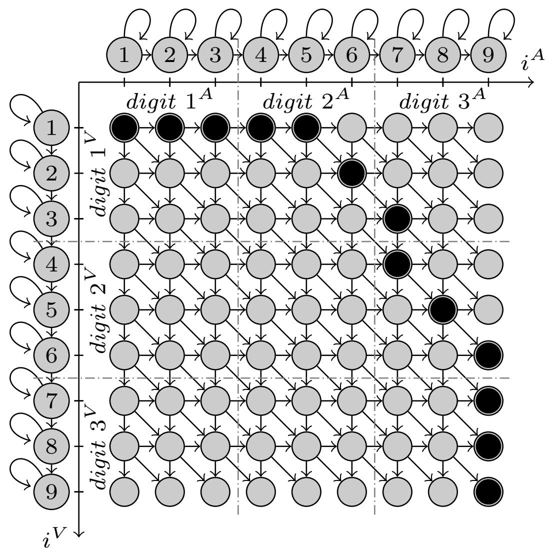
For audio-visual speech recognition, the corresponding audio HMM and video HMM are, like in [57] and [56], used as marginal HMMs creating a combined word CHMM as a Cartesian product model, cf. Equations (2.19) and (2.20). The single word models may then be combined according to a task grammar. With this approach, the audio and the visual streams can be asynchronous within one word, but not across different words. This is sufficient for audio-visual speech recognition since the audio and the visual stream can usually be assumed to be synchronous.



(a) Synchronous streams



(b) Invalid visual stream



(c) Asynchronous streams

Figure 2.2: The CHMMs for different scenarios and possible paths through the CHMM with $M = 3$ digits as the challenge: Figure 2.2a shows a synchronous utterance, Figure 2.2b is an example of an utterance with an invalid visual stream, and Figure 2.2c shows an utterance with asynchronous streams.

Construction. In the case of a spoofing attack, where either the two streams do not match, or one stream is missing completely, the synchronicity may not be given. For a spoofing detection, this asynchrony can be assessed. Thus, in the following, grammar models are built on the level of the single HMMs and combined to one CHMM for the entire utterance. With this approach, the audio and the visual streams may be in completely different words at the same time step.

For our experiments, a sequence of random digits is used as the utterance. With ten different digits (‘zero’ to ‘nine’), 10^M combinations of different utterances are possible, where M is the number of digits in the sequence.

In Figure 2.2, different spoofing scenarios and their possible paths through a CHMM are sketched for an utterance with $M = 3$ digits. For easier visualization, the CHMMs in the figure are simplified. In general, the audio HMM requires more states than the video HMM, but here, they are depicted for $N^A = N^V$, and we show only three digits. The CHMM used for the spoofing detection has many more states and thus possible paths. Further, the different digits do not necessarily have the same number of states and not all possible state transitions are sketched. In general, transitions are only possible top-to-bottom and left-to-right, according to Equations (2.18) and (2.20).

In Figure 2.2a, a possible path for a synchronous utterance is shown. Although the streams are synchronous, the recognized digits still have to be compared to the challenge. Figure 2.2b depicts a spoofing scenario with an invalid visual stream (e. g., a still image of the victim). In this example, the visual recognition stays in the first visual state, while the audio recognition proceeds. Due to the structure of the CHMM, for the video, any arbitrary transcription will be recognized as well. Figure 2.2c represents a spoofing scenario where the two streams are not synchronous. In the latter examples, an analysis of the coupled state sequence provides useful information about the synchronicity.

Resource Optimization. Due to the combinatorial nature of our CHMM construction scheme and the resulting high number of coupled states, the computations would get infeasible, if we were to evaluate all $(10 \cdot M)^2$ possible combinations of digits in one compound CHMM. Therefore, we limit the construction of the marginal HMMs for asynchrony detection to only the most likely digits at each of the M positions. To obtain these digits, the M^* best digits are determined for each position for the audio and the video stream. These resulting $2 \cdot M^*$ digits per position are considered to construct the two marginal HMMs. Since some of the $2 \cdot M^*$ digits of each position will often be recognized by both, the audio and the video model, among M^* best digits, the redundant digits are discarded for the construction of the marginal HMMs. The resulting CHMM has at most $(2 \cdot M^* \cdot M)^2$ combinations of digits. This reduces the computational cost significantly.

2.2.2 Proposed Features

For the recognition, the forward-backward algorithm is used to obtain the matrix Γ with $N \times T$ values, describing the probabilities for being in all coupled states at time $t = 1, \dots, T$. With the Viterbi algorithm and Γ, the most likely path through the CHMM is calculated. The resulting path is a sequence of coupled states

$$\boldsymbol{q} = \Big[q(1) = [i^A(1), i^V(1)], \dots, q(T) = [i^A(T), i^V(T)]\Big], \tag{2.21}$$

describing the recognized coupled states in the order of recognition.

Synchronicity Features

The audio HMMs are defined with three states per phoneme, whereas the video HMMs use only one state per phoneme. Thus, the time alignment difference between the audio and the video stream is calculated via:

$$\lambda(t) = \left\lceil \frac{i^A(t)}{3} \right\rceil - i^V(t). \tag{2.22}$$

In the case of a genuine utterance, the values of $|\lambda|$ should be small. In contrast, a spoofed utterance with non-matching streams will typically show larger values. As features for the recognition, two different values have shown to be useful, the entropy $\mathcal{H}$ of the time alignment difference λ, which is higher for asynchronous utterances and the mean value of $\lambda(t)$, denoted by Λ, over all time steps $t = 1, \dots, T$.

However, using only these features, the audio and the video stream may appear synchronous, even if the recognized digits of the streams are different, especially, if the two different digits have the same number of states. As a proposed countermeasure, an additional CHMM is constructed containing only the challenged digits. With this CHMM, the distances $\lambda_\kappa(t)$ are calculated according to Equation (2.22). In a genuine scenario, both distance vectors λ and λ_κ should be very similar. Therefore, we propose two more features:

$$\Lambda_\kappa = \frac{1}{T}\sum_{t=1}^{T} \lambda(t) - \lambda_\kappa(t), \tag{2.23}$$

$$\Lambda_{|\kappa|} = \frac{1}{T}\sum_{t=1}^{T} |\lambda(t) - \lambda_\kappa(t)|. \tag{2.24}$$

Although the measures are similar, the combination of Λ_κ and $\Lambda_{|\kappa|}$ leads to a more robust recognition.

Transcription Features

As additional features for the spoofing detection, the transcriptions of both streams, obtained with the CHMM-based recognition, are used. This is necessary to prevent replay attacks with videos, where the streams may be synchronous, but do not contain the utterance of the challenge.

To detect this situation, the differences of the audio transcription $\tau_A(m)$ and video transcription $\tau_V(m)$ to the challenged digits $\tau(m)$ over all positions $m = 1, .., M$ are calculated with the Hamming distance, such that each substituted digit increases the calculated distance by 1. Thus, the resulting distances may be between 0 and M and are considered as features τ_A and τ_V for the distance of the audio transcription and the video transcription, respectively.

2.2.3 Experimental Results

For the experimental evaluation, we have used a set of 30 speakers (15 female and 15 male) with utterances of single digits from 0–9. 270 utterances of each speaker are used. The dataset is recorded with Microsoft's Kinect sensor, which also provides reliable information about the location of the mouth region. In the following experiments, we have used the first of the four available microphone channels.

As audio features, the first 13 MFCCs and their first and second derivatives have been considered. As video features, we have used the first 8×8 coefficients of a two-dimensional DCT of the cropped mouth region.

In Figure 2.3 the training and spoofing detection is sketched. 170 recordings for each of the 30 speakers have been used in HMM training to get a speaker-independent

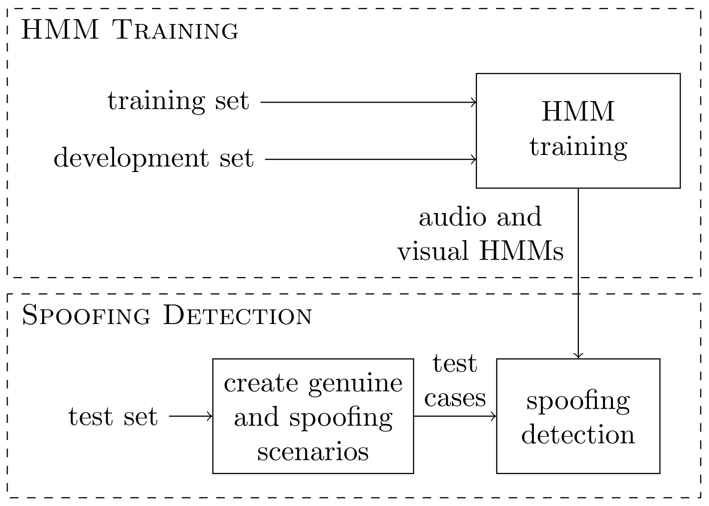


Figure 2.3: The training set is used for the training of the audio and the visual HMMs of the single digits. The development set is considered to verify the performance of the speech recognition system. With the test set, the genuine, and spoofing scenarios are built for the evaluation of the spoofing detection, which uses the trained HMMs. Utterances from all 30 speakers are used for the training, development, and test set.

audio-visual speech model. Further 40 recordings per speaker have been used as the development set to verify the speech recognition performance. The remaining 60 recordings per speaker have been deployed to create the genuine and spoofing scenarios. The spoofing detection is tested together with the scenarios and the trained HMMs.

For the test set $M = 3$ digits were concatenated per utterance to build 20 utterances per speaker for each spoofing scenario and the genuine utterances. We have used $M^* = 3$ for all experiments, such that 3–6 digits per position are considered to construct the marginal HMMs. This is a trade-off between the complexity of the resulting CHMM and the probability of obtaining the uttered digit in the CHMM decoder.

The spoofing detection has been applied in a challenge-response setup. Thus, a recorded video of the victim with the correct utterance is hard to access for the attacker, since the utterance changes for each verification process. However, it is still possible to use recorded videos with different utterances or to use a modified or synthesized video. Such artificially constructed videos may show a delay between audio and video which can be detected, even if the utterance is correct.

To create test scenarios, different combinations of the audio and video stream have been created. In all cases the audio and the video stream is from the same speaker:

- Scenario #1 (still image): The audio stream is the genuine utterance corresponding to the challenge and the video stream is only one image for the entire utterance.
- Scenario #2 (cross-video): The audio stream is the genuine utterance corresponding to the challenge and the video stream is replaced by an arbitrary other one.
- Scenario #3 (wrong utterance): The audio stream and the video stream do match, but do not correspond to the digits of the challenge.
- Scenario #4 (delayed): The audio and the video stream correspond with the challenge, but they have a delay ($\pm 1\,\mathrm{s}$, $\pm 0.5\,\mathrm{s}$, and $\pm 0.25\,\mathrm{s}$).

Baseline System Description

For comparison, we also have implemented one of the latest spoofing detection approaches for speaker verification [58]. This approach can also be used in a challenge-response setup. Additionally, it also does not need specifically enrolled utterances for the spoofing detection. However, in contrast to our method, the baseline method is speaker-dependent. Thus, for training, it needs utterances from each enrolled speaker.

For the spoofing detection, the baseline system uses *Canonical Correlation Analysis* (CCA) to compare the audio and the video stream [59]. For this purpose, the projection matrices W and Z (canonical correlation matrices) are calculated with the training set for each speaker separately. The score for the spoofing detection is calculated by

$$S(\mathcal{A}, \mathcal{V}) = \frac{1}{N} \sum_{n=1}^{N} \operatorname{corr}(\mathcal{A} w_n, \mathcal{V} z_n), \tag{2.25}$$

where w_n and z_n are the n^{th} column of the projection matrices W and Z, respectively, and $\mathcal{A}$ and $\mathcal{V}$ are the audio and the video stream of the test scenario, respectively. The parameter N is tuned on the development set. Hence, only the synchronicity, but not the transcription is verified. Therefore, it is not possible to detect scenario

#3 with synchronous video, but the wrong utterance, so this scenario has not been considered in our evaluation of the baseline method.

As features for the baseline method, the same ones as in [58] have been used in the evaluation, since these provided the best results. These are MFCCs for the audio data and space-time auto-correlation of gradients (STACOG) for the visual data [60].

Table 2.3: EER (in %) of different features for the spoofing scenarios and their combinations. The input features are synchronicity features ($SF = [\mathcal{H}, \Lambda, \Lambda_{\kappa}, \Lambda_{|\kappa|}]$), transcription features ($TF = [\tau_A, \tau_V]$), and both together (all). As the baseline, the approach in [58] has been used.

	SF	**TF**	**all**	**baseline**
Scenario #1	3.25	13.75	**1.50**	3.85
Scenario #2	10.50	12.25	**6.25**	21.49
Scenario #3	10.75	**1.75**	2.00	—
Scenario #1–2	5.68	14.45	**5.18**	16.42
Scenario #1–3	8.83	12.50	**5.50**	—
Scenario #4 (±1 s)	**2.34**	43.28	2.71	14.52
Scenario #4 (±0.5 s)	3.10	52.27	**2.86**	14.50
Scenario #4 (±0.25 s)	**8.88**	51.25	9.40	10.85
Scenario #4 (all)	6.96	54.88	**6.46**	13.45
Scenario #1,2,4	**5.89**	55.06	7.11	14.13
Scenario #1–4	8.69	56.72	**6.83**	—

Results

To build a spoofing detection with the different proposed synchronicity and transcription features, support vector machines (SVMs) have been employed in the following. For this purpose, we have evaluated the synchronicity features $SF = [\mathcal{H}, \Lambda, \Lambda_{\kappa}, \Lambda_{|\kappa|}]$ and the transcription features $TF = [\tau_A, \tau_V]$ separately, and all of these features together. Table 2.3 provides an overview of the *Equal Error Rate* (EER) for the different spoofing scenarios and combinations of those. Scenario #4 is separated into different groups, considering different delays.

Table 2.4: EER (in %) for the cross-speaker verification. The speakers in the first row are left out and used to evaluate the spoofing detection.

	S1–S5	**S6–S10**	**S11–S15**	**S16–S20**	**S21–S25**	**S26–S30**	**average**
Scenario #1	1.00	4.50	2.00	2.00	2.50	3.00	2.50
Scenario #2	8.00	10.00	10.00	7.00	8.00	7.50	8.42
Scenario #3	1.00	3.00	2.00	1.00	4.50	3.00	2.42
Scenario #4	4.50	11.17	9.00	7.42	5.50	12.25	8.31
Scenario #1–4	3.56	14.44	12.67	7.78	6.78	10.17	9.23

In some cases, the different features lead to very different EERs. In Table 2.3, the best results are marked in bold. For most spoofing scenarios, a combination of all features leads to the best EER, and especially if the different spoofing scenarios are averaged, a combination of all features clearly provides the best results. In general, the difference of the EER of the single feature groups and the combination never exceeds 1.22 % and in many cases, the combination is better by a large margin.

Scenario #2 benefits the most from the combination of both features, since the video-only speech recognition is not as reliable as the audio-only recognition, such that the mismatch of the visual stream is not always detected, and a genuine, matching visual stream can sometimes be falsely classified as spoofed. Additionally, the synchronicity measure is not as robust here as for scenario #1 were only one image is used for the whole visual stream. Since, overall, both features perform about equally well in this scenario, large improvements are possible due to their complementary information. For scenario #4, the synchronicity features are more valuable than the transcription features. This is no surprise, due to the capability of CHMMs to achieve a reliable recognition for asynchronous data, which clearly distinguishes them from early-integration-based approaches for this task. Therefore, the introduced distance features provide a robust measure of classification.

Comparison with Baseline System. In all cases, the combination of the synchronicity features and the transcription features leads to a lower EER in comparison to the results of the baseline system. Especially for scenario #2, much better results can be achieved with the proposed approach. This indicates that the CCA-based classification focuses on major signal changes. Thus, a wrong transcription in one stream is more difficult to detect.

Cross-Speaker Verification. For many use-cases of spoofing detection, it is not feasible to collect enough data from each enrolled speaker to train the spoofing detection. Therefore, a 6-fold cross-verification has been performed by leaving out a group of speakers during training of the spoofing classification. The utterances of this held-out group of speakers are used for the evaluation in Table 2.4. As input features for all cases, all introduced synchronicity and transcription features $[\mathcal{H}, \Lambda, \Lambda_\kappa, \Lambda_{|\kappa|}, \tau_A, \tau_V]$ are considered.

The results show that the EER is similar for unseen speakers in most of the cases. It can also be observed that some speakers seem to be easier to spoof (group S6-S10), while for some speakers (group S1-S5), the EER is even lower than the corresponding results of Table 2.3 (all features). Thus, some speakers seem to be more vulnerable to spoofing attacks, although all results point to a good performance in general.

2.3 Related Work

Alam et al. [61] have shown that a combined audio-visual speaker identification can increase the accuracy of the single-modality methods. However, in their approach, only the resulting ranks of the single systems are considered. In other audio-visual speaker recognition systems, the fusion is applied earlier: Yu and Huang [62] published an approach for feature fusion and Alam et al. [63] uses deep Boltzmann machines in combination with deep neural networks to fuse both modalities.

For speaker recognition using audio data only, many different approaches for spoofing and liveness detection exist. To counter replay attacks, Wu et al. [9] proposed an approach that uses a text-dependent recognition and different phrases for each identification process.

For the classification of synthesized utterances, the *Constant Q Transform* (CQT) has been shown to achieve robust results for an audio-only recognition [64].

Audio-visual speaker recognition provides much more information to verify a response [63], but only a few recent works have investigated audio-visual spoofing detection. All of these works use synchronicity measures either for a single utterance [65, 66] or multiple utterances [58, 67, 68].

Aides and Aronowitz [65] calculated the difference to stored sample utterances for the audio and the video channel separately via *Dynamic Time Warping* (DTW). The resulting time differences of both modalities are then compared to verify the utterance.

Further approaches use CCA to maximize the cross-correlation between matching audio and video frames [66, 58, 68]. Similar to these works, Bredin and Chollet [67] applied a *Co-Inertia Analysis* (CoIA) to the audio and video data to calculate features for a correlation.

All these approaches are still vulnerable to video replay attacks since they only measure the synchronicity. Particularly the approaches by Aides and Aronowitz [65] and Komulainen et al. [66] may not be able to distinguish between a recorded video and a genuine utterance, since the challenge does not change.

While HMMs have been used for audio-visual speaker verification [57, 69, 70], the authors do not verify their method for spoofing attacks or again only detect the synchronicity but do not consider the transcription. Especially, Rúa et al. [57] where the authors also use CHMMs, only an asynchrony detection is applied by considering major signal changes (e. g., starts or ends of words). Hence, a video replay attack is impossible to detect with this approach.

In recent years, also neural-network-based versions of i-vectors have been proposed [26]. These so-called *x-vectors* replace the JFA with a DNN and directly map the feature representation of an utterance into a voice profile.

2.4 Summary

We have shown that a better speaker identification can be obtained by a fusion of state-of-the-art audio-based and video-based identification. For this purpose, we have proposed a new weighting approach for the two modalities using a discriminative cost function to increase the ratio of the score of the true speaker relative to the speaker with the most competitive score. *Feed-forward neural networks* with multiple hidden layers led to the best results for computing these stream weights, based on a set of confidence measures.

Additionally, We have shown that with a multimodal recognition, it is fairly easy to deal with different kinds of noises or distortions added to the audio and video data by using an estimation of these distortions together with the dispersion of the scores.

On the whole, a large improvement across all considered test cases can be observed. Importantly, in all considered cases, we achieved at least principally the recognition rate we would achieve with the best single modality. Moreover, for the test cases with low recognition rates, we were always able to exceed these values, in many cases very notably.

In the second part, we have proposed a text-dependent audio-visual spoofing detection for speaker verification. For its evaluation, we have considered different spoofing scenarios, which can be used in a real attack.

We have introduced a CHMM-based synchronicity measure, which is available for spoofing scenarios with non-matching streams. Additionally, the assessment of the transcription with the CHMM-setup also provides a simultaneous verification of both feature groups, which can improve the classification in many cases where synchronicity metrics alone are not sufficient. Additionally, it is also possible to detect spoofing attacks that are synchronous but contain the wrong utterance. This shows the great advantage in contrast to approaches using synchronicity-based methods only.

Furthermore, via a cross-speaker validation, we have shown that the proposed spoofing detection can be used speaker-independently, so that new speakers can be enrolled with no extra effort.

The introduced approach needs an additional step for the training of the CHMMs. However, a speaker verification with changing utterances is much harder to spoof, since an attacker either needs to produce all possible utterances or has only limited time to produce a spoofing attack. Therefore, this additional training step should often be justified, and it only needs to be performed once, before system deployment.

In combination with an audio-visual speaker identification system both systems can benefit from each other. Thus, a more secure and robust audio-visual speaker recognition can be achieved.

The proposed approach is not limited to digits and can be used for arbitrary words or sentences as long as speaker-independent HMMs for speaker recognition can be trained.

For future work, a large-vocabulary version, based on triphone-level CHMMs including a deep learning approach should be investigated, to achieve a higher diversity of possible utterances for a still greater resilience against playback or synthesis.

3 | Audio Adversarial Examples

Substantial improvements in ASR accuracy have been achieved in recent years by using acoustic models based on DNNs that evolved into the state-of-the-art approach for many machine learning tasks [71, 72]. They can cope with complex, real-world environments that are typical for many speech interaction scenarios such as voice interfaces. In practice, the importance of DNN-based ASR systems is steadily increasing, e. g., within smartphones or stand-alone devices such as Amazon's Echo/Alexa.

Nevertheless, the number of necessary parameters is significantly larger than that of the previous state-of-the-art GMM probability densities within HMMs (so-called GMM-HMM systems) [73]. As a consequence, this high number of parameters gives much space to explore (and potentially exploit) blind spots that enable an adversary to mislead an ASR system.

In fact, current studies suggest that there can be significant differences in the mechanism of neural network algorithms compared to human expectations [4, 74]. This is a very unfortunate situation, as a rogue party can abuse this knowledge to create input data which leads to arbitrary recognition results, without being noticed [75, 76].

Possible attack scenarios include attacks over radio or TV, which could affect a large number of victims. This could lead to unwanted online shopping orders, which has already happened on normally uttered commands over TV commercials, as Amazon's devices have reacted to a purchase command [1]. ASR systems are often included into smart home setups may lead to a significant vulnerability and in a worst-case scenario, an attacker may be able to take over the entire smart home system, including security cameras or alarm systems.

In the audio domain, Vaidya et al. were among the first to explore adversarial examples against ASR systems [77]. They showed how an input signal (i. e., audio file) can be modified to fit the target transcription by adapting the features of the audio signal. Carlini et al. introduced so-called *hidden voice commands* and demonstrated that targeted attacks against HMM-only ASR systems are feasible [78]. They use inverse feature extraction to create adversarial audio samples. The resulting audio samples are not intelligible by humans (in most of the cases) and may be considered as noise, but may make thoughtful listeners suspicious. To overcome this limitation, Zhang et al. proposed so-called *DolphinAttacks*: they showed that it is possible to hide a transcription by utilizing non-linearities of microphones to modulate the baseband audio signal with ultrasound higher than 20 kHz [79]. The drawback of this and similar ultrasound-based attacks [80, 81] is that the attack is costly as the information on how to manipulate the input features needs to be retrieved from recordings of audio signals with the specific microphone, which is used for the attack.

Carlini and Wagner published the first adversarial audio examples in which they introduce a general targeted attack on ASR systems using the connectionist temporal classification (CTC) loss [76]. Similarly to previous adversarial attacks on image classifiers, it works with a gradient-descent-based minimization [75], but it replaces the former loss function by the CTC-loss, which is optimized for time sequences. The constraint for the minimization of the difference between original and adversarial sample is also borrowed from adversarial attacks on images and therefore does not consider the limits and sensitivities of human auditory perception. Yuan et al. described *CommanderSong*, which is able to hide transcripts within music [82], but their attack does not contain a human-perception-based noise reduction and the attack is clearly perceptible.

In contrast to previous work, in the following chapter we demonstrate an attack that utilizes psychoacoustics to add imperceptible or almost imperceptible noise to the original audio, which fools the ASR system to output a false—attacker-chosen—transcription. We further extend that attack by considering the room acoustics and include a room simulation via *Room Impulse Responds* (RIRs) to enable the attack to remain viable if played over the air. Finally, in the third part, a countermeasure utilizing neural networks that are capable of uncertainty quantification is described. Furthermore, to detect audio adversarial examples, we utilize an outlier detection to remain robust also for unknown attacks.

3.1 Adversarial Attacks via Psychoacoustic Hiding

We introduce a type of adversarial examples against ASR systems based on *psychoacoustic hiding* in order to reduce the perceptible noise. For this purpose, dynamic hearing thresholds are calculated based on psychoacoustic experiments by Zwicker et al. [10]. This limits the adversarial perturbations to those parts of the original audio sample, where they are not (or hardly) perceptible by a human listener. Furthermore, we use gradient descent to find adversarial examples with minimal perturbations. This algorithm has already been successfully used for calculating adversarial examples in other settings [75, 76] an to show the general feasibility of psychoacoustic attacks, for audio signals that are fed directly into the recognizer and also if played over the air.

A key feature of our approach is the integration of the preprocessing step into the DNN. As a result, it is possible to change the raw audio signal directly. The preprocessing operates as a feature extraction and is fundamental to the accuracy of an ASR system. Due to the differentiability of each single preprocessing step, we are able to include it in the gradient descent step without the necessity to invert the feature extraction. In addition, ASR highly depends on temporal alignment as it is a continuous process. We enhance our attack by computing an optimal alignment with the *forced alignment* algorithm, which calculates the best starting point for the gradient descent. Hence, we make sure to move the target transcription into parts of the original audio sample which are the most promising. We optimize the algorithm to provide a high success rate and to minimize the perceptible noise.

We have implemented the proposed attack to demonstrate the practical feasibility of our approach. We evaluated it against the state-of-the-art *Deep Neural Network Hidden Markov Model* (DNN-HMM)-based ASR system *Kaldi* [83], which is one of the most popular toolchains for ASR among researchers [84, 85, 86, 87, 88, 89, 90, 91, 82] and is also used in commercial products such as Amazon's Echo/Alexa and by IBM and Microsoft [92, 93]. Note that commercial ASR systems do not provide information about their system setup and configuration.

Such information could be extracted via model stealing and similar attacks (e. g., [94, 95, 96, 97, 98]). However, such a blackbox attack would go beyond the contributions of this work and hence we focus on the general feasibility of adversarial attacks on state-of-the-art ASR systems in a white-box setting. More specifically, we show that it is possible to hide any target transcription in virtually any audio file with a minimum of perceptible noise. We analyze the optimal parameter settings, including different phone rates, and allowed deviations from the hearing thresholds. We need less than two minutes on an Intel Core i7 processor to generate an adversarial example for a ten-second audio file. We also demonstrate that it is possible to limit the perturbations to parts of the original audio files, where they are not (or only barely) perceptible by humans. The experiments show that in comparison to other targeted attacks [82], the amount of noise is significantly reduced.

This observation is confirmed during a two-part audibility study, where test listeners transcribe adversarial examples and rate the quality of different settings. The results of the first user study indicate that it is impossible to comprehend the target transcription of adversarial perturbations and only the original transcription is recognized by human listeners. The second part of the listening test is a *Multiple Stimuli with Hidden Reference and Anchor* (MUSHRA) test [99] in order to rate the quality of generated audio from different algorithm setups. The results show that the psychoacoustic model greatly increases the quality and hence the inconspicuousness of the adversarial examples. In summary, we make the following contributions:

- **Psychoacoustic Hiding.** We describe a novel type of adversarial examples against DNN-HMM-based ASR systems based on a psychoacoustically designed attack for hiding transcriptions in arbitrary audio files. Besides the psychoacoustic modeling, the algorithm utilizes an optimal temporal alignment and backpropagation up to the raw audio input.

- **Experimental Evaluation.** We evaluate the proposed attack algorithm in different settings in order to find adversarial perturbations that lead to the best recognition result with the least human-perceptible noise.

- **User Study.** To measure the human perception of adversarial audio samples, we performed a user study. More specifically, human listeners were asked to transcribe what they understood when presented with adversarial examples and to compare their overall audio quality compared to original unmodified audio files.

3.1.1 Attacking Automatic Speech Recognition

In the following, we show how the added audible noise can be limited by applying hearing thresholds during the creation of adversarial examples. As an additional challenge, we need to find the optimal temporal alignment between the audio file and the target text, which gives us the best starting point for the insertion of malicious perturbations. Note that our attack integrates well into the DNN-based speech recognition process: we use the trained ASR system and apply gradient descent, which is already part of the toolchain, to update the input, eventually resulting in adversarial examples.

Threat Model

Throughout the rest of this section, we assume the following adversary model.

First, we assume a white-box attack, where the adversary knows the ASR mechanism of the attacked system and its specific model parameters. Using this knowledge, the attacker generates audio samples containing malicious perturbations before the actual attack takes place, i. e., the attacker exploits the ASR system to obtain an audio file that produces the desired recognition result. Second, we assume the ASR system to be configured in such a way that it gives the best possible recognition rate. In addition, the trained ASR system, including the DNN, remains unchanged over time. Finally, we assume a perfect transmission channel for replaying the manipulated audio samples, hence, we do not take perturbations through audio codecs, compression, hardware, etc. into account by feeding the audio file directly into the recognizer. Note that we only consider targeted attacks, where the target transcription is predefined.

High-Level Overview

There is a variety of commercial and non-commercial ASR systems available. In the research community, *Kaldi* [83] is very popular given that it is an open-source toolkit which provides a wide range of state-of-the-art algorithms for ASR and is also used in commercial tools like Amazon's Alexa (cf. Chapter 4). The tool was developed at

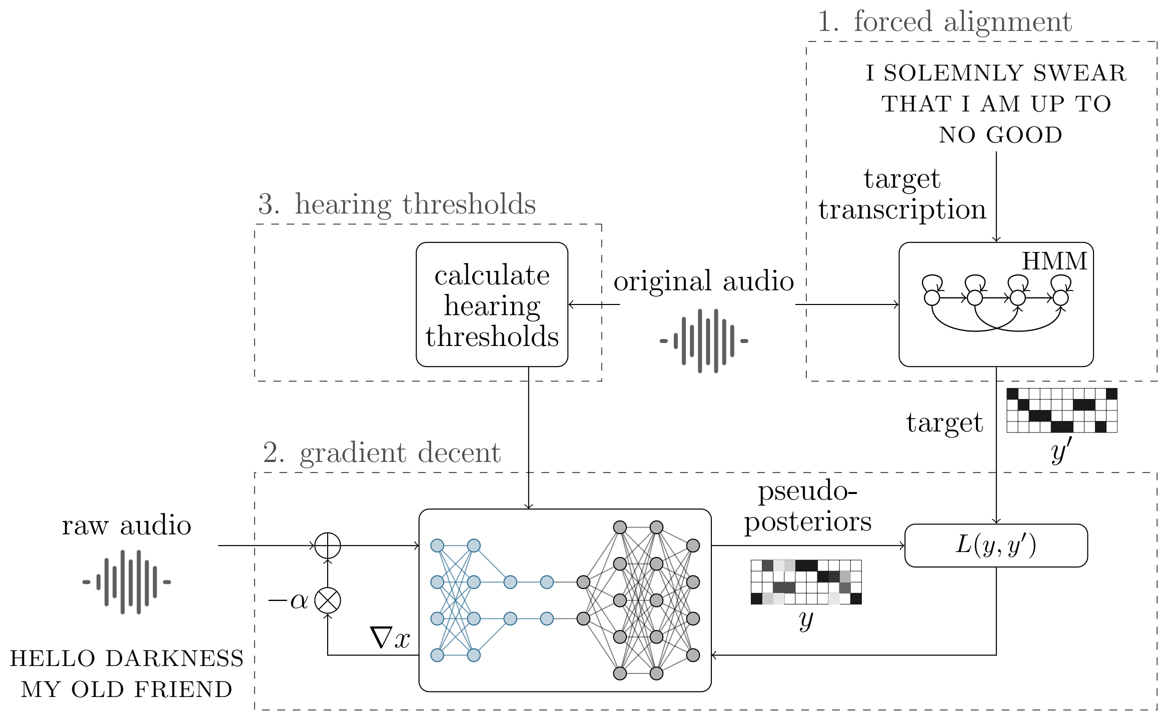


Figure 3.1: The calculation of adversarial examples can be divided into three components: (1) *forced alignment* to find an optimal target for the (2) gradient descent and (3) the integration of the hearing thresholds.

Johns Hopkins University and is written in C++. We performed a partial reverse engineering of the firmware of an Amazon Echo and our results indicate that this device also uses *Kaldi* internally to process audio inputs. Given *Kaldi*'s popularity and its accessibility, this ASR system hence represents an optimal fit for our experiments.

The algorithm for the calculation of adversarial examples can be divided into three parts, which are sketched in Figure 3.1. The main difference between *original audio* and *raw audio* is that the *original audio* does not change during the run-time of the algorithm, but the *raw audio* is updated iteratively in order to result in an adversarial example. Before the gradient descent step, the best possible temporal alignment is calculated via so-called *forced alignment*. The algorithm uses the original audio signal and the target transcription as inputs in order to find the best target pseudo-posteriors. The *forced alignment* is performed just once at the beginning of the algorithm.

With the resulting target posteriogram, we are able to apply gradient descent to manipulate our input signal in such a way that the speech recognition system produces the desired output. The gradient descent step is an iterative process and will, therefore, be repeated until it converges or a fixed upper limit for the number of iterations is reached.

The hearing thresholds are applied during the gradient descent step in order to limit the changes that are perceptible by a human. The hearing thresholds are also calculated once and stored for the gradient descent step.

Forced Alignment

One major problem of attacks against ASR systems is that they require the recognition to pass through a certain sequence of HMM states in such a way that it leads to the target transcription. However, due to the decoding step—which includes a graph search—for a given transcription, many valid pseudo-posterior combinations exist. For example, when the same text is spoken at different speeds, the sequence of the HMM states is correspondingly faster or slower. We can benefit from this fact by using that version of pseudo-posteriors which best fits the given audio signal and the desired target transcription.

We use forced alignment as an algorithm for finding the best possible temporal alignment between the acoustic signal that we manipulate and the transcription that we wish to obtain. This algorithm is provided by the *Kaldi* toolkit. Note that it is not always possible to find an alignment that fits an audio file to any target transcription. In this case, we set the alignment by dividing the audio sample equally into the number of states and set the target according to this division.

Integrating Preprocessing

We integrate the preprocessing step and the DNN step into one joint DNN and a PGD-based approach. This approach is sketched in Figure 3.2. This design choice does not affect the accuracy of the ASR system, but it allows for manipulating the raw audio data by applying backpropagation to the preprocessing steps, directly giving us the optimally adversarial audio signal as result.

With the integration of preprocessing into the DNN, the gradient is calculated via

$$\nabla_x = \frac{\partial \mathcal{L}(\hat{y}, y')}{\partial \mathcal{F}(\chi)} \cdot \frac{\partial \mathcal{F}(\chi)}{\partial \mathcal{F}_P(x)} \cdot \frac{\partial \mathcal{F}_P(x)}{\partial x}, \tag{3.1}$$

where we ignore the iteration index i for simplicity. All preprocessing steps are included in $\chi = \mathcal{F}_P(x)$ and return the input features χ for the DNN. In order to calculate $\frac{\partial \mathcal{F}_P(x)}{\partial x}$, it is necessary to know the derivatives of each of the four preprocessing steps. We will introduce these preprocessing steps and the corresponding derivatives in the following.

Framing and Window Function. In the first step, the raw audio data is divided into T frames of length N and a window function is applied to each frame to avoid too many artifacts in the frequency domain due to spectral leakage. A window function is a simple, element-wise multiplication with fixed values $\omega(n)$

$$x_\omega(t,n) = x(t,n) \cdot \omega(n), \quad n = 1, \dots, N, \tag{3.2}$$

with $t = 1, \dots, T$. Thus, the derivative is just

$$\frac{\partial x_\omega(t,n)}{\partial x(t,n)} = \omega(n). \tag{3.3}$$

Discrete Fourier Transform. For transforming the audio signal into the frequency domain, we apply a DFT to each frame x_ω. This transformation is a common choice for audio features. The DFT is defined as

$$X(t,k) = \sum_{n=0}^{N-1} x_\omega(t,n) e^{-i2\pi \frac{kn}{N}}, \quad k = 1, \dots, N. \tag{3.4}$$

Since the DFT is a weighted sum with fixed coefficients $e^{-i2\pi \frac{kn}{N}}$, the derivative for the backpropagation is simply the corresponding coefficient

$$\frac{\partial X(t,k)}{\partial x_\omega(t,n)} = e^{-i2\pi \frac{kn}{N}}, \quad k, n = 1, \dots, N. \tag{3.5}$$

Magnitude. The output of the DFT is complex valued, but as the phase is not relevant for speech recognition, we just use the magnitude of the spectrum, which is defined as

$$|X(t,k)| = \operatorname{Re}\big(X(t,k)\big)^2 + \operatorname{Im}\big(X(t,k)\big)^2, \tag{3.6}$$

with $\mathrm{Re}(X(t,k))$ and $\mathrm{Im}(X(t,k))$ as the real and imaginary part of $X(t,k)$. For the backpropagation, we need the derivative of the magnitude. In general, this is defined as

$$\frac{\partial|X(t,k)|}{\partial X(t,k)} = 2\cdot\mathrm{Re}(X(t,k)) + 2i\cdot\mathrm{Im}(X(t,k)). \tag{3.7}$$

We consider the real and imaginary parts separately and calculate the derivatives for both cases

$$\nabla_{X(t,k)} = \begin{pmatrix} \frac{\partial|X(t,k)|^2}{\partial\,\mathrm{Re}(X(t,k))} \\ \frac{\partial|X(t,k)|^2}{\partial\,\mathrm{Im}(X(t,k))} \end{pmatrix} = \begin{pmatrix} 2\cdot\mathrm{Re}(X(t,k)) \\ 2\cdot\mathrm{Im}(X(t,k)) \end{pmatrix}. \tag{3.8}$$

This is possible, as real and imaginary parts are stored separately during the calculation of the DNN, which is also sketched in Figure 3.2, where pairs of nodes from layer 2 are connected with only one corresponding node in layer 3. Layer 3 represents the calculation of the magnitude and therefore halves the data size.

Logarithm. The last step is to form the logarithm of the squared magnitude

$$\chi = \log(|X(t,k)|), \tag{3.9}$$

which is the common feature representation in speech recognition systems. It is easy to find its derivative as

$$\frac{\partial\chi}{\partial|X(t,k)|^2} = \frac{1}{|X(t,k)|^2}. \tag{3.10}$$

Hearing Thresholds

For an attack, the psychoacoustic hearing thresholds introduced in Section 1.1.2 are used to limit the changes in the audio signal to time-frequency-ranges, where the added perturbations are not, or barely, perceptible by humans.

Psychoacoustic hearing thresholds allow us to limit audible distortions from all signal manipulations. More specifically, we use the hearing thresholds during the manipulation of the input signal in order to limit audible distortions. For this purpose, we use the original audio signal to calculate the hearing thresholds $\boldsymbol{H}$ as described in Section 1.1.2. We limit the differences $\boldsymbol{P}$ between the original signal spectrum $\boldsymbol{S}$ and

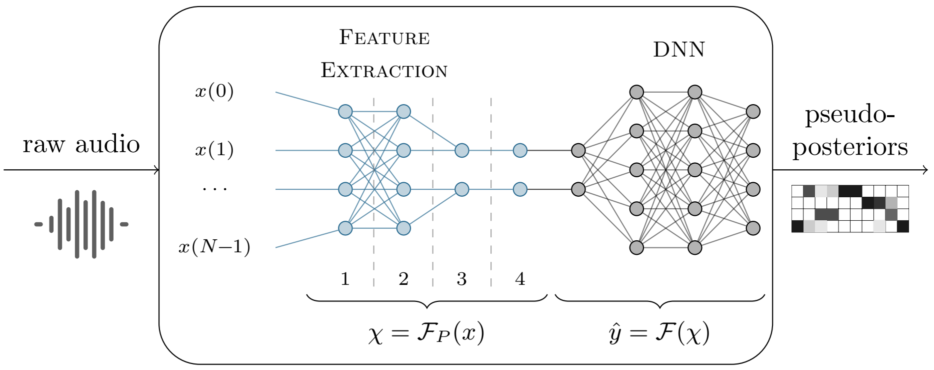


Figure 3.2: For the calculation of adversarial examples, we use an ASR system where the feature extraction is integrated into the DNN. Layers 1–4 represent the separate preprocessing steps. Note that this is only a sketch and that the used DNN contains far more neurons.

the modified signal spectrum $\boldsymbol{M}$ to the threshold of human perception for all times t and frequencies k

$$\boldsymbol{P}(t,k) \leq \boldsymbol{H}(t,k), \quad \forall t,k, \tag{3.11}$$

$$\text{with} \quad \boldsymbol{P}(t,k) = 20 \cdot \log_{10} \frac{|\boldsymbol{S}(t,k) - \boldsymbol{M}(t,k)|}{\max_{t,k}(|\boldsymbol{S}|)}. \tag{3.12}$$

The maximum value of the power spectrum $|\boldsymbol{S}|$ defines the reference value for each utterance, which is necessary to calculate the difference in dB. Examples for $|\boldsymbol{S}|$, $|\boldsymbol{M}|$, $|\boldsymbol{P}|$, and $\boldsymbol{H}$ in dB are shown in Figure 3.3, where the power spectra are plotted for one utterance.

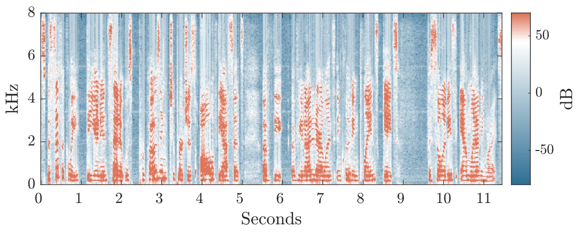
We calculate the amount of distortion that is still acceptable via

$$\boldsymbol{\Phi} = \boldsymbol{H} - \boldsymbol{P}. \tag{3.13}$$

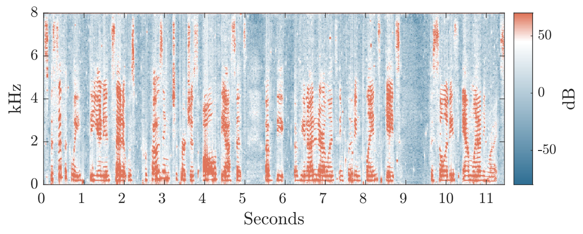
The resulting matrix $\boldsymbol{\Phi}$ contains the difference in dB to the calculated hearing thresholds. In the following step, we use the matrix $\boldsymbol{\Phi}$ to derive scaling factors. First, because the thresholds are too tight to find successful adversarial examples, an additional variable η is added, to allow the algorithm to differ from the hearing thresholds by small amounts

$$\boldsymbol{\Phi}^* = \boldsymbol{\Phi} + \eta. \tag{3.14}$$

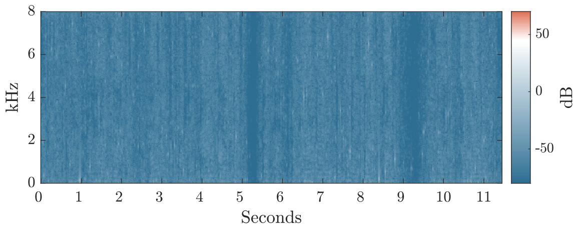
In general, a negative value for $\Phi^*(t,k)$ indicates that we crossed the threshold. As we want to avoid more noise for these time-frequency-bins, we set all $\Phi^*(t,k) < 0$ to



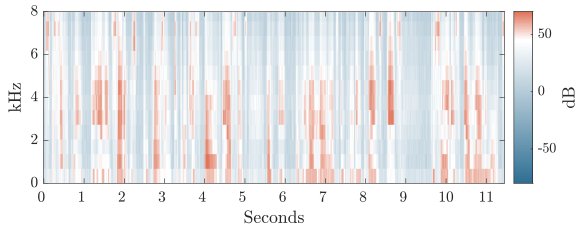
(a) Original audio signal power spectrum $|\boldsymbol{S}|$ with transcription: SPECIFICALLY THE UNION SAID IT WAS PROPOSING TO PURCHASE ALL OF THE ASSETS OF THE OF UNITED AIRLINES INCLUDING PLANES GATES FACILITIES AND LANDING RIGHTS



(b) Adversarial audio signal power spectrum $|\boldsymbol{M}|$ with transcription: DEACTIVATE SECURITY CAMERA AND UNLOCK FRONT DOOR



(c) Power spectrum of the difference between original and adversarial $|\boldsymbol{P}|$.



(d) Hearing thresholds $\boldsymbol{H}$.

Figure 3.3: Original audio sample (3.3a) in comparison to the adversarial audio sample (3.3b). The difference of both signals is shown in Figure 3.3c. Figure 3.3d visualizes the hearing thresholds of the original sample, which are used for the attack algorithm.

zero. We then obtain a time-frequency matrix of scale factors $\hat{\boldsymbol{\Phi}}$ by normalizing $\boldsymbol{\Phi}^*$ to values between zero and one, via

$$\hat{\Phi}(t,k) = \frac{\Phi^*(t,k) - \min_{t,k}(\boldsymbol{\Phi}^*)}{\max_{t,k}(\boldsymbol{\Phi}^*) - \min_{t,k}(\boldsymbol{\Phi}^*)}, \quad \forall t,k. \tag{3.15}$$

The scaling factors are applied during each backpropagation iteration. Using the resulting scaling factors $\hat{\Phi}(t,k)$ typically leads to good results, but especially in the cases where only very small changes are acceptable, including the margin η, this scaling factor alone is not enough to satisfy the hearing thresholds as already one iteration would exceed the maximum allowed changes. Therefore, we use another, fixed scaling factor, which only depends on the hearing thresholds $\boldsymbol{H}$. For this purpose, $\boldsymbol{H}$ is also scaled to values between zero and one, denoted by $\hat{\boldsymbol{H}}$.

Therefore, the gradient $\nabla X(t,k)$ calculated via Equation (3.8) between the DFT and the magnitude step is scaled by both scaling factors

$$\nabla_{X^*(t,k)} = \nabla_{X(t,k)} \cdot \hat{\Phi}(t,k) \cdot \hat{\boldsymbol{H}}(t,k), \quad \forall t,k. \tag{3.16}$$

3.1.2 Experiments and Results

With the help of the following experiments, we verify and assess the proposed attack. We target the ASR system *Kaldi* and use it for our speech recognition experiments. We also compare the influence of the suggested improvements on the performance of the algorithm and assess the effect of significant parameter settings on the success of the adversarial attack.

Experimental Setup

To verify the feasibility of targeted adversarial attacks on state-of-the-art ASR systems, we have used the default settings for the *Wall Street Journal* (WSJ) training recipe of the *Kaldi* toolkit [83]. Only the preprocessing step was adapted for the integration into the DNN. The WSJ data set is well suited for large vocabulary ASR: it is phone-based and contains more than 80 hours of training data, composed of read sentences of the Wall Street Journal recorded under mostly clean conditions. Due to the large dictionary with more than $100,000$ words, this setup is suitable to show the feasibility of targeted adversarial attacks for arbitrary transcriptions.

For the evaluation, we embedded the hidden voice commands (i. e., target transcriptions) in two types of audio data: speech and music. We collected and compared results with and without the application of hearing thresholds, and with and without the use of forced alignment. All computations were performed on a 6-core Intel Core i7-4960X processor.

Metrics

In the following, we describe the metrics that we used to measure recognition accuracy and to assess to which degree the perturbations of the adversarial attacks needed to exceed hearing thresholds in each of our algorithm's variants.

Word Error Rate. As the adversarial examples are primarily designed to fool an ASR system, a natural metric for our success is the accuracy with which the target transcription was actually recognized. For this purpose, we use the Levenshtein distance [100] to calculate the *Word Error Rate* (WER). A dynamic-programming algorithm is employed to count the number of deleted D, inserted I, and substituted S words in comparison to the total number of words N in the sentence, which together allows for determining the word error rate via

$$WER = \frac{D + I + S}{N}. \tag{3.17}$$

When the adversarial example is based on audio samples with speech, it is possible that the original text is transcribed instead of—or in addition to—the target transcription. Therefore, it can happen that many words are inserted, possibly even more words than contained in the target text. This can lead to WERs larger than 100 %, which can also be observed in Table 3.1, and which is not uncommon when testing ASR systems under highly unfavorable conditions.

Difference Measure. To determine the amount of perceptible noise, measures like the SNR are not sufficient, given that they do not represent the subjective, perceptible noise. Hence, we have used $\mathbf{\Phi}$ of Equation (3.13) to obtain a comparable measure of audible noise. For this purpose, we only consider values > 0, as only these are in excess of the hearing thresholds. This may happen when η is set to values larger than zero, or where changes in one frequency bin also affect adjacent bins.

We sum all values $\Phi(t, k) > 0$ for $t = 0, \ldots, T-1$ and $k = 0, \ldots, N-1$ and divide the sum by $T \cdot N$ for normalization. This value is denoted by Ψ. It constitutes our measure of the degree of perceptibility of noise. Note that the measure does not provide a psychoacoustically validated measure, but allows us to compare the results.

Improving the Attack

As a baseline, we used a simplified version of the algorithm, forgoing both the hearing thresholds and the forced alignment stage. In the second scenario, we included the proposed hearing thresholds. This minimizes the amount of added noise but also decreases the chance of a valid adversarial example. In the final scenario, we added the forced alignment step, which results in the full version of the suggested algorithm, with a clearly improved WER.

For the experiments, a subset of 70 utterances for 10 different speakers from one of the WSJ test sets was used.

Backpropagation. First, the adversarial attack algorithm was applied without the hearing thresholds or the forced alignment. Hence, for the alignment, the audio sample was divided equally into the states of the target transcription. We used 500 iterations of backpropagation. This gives robust results and requires a reasonable time for computation. We chose a learning rate of 0.05, as it gave the best results during preliminary experiments. This learning rate was also used for all following experiments.

In the baseline test, we achieved a WER of 1.43 %, but with perceptible noise. This can be seen in the average Ψ, which was 11.62 dB for this scenario. This value indicates that the difference is clearly perceptible. However, the small WER shows that targeted attacks on ASR systems are possible and that our approach of backpropagation into the time domain can very reliably produce valid adversarial audio samples.

Hearing Thresholds. Since the main goal of the algorithm is the reduction of the perceptible noise, we included the hearing thresholds as described in Section 3.1.1. For this setting, we ran the same test as before.

In this case, the WER increases to 64.29 %, but it is still possible to create valid adversarial samples. On the positive side, the perceptible noise is clearly reduced. This is also indicated by the much smaller value of Ψ of only 7.04 dB.

For this experiment, we chose $\eta = 20$, which is the smallest value for η with a WER far below 100 % in Table 3.1 and has therefore shown to be a good trade-off.

Forced Alignment. To evaluate the complete system, we replaced the equal partitioning by forced alignment. Again, the same test set and the same settings as in the previous scenarios were used. Figure 3.4 shows a comparison of the algorithm's performance with and without forced alignment for different values of η, shown on the x-axis. The parameter η is defined in Equation (3.14) and describes the amount the result can exceed the thresholds in dB. In all relevant cases, the WER and Ψ show better results with forced alignment. The only exception is the one case of $\eta = 0$, where the WER is very high in all scenarios.

In the specific case of $\eta = 20$, set as in Section 3.1.2, a WER of 36.43 % was achieved. This result shows the significant advantage of the forced alignment step.

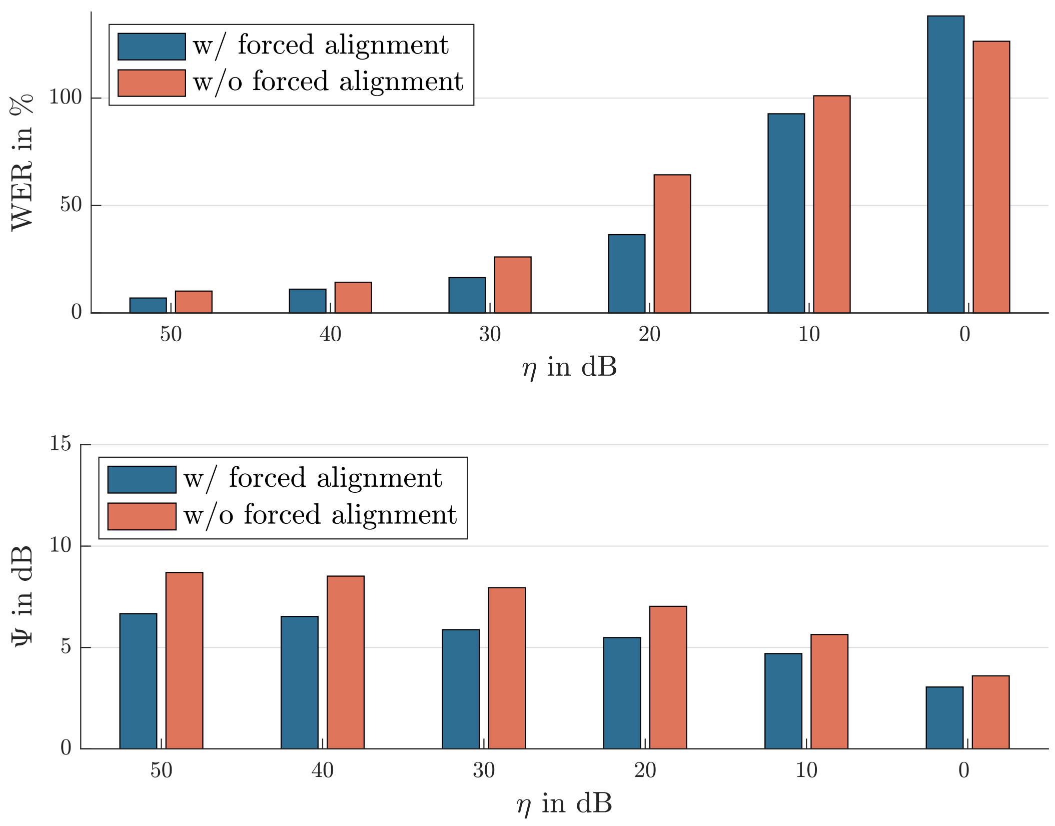


Figure 3.4: Comparison of the algorithm with and without forced alignment, evaluated for different values of η.

At the same time, the noise was again noticeably reduced, with $\Psi = 5.49\,\text{dB}$. This demonstrates that the best temporal alignment increases the success rate in the sense of the WER, while at the same time reducing the amount of noise—a rare win-win situation in the highly optimized domain of ASR. In Figure 3.3, an example of an original spectrum of an audio sample is compared with the corresponding adversarial audio sample. One can see the negligible differences between both signals. The added noise is plotted in Figure 3.3c. Figure 3.3d depicts the hearing thresholds of the same utterance, which were used in the attack algorithm.

Evaluation

In the next steps, the optimal settings are evaluated, considering the success rate, the amount of noise, and the time required to generate valid adversarial examples.

Evaluation of Hearing Thresholds. In Table 3.1, the results for speech and music samples are shown for 500 and for 1000 iterations of backpropagation, respectively. The value in the first row shows the setting of η. For comparison, the case without the use of hearing thresholds is shown in the column 'None.' We applied all combinations of settings on a test set of speech containing 72 samples and a test set of music containing 70 samples. The test set of speech was the same as for the previous evaluations and the target text was the same for all audio samples.

The results in Table 3.1 show the dependence on the number of iterations and on η. The higher the number of iterations and the higher η, the lower the WER becomes. The experiments with music show some exceptions to this rule, as a higher number of iterations slightly increases the WER in some cases. However, this is only true where no thresholds were employed or for $\eta = 50$.

As is to be expected, the best WER results were achieved when the hearing thresholds were not applied. However, the results with applied thresholds show that it is indeed feasible to find a valid adversarial example very reliably even while minimizing human perceptibility. Even for the last column, where the WER increases to more than 100 %, it was still possible to create valid adversarial examples, as we will show in the following evaluations.

In Table 3.2, the corresponding values for the mean perceptibility Ψ are shown. In contrast to the WER, the value Ψ decreases with η, which shows the general success of the thresholds, as smaller values indicate a smaller perceptibility. Especially when no thresholds are used, Ψ is significantly higher than in all other cases. The evaluation of music samples shows smaller values of Ψ in all cases, which indicates that it is much easier to conceal adversarial examples in music. This was also confirmed by the listening tests (cf. Section 3.1.3).

Phone Rate Evaluation. For the attack, timing changes are not relevant as long as the target text is recognized correctly. Therefore, we have tested different combinations of audio input and target text, measuring the number of phones that we could hide per second of audio, to find an optimum phone rate for our ASR system. For this purpose, different target utterances were used to create adversarial examples from audio samples of different lengths. The results are plotted in Figure 3.5. For the evaluations, 500 iterations and $\eta = 20$ were used. Each point of the graph was com-

Table 3.1: WER in % for different values for η in the range of 0 dB to 50 dB, comparing speech and music as audio inputs.

	Iter.	**None**	**50 dB**	**40 dB**	**30 dB**	**20 dB**	**10 dB**	**0 dB**
Speech	500	2.14	6.96	11.07	16.43	36.43	92.69	138.21
	1000	1.79	3.93	5.00	7.50	22.32	76.96	128.93
Music	500	1.04	8.16	13.89	22.74	31.77	60.07	77.08
	1000	1.22	10.07	9.55	15.10	31.60	56.42	77.60

Table 3.2: The perceptibility Ψ over all samples in the test sets in dB.

	Iter.	**None**	**50 dB**	**40 dB**	**30 dB**	**20 dB**	**10 dB**	**0 dB**
Speech	500	10.11	6.67	6.53	5.88	5.49	4.70	3.05
	1000	10.80	7.42	7.54	6.85	6.46	5.72	3.61
Music	500	4.92	3.92	3.56	3.53	3.39	2.98	2.02
	1000	5.03	3.91	3.68	3.40	3.49	3.20	2.30

puted based on 200 adversarial examples with changing targets and different audio samples, all of them speech.

Figure 3.5 shows that the WER increases clearly with an increasing phone rate. We observe a minimum for 4 phones per second, which does not change significantly at a smaller rate. As the time to calculate an adversarial sample increases with the length of the audio sample, 4 phones per second is a reasonable choice.

Calculation Time. The algorithm is easy to parallelize and for a ten-second audio file, it takes less than two minutes to calculate the adversarial perturbations with 500 backpropagation steps on a 6-core (12 threads) Intel Core i7-4960X processor.

Comparison with *CommanderSong*

We compare the amount of noise with *CommanderSong* [82], as their approach is also able to create targeted attacks using *Kaldi* and therefore the same DNN-HMM-based ASR system. Additionally, it is the only recent work, which reported the SNR

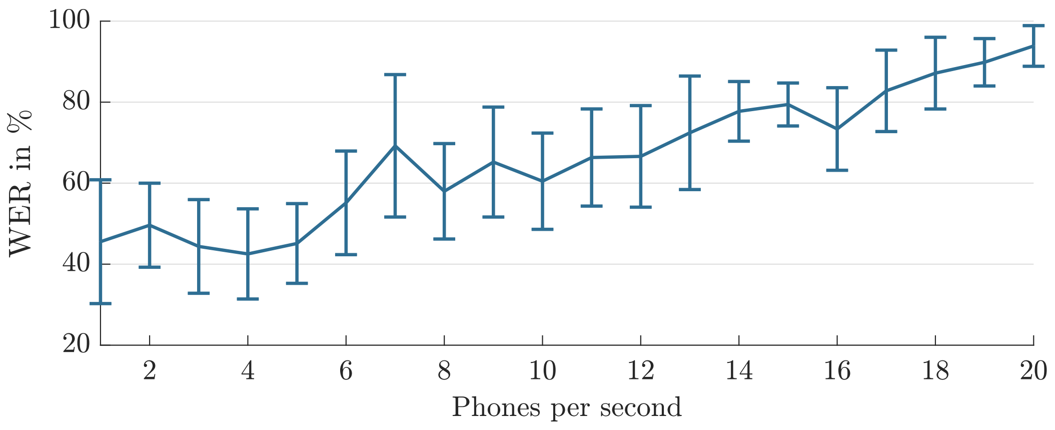


Figure 3.5: Accuracy for different phone rates. To create the examples, 500 iterations of backpropagation and $\eta = 20$ are used. The vertical lines represent the standard deviation.

Table 3.3: Comparison of SNR with *CommanderSong* [82], best result shown in bold.

	None	**40 dB**	**20 dB**	**0 dB**	***CommanderSong* [82]**
SNR	15.88	17.93	**21.76**	19.38	15.32

of their results. The SNR measures the amount of noise σ, added to the original signal x, and is computed via

$$\mathrm{SNR(dB)} = 10 \cdot \log_{10} \frac{P_x}{P_\sigma}, \tag{3.18}$$

where P_x and P_σ are the energies of the original signal and the noise. This means, the *higher* the SNR, the *less* noise was added.

Table 3.3 shows the SNR for successful adversarial samples, where no hearing thresholds are used (None) and for different values of η (40 dB, 20 dB, and 0 dB) in comparison to *CommanderSong*. Note, that the SNR does not measure the perceptible noise and therefore, the resulting values are not always consistent with the previously reported Ψ. Nevertheless, the results show, that in all cases, even if no hearing thresholds are used, we achieve higher SNRs, meaning, less noise was added to create a successful adversarial example.

3.1.3 User Study

We have evaluated the human perception of our audio manipulations through a two-part user study. In the first part, a transcription test, we verified that it is impossible to understand the voice command hidden in an audio sample. The second part, a MUSHRA test, provides an estimate of the perceived audio quality of adversarial examples, where we tested different parameter setups of the hiding process.

Transcription Test

While the original text of a speech audio sample should still be understandable by human listeners, we aim for a result where the hidden command cannot be transcribed or even identified as speech. Therefore, we performed a *transcription test*, in which test listeners were asked to transcribe the utterances of original and adversarial audio samples.

Study Setup. Each test listener was asked to transcribe 21 audio samples. The utterances were the same for everyone, but with randomly chosen conditions: 9 original utterances, 3 adversarial examples with $\eta = 0$, $\eta = 20$, and $\eta = 40$, respectively and 3 difference signals of the original and the adversarial example, one for each value of η. For the adversarial utterances, we made sure that all samples were valid, such that the target text was successfully hidden within the original utterance. We only included adversarial examples that required ≤ 500 iterations.

We conducted the tests in a soundproofed chamber and asked the participants to listen to the samples via headphones. The task was to type all words of the audio sample into a blank text field without any provision of auto-completion, grammar, or spell checking. Participants were allowed to repeat each audio sample as often as needed and enter whatever they understood. In a post-processing phase, we performed manual corrections on minor errors in the documented answers to address typos, misspelled proper nouns, and numbers. After revising the answers in the post-processing step, we calculated the WER using the same algorithms as introduced in Section 3.1.2.

Results. For the evaluation, we have collected data from 22 listeners during an internal study at our university. None of the listeners were native speakers, but all

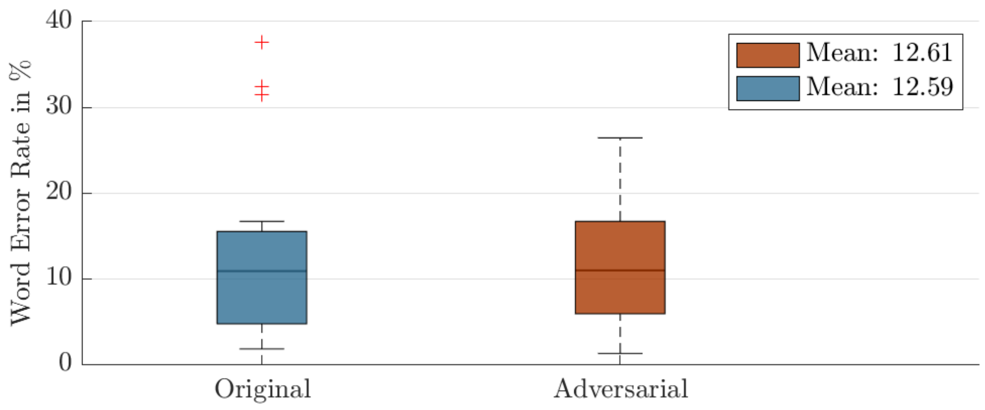


Figure 3.6: WER for all 21 original and adversarial utterances over all test listeners.

had sufficient English skills to understand and transcribe English utterances. As we wanted to compare the WER of the original utterances with the adversarial ones, the average WER of 12.52 % overall test listeners was sufficient. This number seems high, but the texts of the WSJ are quite challenging. For the evaluation, we ignored all cases where only the difference of the original and adversarial sample was played. For all of these cases, none of the test listeners was able to recognize any kind of speech and therefore no text was transcribed.

For the original utterances and the adversarial utterances, an average WER of 12.59 % and 12.61 % was calculated. The marginal difference shows that the change in the audio does not influence the intelligibility of the utterances. Additionally, we have tested the distributions of the transcription error rates of the original utterances and the adversarial utterances with a two-sided t-test to verify whether both distributions have the same mean and variance. The test shows no difference for the distributions of original and adversarial utterances at a significance level of 1 % .

In the second step, we have also compared the text from the test listeners with the text which was hidden in the adversarial examples. In this evaluation, we have measured a WER far above 100 %, which shows that the hidden text is not intelligible. Also, the only correct words here were also present in the original text, and in all cases these were frequent, short words like *is*, *in*, or *the*. The results are shown in Figure 3.6.

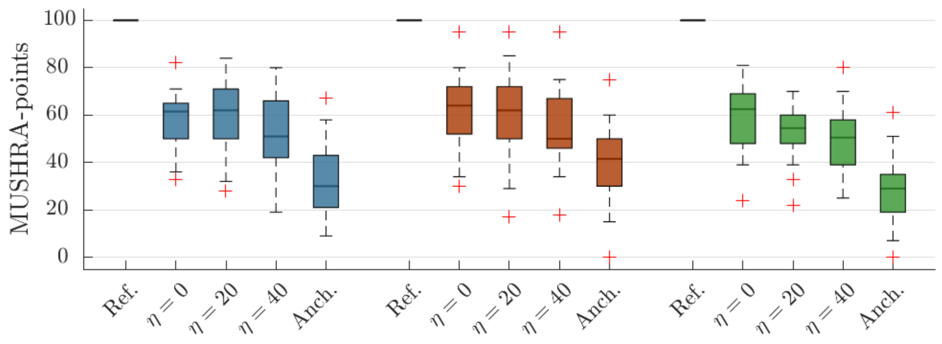
MUSHRA Test

In the second part of the study, we have conducted a Multiple Stimuli with Hidden Reference and Anchor (MUSHRA) test, which is commonly used to rate the quality of audio signals [99].

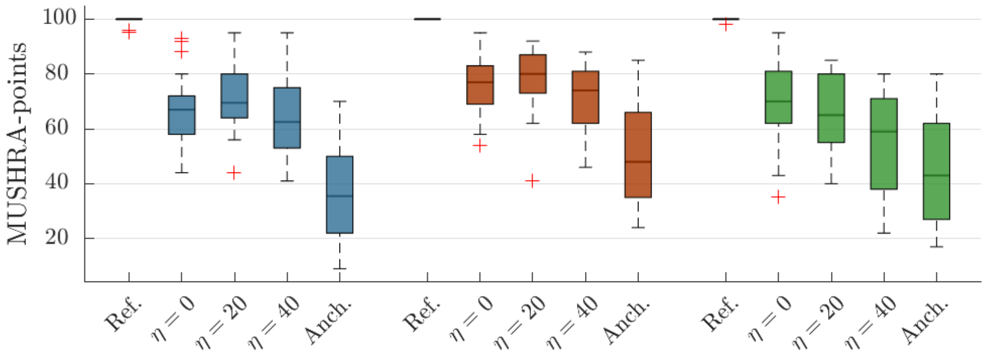
Study Setup. The participants were asked to rate the quality of a set of audio signals with respect to the original signal. The set contains different versions of the original audio signal under varying conditions. As the acronym shows, the set includes a *hidden reference* and an *anchor*. The former is the sample with the best and the latter the one with the worst quality. In our case, we have used the original audio signal as the *hidden reference* and the adversarial example, that was derived without considering the hearing thresholds, as *anchor*. Both the *hidden reference* and the *anchor* are used to exclude participants, who were not able to identify either the *hidden reference* or the *anchor*. As a general rule, the results of participants who rate the *hidden reference* with less than 90 MUSRHA-points more than 15 % of the time are not considered. Similarly, all results of listeners who rate the *anchor* with more than 90 MUSRHA-points more than 15 % of the time are excluded. We used the *webMUSHRA* implementation, which is available online and was developed by AudioLabs [101].

We have prepared a MUSHRA test with nine different audio samples, three for speech, three for music, and three for recorded twittering birds. For all these cases, we have created adversarial examples for $\eta = 0$, $\eta = 20$, $\eta = 40$, and without hearing thresholds. Within one set, the target text remained the same for all conditions, and in all cases, all adversarial examples were successful with ≤ 500 iterations. The participants were asked to rate all audio signals in the set on a scale between 0 and 100 (0–20: Bad, 21–40: Poor, 41–60: Fair, 61–80: Good, 81–100: Excellent). Again, the listening test was conducted in a soundproofed chamber and via headphones.

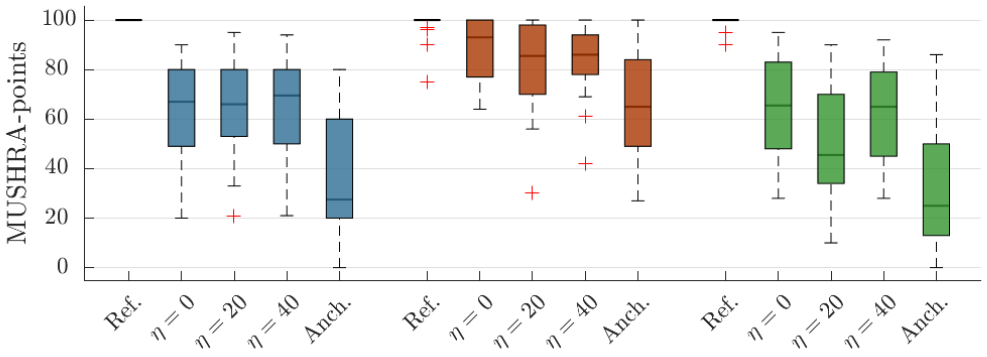
Results. We have collected data from 30 test listeners, 3 of whom were discarded due to the MUSHRA exclusion criteria. The results of the remaining test listeners are shown in Figure 3.7 for all nine MUSHRA tests. In almost all cases, the reference is rated with 100 MUSHRA-points. Also, the anchors are rated with the lowest values in all cases.



(a) Speech



(b) Music



(c) Birds

Figure 3.7: Ratings of all test listeners in the MUSHRA test. We tested three audio samples for speech, music, and twittering birds. The left box plot of all nine cases shows the rating of the original signal and therefore shows very high values. The anchor is an adversarial example of the audio signal that had been created without considering hearing thresholds.

We tested the distributions of the anchor and the other adversarial utterances in one-sided t-tests. For this, we used all values for one condition so, overall for nine MUSHRA tests. The tests, at a significance level of 1 %, show that in all cases, the anchor distribution without the use of hearing thresholds has a significantly lower average rating than those adversarial examples where the hearing thresholds are used. Hence, there is a clearly perceptible difference between adversarial examples with hearing thresholds and adversarial examples without hearing thresholds.

During the test, the original signal was normally rated higher than the adversarial examples. However, it has to be considered that the test listeners directly compared the original signal with the adversarial ones. In an attack scenario, this would not be the case, as the original audio signal is normally unknown to the listeners. Despite the direct comparison, there is one MUSRHA test, where the adversarial examples with hearing thresholds are very frequently rated with a similar value as the reference and more than 80 MUSHRA-points. This is the case for the second test with twittering birds, which shows that there is a barely perceptible difference between the adversarial samples and the original audio signal.

Additionally, we observed that there is no clear preference for a specific value of η. The samples with $\eta = 0$ received a slightly higher average rating in comparison to $\eta = 20$ and $\eta = 40$, but there is only a significant difference for the distributions of $\eta = 0$ and $\eta = 40$.

Attack Parameters

We have shown that it is possible to successfully attack state-of-the-art DNN-HMM ASR systems with targeted adversarial perturbations, which are barely to distinguish from original audio samples. Based on different experiments, we demonstrated means of deriving the best setup for the proposed algorithm for the creation of adversarial examples. The choice of the parameters highly affects the amount of perceptible noise. The evaluation has shown that a higher number of iterations increases the success rate, but simultaneously the amount of noise. However, for iterations < 500, the success rate is already very high and therefore, 500 should not be exceeded. Additionally, by this choice, the required calculation time is reduced as well.

If the success rate needs to be raised, the increase of λ had a higher effect. Although the participants in the MUSHRA test did prefer smaller values for λ, there was no significant difference if λ was only increased by 20 dB. Additionally, the phone rate should be set to an optimum value as this highly affects the success of the attack.

Besides improving the success of the attack, the choice of the original audio sample greatly influences the quality of the adversarial example. There might be use cases, where the original audio sample is fixed, but in general, the choice of the original sample is free. We recommend using music or other dinconspicuous audio samples, like bird twittering, which do not contain speech, as speech has to be obfuscated, typically leading to larger required adversarial perturbations.

The process can be parallelized and is relatively fast in comparison to other attacks proposed in the past, as we have integrated the preprocessing into the backpropagation. Therefore, we recommend to use different promising setups and to choose that one which sounds the most inconspicuous while giving the required success rate.

We assume that training the ASR-DNN with MP3-encoded audio files will only move the vulnerability into the perceptible region of the audio files, but will not circumvent blind spots of DNNs completely.

3.2 Robust Over-the-Air Adversarial Examples

The practical implications and real-world impact of the demonstrated attack are unclearup to this point as the audio examples are fed into the ASR system *directly*, hence ignoring all side effects (e. g., echo or reverberation) of a real-world environment, where the sound is transmitted from a loudspeaker to the input microphone of the recognition engine. Other works demonstrated adversarial examples that can be played over-the-air [78, 102, 103, 104], but these proof-of-concept attacks are either tailored to a single, static room setup or are hard to reproduce systematically with a proven success rate in a different environment like the attack sketched in *CommanderSong* [82]. Also, in cases where over-the-air adversarial examples have been used in black-box settings, the target transcription is easy to perceive by human listeners, once the intended attack is known [78, 102].

We argue that adversarial examples for ASR systems can only be considered a real threat if the targeted recognition is produced even when the signal is played over the air. Compared to previous attacks, where the manipulated speech signal is fed directly into the ASR system, over-the-air attacks are more challenging, as the transmission over the air significantly alters the signal.

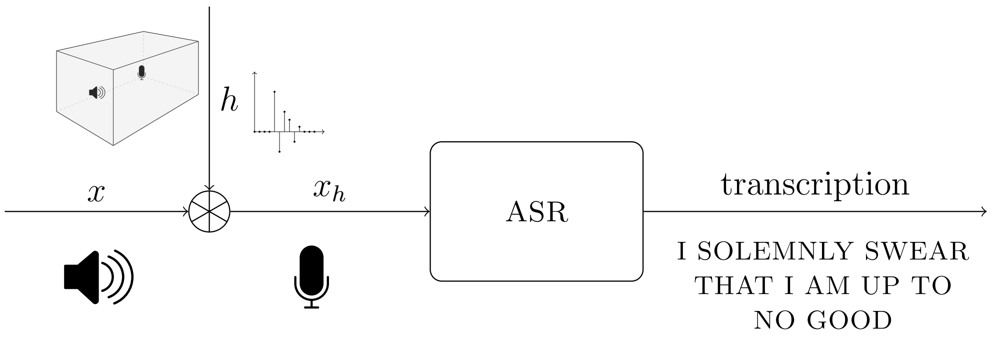


Figure 3.8: For an over-the-air attack against ASR systems, the attack needs to remain viable after the transmission over the air. This transmission can be modeled as a convolution of the original audio signal x with the RIR h.

Our key insight that we study in this chapter is that this transmission can be modeled as a convolution of the original audio signal with the room-dependent RIR, which describes the transmission of an acoustic signal from the loudspeaker to the microphone (see Figure 3.8 for an illustration) where the RIR depends on various factors [105]. In practice, it is nearly impossible to estimate an exact RIR without having access to the actual room. Therefore, robust adversarial examples need to take a range of possible RIRs into account to increase the success rate. Nevertheless, we show that for a successful attack, it is not necessary to acquire precise knowledge about the attack setup. Instead, a generic adversarial example is enough, provided that it is optimized for a large range of possible room setups.

Our approach is inspired by Athalye et al.'s seminal work: a real-world 3D-printed turtle, which is recognized as a rifle from almost every point of view due to an adversarial perturbation [106]. The algorithm for creating this 3D object not only minimizes the distortion for one image, but for *all* possible projections of a 3D object into a 2D image. We borrow the idea of this approach—also referred to as *Expectation over Transformation* (EoT)—and transfer it to the audio domain, replacing the projections by convolutions with RIRs, thereby hardening the audio adversarial example against the transmission through varying rooms.

We introduce the first method to compute generic and robust over-the-air adversarial examples against hybrid ASR systems. We archive this by utilizing an RIR generator to sample from different room setups. We implement a full, end-to-end attack that works in both cases, with and without psychoacoustic hiding. In either case, we can produce successful robust adversarial examples. With our generic approach, it is possible to induce an arbitrary target transcription in any kind of audio without physical access to the room where the attack takes place.

More specifically, for the simulation, the convolution with the sampled RIR is added to the ASR's DNN as an additional layer, which enables us to update the original audio signal directly under the constraints given by the simulated RIR. For this purpose, the RIRs are drawn out of a distribution of room setups to simulate the over-the-air attack. The algorithm is repeated until the target transcription is recognized or a maximum number of iterations is reached. Using this approach, the adversarial examples are hardened to remain robust in real over-the-air attacks across various room setups. We also show an improvement that is based on psychoacoustic hiding [10].

We have implemented the proposed algorithm to attack the DNN-HMM ASR system *Kaldi* [83] under varying room conditions. During our experiments, we found that the generic approach, even though it is also the much more powerful attack, is even more robust than the adapted version, which uses measured RIRs. This proves that the used room simulation is an adequate and broadly applicable approach to simulate non-specified environments and that for a successful attack, no prior knowledge about the attack setup is required. Furthermore, we demonstrate that the same adversarial examples are transferable to different rooms. They work without line-of-sight, distances in the range of meters, and even if no direct sound but only a reflection is recorded by the microphone.

In summary, in this chapter, we make the following three key contributions:

- **Robust Over-the-Air Attack.** We propose a generic approach to generate robust over-the-air adversarial examples for DNN-HMM-based ASR systems. The attack uses a DNN convolution layer to simulate the effect of RIRs, which allows us to alter the raw audio signal.

- **Psychoacoustics.** We show that the attack can be combined with psychoacoustic methods for reducing the perceived distortions.

- **Performance Analysis.** We evaluate the accuracy of the adversarial attack and analyze the amount of added perturbation. We propose an adaptive and a generic version of the attack and investigate the influence of increasing reverberation time, increasing microphone-to-speaker distances, different rooms, and investigate if no direct line-of-sight between speaker and microphone exists.

3.2.1 Over-the-Air Adversarial Examples

Our goal is to compute robust audio adversarial examples, which still work after transmission through the air. For this purpose, we simulate different RIRs and employ an iterative algorithm to compute adversarial examples robust against signal modifications that are incurred during playback in a room.

Threat Model

Throughout the rest of this section, we consider the following threat model, similar to prior work in this area. We assume a white-box attack, where the adversary knows the internals of the ASR system, including all its model parameters. This requirement is in line with prior work on this topic [76, 82]. Using this knowledge, the attacker generates malicious audio samples offline before the actual attack takes place, i. e., the attacker exploits the ASR system to create an audio file that produces the desired recognition result, which is then played via a loudspeaker. Additionally, we assume that the trained ASR system, including the DNN, remains unchanged over time. Finally, we assume that the adversarial examples are played over the air. Note that we only consider targeted attacks, where the target transcription is predefined (i. e., the adversary chooses the target transcription). Finally, we assume a threat model where a potential attacker can run an extensive search. Specifically, the attacker is able to calculate a batch of potential adversarial examples and select those examples that are especially robust.

Room Impulse Response

When an audio signal is transmitted through a room, as visualized in Figures 3.8 and 3.9, the recorded signal can be approximated by convolving the room's impulse response h with the original audio signal x as

$$x_h = x * h. \tag{3.19}$$

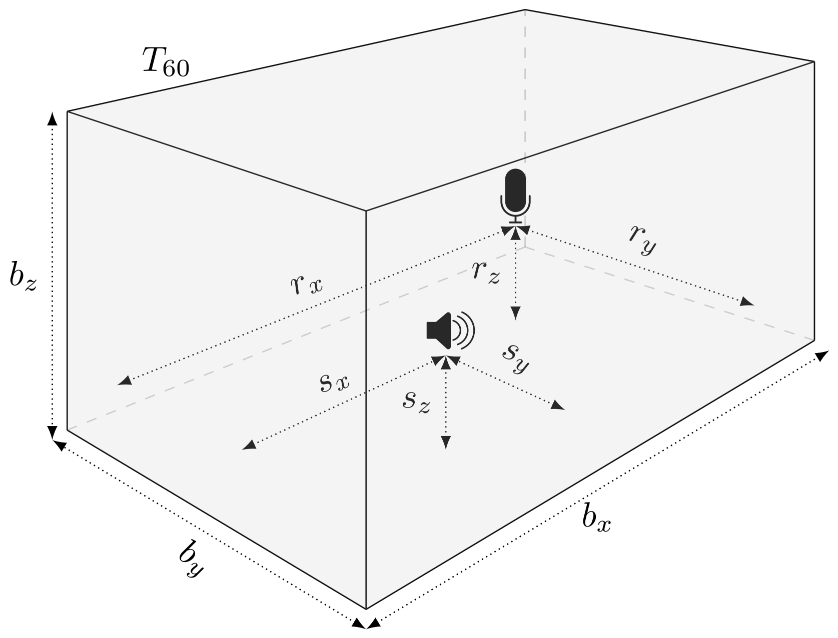


Figure 3.9: For the room simulation model, we assume a probability distribution over all possible rooms by defining relevant simulation parameters like the room geometry, the reverberation time T_{60}, and positions of source and receiver as random variables. To optimize our over-the-air adversarial examples, we sample from this distribution to get a variety of possible RIRs.

Here, the convolution operator $*$ is a shorthand notation for the multi-path transmission model

$$x_h(n) = \sum_{m=n-M+1}^{n} x(m) \cdot h(n-m) \quad \text{with} \quad n = 0, \dots, M-1, \tag{3.20}$$

where M is the length of the audio signal, M the length of the RIR h, and all $x(n)$ with $n < 0$ are assumed to be zero.

In general, the RIR h depends on the size of the room, the positions of the source and the receiver, and other room characteristics such as the sound reflection properties of the walls, any furniture, people, or other contents of the room. Hence, the audio signal received by the ASR system is never identical to the original audio, and an exact RIR is practically impossible to predict. We describe a possible solution for a sufficient approximation in Section 3.2.1.

Room Impulse Response Simulator

To simulate RIRs, we use the *AudioLabs* implementation based on the image method from Allen and Berkley [105]. The simulator takes as input the room dimensions, the reverberation time T_{60}, and the position of source and receiver and approximates the corresponding RIR for the given parameters.

For our attack, we model cuboid-shaped rooms, which can be described by their length b_x, width b_y, and height b_z, defined as $\mathbf{b} = [b_x, b_y, b_z]$. In addition to this, we model the three-dimensional source position $\mathbf{s} = [s_x, s_y, s_z]$, receiver position $\mathbf{r} = [r_x, r_y, r_z]$, and the reverberation time T_{60}, which is a standard measure for the audio decay time, defined as the time it takes for the sound pressure level to reduce by 60 dB. This results in ten freely selectable parameters. All parameters are also sketched in Figure 3.9. Even though this might seem like an overly simple model, we show that the computed adversarial examples are indeed robust for real rooms that are more complex.

In order to sample random RIRs, we interpret these ten parameters to be random variables. We draw each value from a uniform distribution between a minimum and a maximum allowed value. For the room size and for T_{60}, the minimum and the maximum values can be chosen arbitrarily and are thus selected first. After those parameters are drawn, the ranges for source and receiver positions are drawn to guarantee that the source and the receiver are located inside the room.

To simplify the notation, we use the 10-dimensional parameter vector ξ in the following to describe all of these parameters. The RIR h can be considered as a sample of the distribution $\mathcal{H}_\xi$. An example of a simulated RIR in the time and the frequency domain is shown in Figure 3.10.

Robust Audio Adversarial Examples

Unlike earlier approaches that feed adversarial examples directly into the ASR system [76], we explicitly include characteristics of the room, in the form of RIRs, in the optimization problem. This hardens the adversarial examples to remain functional in an over-the-air attack.

For the attack, we therefore use the optimization criterion

$$\max_{\delta} \mathbb{E}_{h \sim \mathcal{H}_\xi}[P(y'|x * h)]. \tag{3.21}$$

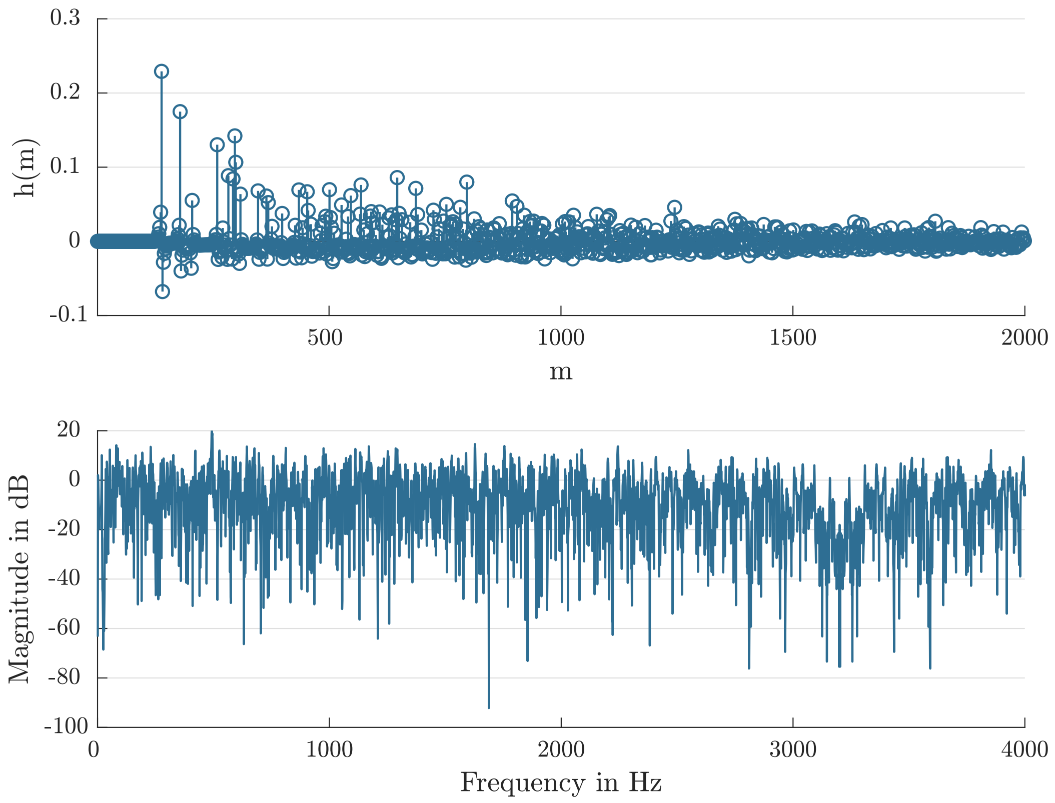


Figure 3.10: Simulated RIR for $\mathbf{b} = [8\,\text{m}, 7\,\text{m}, 2.8\,\text{m}]$, $\mathbf{s} = [3.9\,\text{m}, 3.4\,\text{m}, 1.2\,\text{m}]$ and $\mathbf{r} = [1.4\,\text{m}, 1.8\,\text{m}, 1.2\,\text{m}]$, and $T_{60} = 0.4$ in the time domain (top) and the frequency domain (bottom).

This approach is derived from the *Expectation Over Transformation* (EOT) approach in the visual domain, where it is used to consider different two- and three-dimensional transformations, which leads to robust real-world adversarial examples [106]. In our case, instead of visual transformations, we use the convolution with RIRs, drawn from $\mathcal{H}_\xi$, to maximize the expectation over varying RIRs, as shown in Equation (3.21).

For the implementation, we use a DNN that already has been augmented to include the feature extraction and prepend an additional layer to the DNN. This layer simulates the convolution of the input audio file with the RIR h to model the transmission through the room. Integrating this convolution as an additional layer allows us to apply gradient descent directly to the raw audio signal. A schematic overview of the proposed DNN is given in Figure 3.11. The first part (“Convolution”) describes the convolution with the RIR h. Note that the RIR simulation layer is only used for the calculation of adversarial examples and removed during testing, as the actual RIR will then act during the transmission over the air. The center and right part ("Feature extraction" and "DNN") show the feature extraction and the acoustic model DNN, which is used to obtain the pseudo-posteriors for the decoding stage.

The inclusion of the convolution as a layer in the DNN requires it to be differentiable. Using (3.20), the derivative can be written as

$$\frac{\partial x_h(n)}{\partial x(m)} = h(n-m) \quad \forall n, m. \tag{3.22}$$

or written as the Jacobian Matrix

$$\frac{\partial x_h}{\partial x} = \begin{pmatrix} \frac{\partial x_h(0)}{\partial x(1-M)} & \frac{\partial x_h(1)}{\partial x(2-M)} & \cdots & \frac{\partial x_h(M-1)}{\partial x(M-M)} \\ \frac{\partial x_h(0)}{\partial x(2-M)} & \frac{\partial x_h(1)}{\partial x(3-M)} & \cdots & \frac{\partial x_h(M-1)}{\partial x(M-(M-1))} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_h(0)}{\partial x(0)} & \frac{\partial x_h(1)}{\partial x(1)} & \cdots & \frac{\partial x_h(M-1)}{\partial x(M-1)} \end{pmatrix} \tag{3.23}$$

$$= \begin{pmatrix} h(M-1) & h(M-1) & \dots & h(M-1) \\ h(M-2) & h(M-2) & \dots & h(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ h(0) & h(0) & \dots & h(0) \end{pmatrix}. \tag{3.24}$$

This can be included in the calculation of the gradient ∇x as

$$\nabla_x = \frac{\partial \mathcal{L}(y, y')}{\partial \mathcal{F}(\chi)} \cdot \frac{\partial \mathcal{F}(\chi)}{\partial \mathcal{F}_P(x_h)} \cdot \frac{\partial \mathcal{F}_P(x_h)}{\partial x_h} \cdot \frac{\partial x_h}{\partial x}, \tag{3.25}$$

as an extension of Equation (3.1).

Over-the-Air Adversarial Examples

To assess the robustness of the hardened over-the-air adversarial attack, the adversarial examples x' have to be played back via a loudspeaker, and the recorded audio signals are used to determine the accuracy. For the calculation, we implemented the optimization as defined in Equation (3.21), by sampling a new RIR h after every set of Q gradient descent iterations. This simulates different rooms and recording conditions. Therefore, the generated adversarial example depends on the distribution $\mathcal{H}_\xi$, from which the RIR h is drawn. After each gradient descent step, the audio signal x' is updated via the calculated gradient ∇x at the learning rate α.

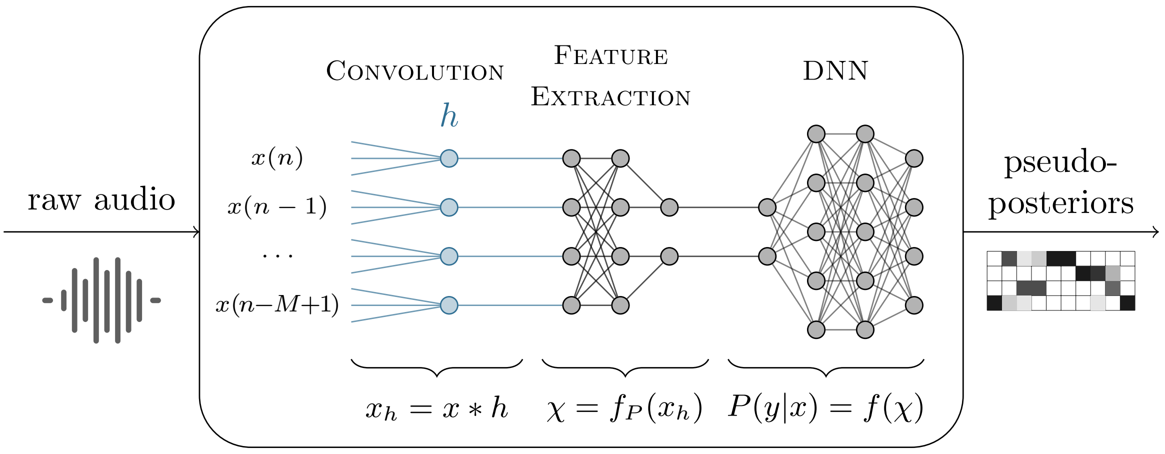


Figure 3.11: To simulate any RIR and to update the time domain audio signal directly, the RIR is integrated as an additional layer into the DNN.

The total maximum number of iterations is limited to at most G iterations. However, if a successful robust adversarial example is created before the maximum number of iterations is reached, the algorithm does not need to continue. To efficiently calculate adversarial examples, we use an RIR h_{test} to simulate the over-the-air scenario during the calculation to verify whether the example has already achieved over-the-air robustness. This RIR is only used for verification and can, for example, be drawn out of $\mathcal{H}_\xi$ once at the beginning of the algorithm.

The entire approach is summarized with Algorithm 1. As can be seen, the psychoacoustic hearing thresholds $\boldsymbol{H}$ are optionally used during the gradient descent to limit modifications of the signal to those time-frequency ranges, where they are (mostly) imperceptible. Here, DNN$_0$ describes the augmented DNN ("Feature extraction" and "DNN") in Figure 3.11 without the RIR simulation, since, for the actual attack, this is replaced by the simulated RIR h_{test}.

3.2.2 Experimental Evaluation

In the following, we evaluate the performance of the proposed algorithm for adversarial examples played over-the-air and compare the performance for varying reverberation times, distances, and adversarial examples restricted by psychoacoustic hearing thresholds. Additionally, we compare the generic approach with an adapted version of the attack where an attacker has prior knowledge of the target room. Finally, we

Algorithm 1 Calculation of robust adversarial examples.

```
1:  input: original audio $x$, target transcription $y'$, hearing thresholds $\boldsymbol{H}$, distribution $\mathcal{H}_\xi$, learning rate $\alpha$, test RIR $h_{\text{test}}$
2:  result: robust adversarial example $x'$
3:  initialize: $g \leftarrow 0$, $x' \leftarrow x$, $\hat{y} \leftarrow$ decode($x' * h_{\text{test}}$) with DNN$_0$
4:  while $g < G$ and $\hat{y} \neq y'$ do
5:      $g \leftarrow g + 1$
6:      draw random sample $h \sim \mathcal{H}_\xi$
7:      update first layer of DNN with $h$
8:      for 1 to $Q$ do
9:          // gradient descent, optionally constrained by hearing thresholds $\boldsymbol{H}$
10:         $\nabla_x \leftarrow \frac{\partial \mathcal{L}(\hat{y}, y')}{\partial x}$
11:         $x' \leftarrow x' + \alpha \cdot \nabla_x$
12:     $x'_h \leftarrow x' * h_{\text{test}}$
13:     $\hat{y} \leftarrow$ decode($x'_h$) with DNN$_0$
```

measure the changes of generic adversarial examples replayed in different rooms, even if there is no direct line-of-sight between the microphone and the speaker.

Metrics

We use the following standard measures to assess the quality of the computed adversarial examples.

Word Error Rate. To measure performance, we use the WER with respect to the target transcription with the definition from Section 3.1.2. For a real attack, an adversarial example can only be considered successful if a WER of 0 % is achieved (i.e., the hypothesis of the system matches with the target transcription chosen by the attacker).

Segmental Signal-to-Noise Ratio. The *segmental Signal-to-Noise Ratio* (SNRseg) measures the amount of noise σ added to the original signal x and is computed as

$$\text{SNRseg(dB)} = \frac{10}{K} \sum_{k=0}^{K-1} \log_{10} \frac{\sum_{t=Tk}^{Tk+T-1} x^2(t)}{\sum_{t=Tk}^{Tk+T-1} \sigma^2(t)}, \tag{3.26}$$

where T is the segment length and K the number of segments. Thus, the higher the SNRseg, the *less* noise was added. The SNRseg measures any added noise, not only the perceptible noise components. Therefore, the perceptible noise is even lower than the SNRseg would suggest for the versions where hearing thresholds are used.

In contrast to the SNR, the SNRseg [107] is computed frame-wise and gives a better assessment of an audio signal if the original and the added noise are aligned [108], as it is the case in our experiments.

Calculation Time

All experiments were performed on a machine with an Intel Xeon Gold 6130 CPU and 128 GB of DDR4 memory.

For our experiments, we limit the maximum number of iterations to 2000, since in every iteration more distortions are added to the audio file, which decreases the audio quality of the adversarial examples. Also, this number is sufficient for the attack to converge, as can be seen in Figure 3.12, where the WER is plotted as a function of the maximum number of iterations G.

Computing an adversarial example for 10 seconds of audio with the maximum number of $G = 2000$ iterations and $M = 512$ takes about 80 minutes. Note that the computation for a single audio file is limited by the single-core performance of the machine, and the attack is fully parallelizable for multiple audio files.

Over-the-Air Attacks

We evaluate the attack for the lab setup as shown in Appendix A in Figure A.4. The approximate dimensions of this room are $\mathbf{b}_{\text{real}} \approx [8\,\text{m}, 7\,\text{m}, 2.8\,\text{m}]$ and the positions of the loudspeaker and the microphone are $\mathbf{s}_{\text{real}} = [3.9\,\text{m}, 3.4\,\text{m}, 1.2\,\text{m}]$ and $\mathbf{r}_{\text{real}} = [1.4\,\text{m}, 1.8\,\text{m}, 1.2\,\text{m}]$, respectively.

We compute all adversarial examples with Algorithm 1. Based on preliminary experiments, we set $G = 2000$ and $Q = 10$. For the distributions $\mathcal{H}_\xi$, we used two different versions, shown in Table 3.4. $\mathcal{H}_{\xi_{\text{gen}}}$ describes a generic room, while $\mathcal{H}_{\xi_{\text{adp}}}$ is used as an approximation to reassemble the real room from Appendix A in Figure A.4. If not specified otherwise, h_{test}, which is used for testing during the attack, is drawn once at the beginning of the algorithm from the same distribution $\mathcal{H}_\xi$.

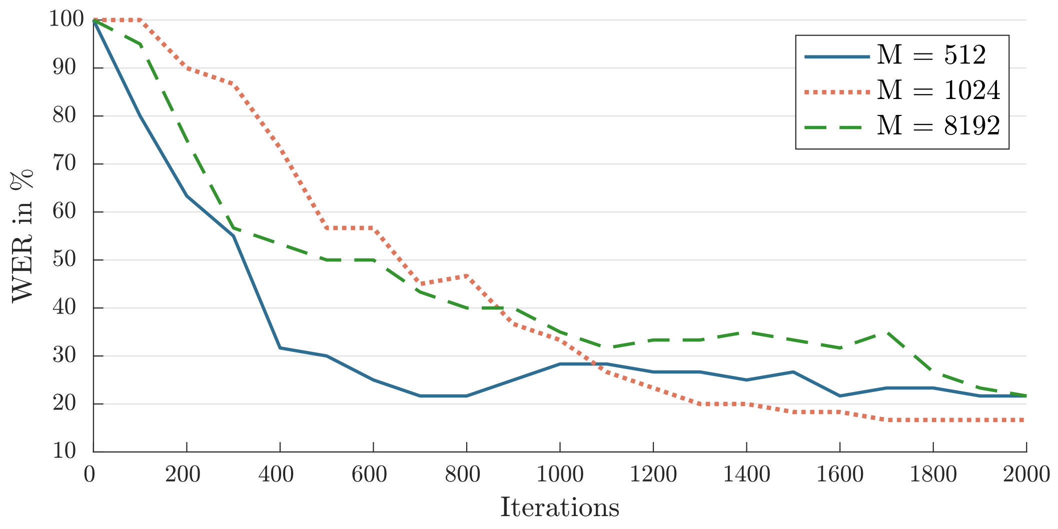


Figure 3.12: WERs for simulated over-the-air attacks as a function of the maximum number of iterations G.

The WER is measured for the recorded adversarial examples after playing them via loudspeaker. The SNRseg is calculated after applying a measured RIR h_{real} to both the original signal and the adversarial example. We chose this approach since it corresponds to the actual signal perceived by human listeners if the adversarial examples are played over the air.

For all cases, we calculated 20 adversarial examples. In some cases, the audio samples clipped too much (exceeded the maximum defined value of the audio, after the addition of the adversarial distortion). As it would not be possible to replay those examples, we removed them from the evaluation of the real over-the-air attack. Each

Table 3.4: Range of room dimensions for sampling the different distributions. $\mathcal{H}_{\xi_{\text{gen}}}$ describes a generic room, which is used for the generic version of the attack, where we assume the attacker to have no prior knowledge. In case of $\mathcal{H}_{\xi_{\text{adp}}}$, the distributions are adapted to the lab setup in Appendix A in Figure A.4.

	b_x		b_y		b_z		T_{60}	
	min	max	min	max	min	max	min	max
$\mathcal{H}_{\xi_{\text{gen}}}$	2.0 m	15.0 m	2.0 m	15.0 m	2.0 m	5.0 m	0.0 s	1.0 s
$\mathcal{H}_{\xi_{\text{adp}}}$	6.0 m	10.0 m	5.0 m	9.0 m	3.0 m	5.0 m	0.2 s	0.6 s

Table 3.5: WER, ratio of successful adversarial examples, and SNRseg for generic over-the-air attacks using $\mathcal{H}_{\xi_{\text{gen}}}$ with speech data for different M and varying T_{60}.

	$M = 512$		$M = 1024$		$M = 8192$	
	WER	AEs	WER	AEs	WER	AEs
$T_{60} = 0.42\,\text{s}$	42.2 %	5/20	34.9 %	5/20	33.3 %	2/20
$T_{60} = 0.51\,\text{s}$	68.9 %	1/20	56.4 %	2/20	42.0 %	2/20
$T_{60} = 0.65\,\text{s}$	91.6 %	0/20	88.0 %	0/20	68.7 %	2/20
SNRseg	7.6±6.7 dB		7.7±6.7 dB		3.2±6.1 dB	

AEs: Successful adversarial examples.

of the remaining adversarial examples was played five times, and we reported the number of adversarial examples that could be transcribed with 0 % WER.

Generic Over-the-Air Attack. First, we evaluate the attack under the assumption that the attacker has no prior knowledge about the attack setup. Specifically, we use $\mathcal{H}_{\xi_{\text{gen}}}$ and calculate adversarial examples for different reverberation times T_{60} and varying length M of RIRs. M describes how many past sampling values are considered, and the larger the reverberation time, the more important are the past sampling values. We assume that, especially in setups with high reverberation times T_{60}, a larger M will result in more robust adversarial examples, as it is a better match to the real-world conditions.

For the experiments, we used the variable-acoustics lab in Appendix A in Figure A.4 to adjust the reverberation time and tested three versions of the RIR length $M = 512$, $M = 1024$, and $M = 8192$ for speech data. The results in Table 3.5 confirm the above assumption: for $M = 8192$, we can obtain the best WERs, especially for the longer reverberation times. Note that even if the WER seems to be high, for an attacker, it is sufficient to play *one* successful adversarial example with 0 % WER, which is also in line with the definition in Section 3.2.1 and, in fact, possible (cf. Table 3.9). The SNRseg decreases with increasing values for M, which indicates that more noise needs

Table 3.6: WER, ratio of successful adversarial examples, and SNRseg for generic over-the-air attacks using $\mathcal{H}_{\xi_{\text{gen}}}$ and hearing thresholds with speech data for different M.

	$M = 512$		$M = 1024$		$M = 8192$	
	WER	AEs	WER	AEs	WER	AEs
$T_{60} = 0.42\,\text{s}$	70.0 %	0/20	62.7 %	0/20	69.6 %	2/20
SNRseg	11.5±5.2 dB		10.4±6.9 dB		5.5±4.8 dB	

AEs: Successful adversarial examples.

to be added to these adversarial examples. Additionally, the calculation time of the adversarial examples increases by the factor of four from $M = 512$ to $M = 8192$.

Hearing Thresholds. To measure the impact of hearing thresholds, we conducted the same experiments as for Table 3.5 with $T_{60} = 0.42$ and hearing thresholds. The results are shown in Table 3.6. Compared to the version without hearing thresholds, the WER and the total number of successful adversarial examples decrease. Nevertheless, it was possible to find successful adversarial examples for $M = 8192$. At the same time, the SNRseg has improved values. Additionally, the SNRseg measures any added noise, not only the perceptible noise components. Therefore, the perceptible noise is even lower than the SNRseg would suggest for the versions where hearing thresholds are used.

Distance between Speaker and Microphone. In Figure 3.13, we measured the effect of an increasing distance between the microphone and loudspeaker. We used the shortest reverberation time $T_{60} = 0.42\,\text{s}$ and varied the distance from 1 m to 6 m for $M = 8192$ with and without hearing thresholds.

In general, we find that the WER increases with increasing distance. Nevertheless, starting from a distance of approximately 2 m, the WER does not increase as rapidly as for smaller distances if we use hearing thresholds. In cases where no hearing thresholds are used, the WER even starts to decrease again for larger distances.

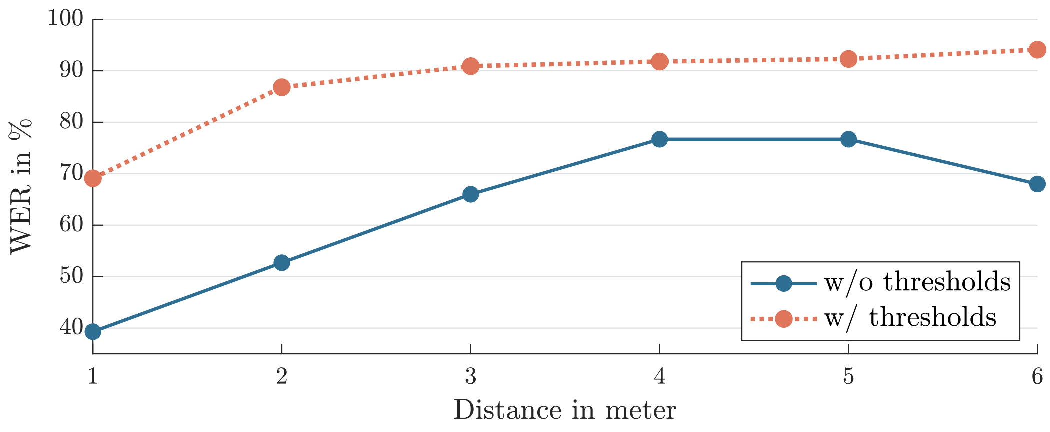


Figure 3.13: WER for over-the-air attacks plotted over the distance between microphone and speaker for $M = 8192$ with and without hearing thresholds.

Varying Audio Content. In Table 3.7, we evaluated the effect of varying audio contents of the original audio samples. For this, we used speech, music, and samples of chirping birds. Using speech audio samples for the attack results in the best WERs.

The average SNRseg indicates that most distortions have to be added to the original audio samples for bird chirping, while we achieve better results for music and speech data.

Adaptive Attack. In the following, we compare the generic attack, where the attacker has no prior knowledge about the attack setup, with an adapted version of the attack. Note that the generic attack is the more powerful attack compared to the adapted version, as it requires no access nor any information about the room where the attack is conducted.

For the evaluation, we used $\mathcal{H}_{\xi_{\text{adp}}}$ and $\mathcal{H}_{\xi_{\text{gen}}}$ in Table 3.4, combined with a measured RIR h_{real} and a simulated RIR h_{gen} for h_{test} in Algorithm 1. h_{gen} was drawn once at the beginning of the algorithm from the same distribution, described via $\mathcal{H}_{\xi_{\text{gen}}}$. h_{real} is a measured RIR, obtained from the recording setup that is actually used during the attack. Consequently, the version with $\mathcal{H}_{\xi_{\text{gen}}}$ and h_{gen} does not use *any* prior knowledge of the room or the recording setup, while the version with $\mathcal{H}_{\xi_{\text{adp}}}$ and h_{real} is tailored to the room. Surprisingly, the generic version clearly outperforms the adapted versions ($\mathcal{H}_{\xi_{\text{adp}}}$, h_{real}) in Table 3.8, and we were able to find fully successful adversarial examples for those cases, i. e., adversarial examples with a WER of 0 %.

Table 3.7: WER, ratio of successful adversarial examples, and SNRseg for different audio content for $M = 512$.

	Music	Speech	Birds
AEs	1/20	5/20	0/20
WER	61.1 %	42.2 %	71.7 %
SNRseg	10.7±2.7 dB	7.6±6.7 dB	1.2±3.0 dB

AEs: Successful adversarial examples.

Table 3.8: WER, ratio of successful adversarial examples, and SNRseg for different audio content for $M = 512$. Comparing generic over-the-air attacks with adapted over-the-air attacks.

	Music		Speech		Birds	
	WER	AEs	WER	AEs	WER	AEs
$\mathcal{H}_{\xi_{\text{gen}}}$, h_{gen}	61.1 %	1/20	42.2 %	5/20	71.7 %	0/20
$\mathcal{H}_{\xi_{\text{adp}}}$, h_{adp}	63.2 %	2/20	65.0 %	2/20	84.5 %	2/20
Δ in WER	+ 2.1 %		+ 22.8 %		+ 12.8 %	

AEs: Successful adversarial examples.

As a consequence, for an attacker, it is not only unnecessary to acquire prior knowledge about the room characteristics, but the likelihood of success is even higher if a generic attack is chosen.

Varying Room Conditions. To evaluate the adversarial examples in varying rooms, we chose three rooms of differing sizes: a lecture room with approximately 77 m^2, a meeting room with approximately 38 m^2, and an office with approximately 31 m^2. Layout plans of the rooms are shown in Appendix A, including positions of the speaker and microphone for all recording setups and the measured reverberation time.

Table 3.9: WER and ratio of successful adversarial examples for generic over-the-air attacks with and without direct line-of-sight in varying rooms based on speech data for $M = 8192$.

	Lecture Room		Meeting Room		Office	
	WER	AEs	WER	AEs	WER	AEs
w/ line-of-sight	40.0 %	2/20	55.3 %	1/20	74.0 %	1/20
w/o line-of-sight	71.3 %	0/20	62.0 %	0/20	82.7 %	1/20
Δ in WER	+ 31.3 %		+ 6.7 %		+ 8,7 %	

AEs: Successful adversarial examples.

Direct Line-of-Sight Attack. The first attacks were conducted with a direct line-of-sight between the microphone and the speaker. The results are shown in Table 3.9. Even though the results vary depending on the room, the WERs remain approximately in the same range as the experiments with varying reverberation times in Table 3.5 would indicate. Surprisingly, the room with the highest reverberation time, the lecture room, actually gave the best results.

Overall, the results show that our generic adversarial examples remain robust for different kinds of rooms and setups and that it is sufficient to calculate one version of an adversarial example to cover a wide range of rooms (i.e., the attack is transferable).

No-Line-of-Sight Attack. For the rooms in Table 3.9, we also performed experiments where no line-of-sight between the microphone and the speaker exists by blocking the direct over-the-air connection with different kinds of furniture. As a consequence, not the direct sound, but a reflected version of the audio is recorded. An implication is that these attacks could be carried out without being visible to people in the vicinity of the ASR input microphone. We tested different scenarios for our setup: In the lecture and the meeting room, the source and the receiver were separated by a table by simply placing the speaker under the table. In the office, the speaker was even placed *outside* the room. For this recording setup, the door between the rooms was left open and the line-of-sight was blocked by a wall. For all other setups, the doors of the respective rooms were closed. A detailed description of the no-line-of-sight setups is given in Appendix A.

In cases where no line-of-sight exists, the distortions that occur through the transmission can be considered more complex, and consequentially, a prediction of the recorded audio signal is harder. A blocked line-of-sight will most likely decrease the WER, but it is in general possible to find adversarial examples with 0 % WER, even where the source was placed outside the room. This again shows that our generic version of the attack can successfully model a wide range of acoustic environments simultaneously, without any prior knowledge about the room setup.

Attack Parameters

Our experiments show that the adversarial examples, which we calculated with the proposed algorithm, remain robust even for high reverberation times or large distances between speaker and microphone. Also, the same adversarial examples can be successfully played over the air, even for setups where no direct line-of-sight exists.

Our comparison between the generic and the adapted version of the attack shows that the more powerful generic attack does not only have a similar success rate but can even outperform an adapted version where the attacker has prior knowledge of the target room. Consequently, an attacker only needs to calculate one generic adversarial example to cover a wide range of possible recording setups simultaneously.

For an attack, one successful adversarial example, which remains robust after being replayed (with a WER of 0 %), is already enough. Therefore, the best strategy for an actual attack would be to calculate a set of adversarial examples containing the malicious transcription and to choose the most robust ones. In general, the results indicate a trade-off between the WER and the noise level: if no hearing thresholds are used, the WER is significantly better in comparison to examples computed with hearing thresholds. Nevertheless, even if the WER is better in cases without hearing thresholds, we have shown that it is indeed possible to calculate over-the-air-robust adversarial examples with distortions limited by hearing thresholds. Those adversarial examples contain less perceptible noise and are, therefore, less likely to be detected by human listeners.

3.3 Detecting Adversarial Examples

In the following section, we show a detection mechanism for potential audio adversarial examples for DNN-HMM ASR systems. For this purpose, we combine the insights about uncertainty quantification from the deep learning community with the core mechanisms of hybrid ASR systems. The approach can be summarized with the following three steps:

- We substitute the ASR system's standard *feed-forward Neural Network* (fNN) with different network architectures, which are capable of capturing model uncertainty, namely *Bayesian Neural Networks* (BNNs) [109], Monte Carlo (MC) dropout [110] and deep ensembles [111].

- We calculate different measures to assess the uncertainty when evaluating the acoustic model for an utterance. Specifically, we measure the entropy, variance, averaged *Kullback-Leibler Divergence* (KLD), and the mutual information of the neural network outputs.

- We train a one-class classifier by fitting a normal distribution on an exemplary set of benign examples. Adversarial examples can then be detected as outliers of the learned distribution. Compared to previous work, this has the advantage that we do not need any adversarial examples to train the classifier and are not tailored to specific kinds of attacks.

The results show that we are able to detect adversarial examples with an area under the receiver operating characteristic curve of more than 0.99 using the neural network output entropy. Additionally, the NNs used for uncertainty quantification are less vulnerable to adversarial attacks when compared to standard feed-forward neural networks.

3.3.1 Neural Networks for Uncertainty Quantification

A range of approaches have recently been proposed for quantifying uncertainty in neural networks:

Bayesian Neural Networks. A mathematically grounded method for quantifying uncertainty in neural networks is given by BNNs [109]. Central to these methods is the calculation of a posterior distribution over the network parameters, which models the probabilities of different prediction networks. The final predictive function is derived as

$$p(\hat{y}|x, D) = \int p(\hat{y}|x, \boldsymbol{\theta})p(\boldsymbol{\theta}|D)d\boldsymbol{\theta}, \tag{3.27}$$

where $p(\boldsymbol{\theta}|D)$ is the posterior distribution of the parameters $\boldsymbol{\theta}$, $\hat{y}$ the output, x the input, and $D = \{(x_i, \hat{y}_i)\}_{i=1}^{n}$ the training set. To approximate the often intractable posterior distribution, variational inference methods can be applied. These fit a simpler distribution $q(\boldsymbol{\theta}|D)$ as close to the true posterior as possible by minimizing their KLD. Minimizing this, again intractable, KLD is equivalent to maximizing the so-called *Evidence Lower Bound* (ELBO) given by

$$\mathbb{E}_{q(\boldsymbol{\theta}|D)}[\log p(\hat{y}_i|x_i, \boldsymbol{\theta})] - \text{KLD}[q(\boldsymbol{\theta}|D)||p(\boldsymbol{\theta})]. \tag{3.28}$$

For the optimization, the expectation is approximated as the mean of $\log p(\hat{y}_i|x_i, \boldsymbol{\theta})$ for all $i = 1, \dots, n$. During the prediction, the integral of Equation (3.27) are approximated by averaging $p(\hat{y}|x, \boldsymbol{\theta}_j)$ for multiple samples $\boldsymbol{\theta}_j$ drawn from $q(\boldsymbol{\theta}|D)$. While there are different approaches to BNNs, we follow Louizos et al. [112] in this paper.

Monte Carlo Dropout. Another approach that scales to deep neural network architectures is Monte Carlo dropout [110], which was introduced as an approximation to the Bayesian inference. In this approach, the neurons of a neural network are dropped with a fixed probability during training and testing. This can be seen as sampling different sub-networks consisting of only a subset of the neurons and leading to different prediction results for the same input. Here, $\boldsymbol{\theta}_j$ denotes the model parameters for the j-th sub-network and the final prediction is given by $p(\hat{y}|x) = \frac{1}{J}\sum_{j=1}^{J} p(\hat{y}|x, \boldsymbol{\theta}_j)$.

Deep Ensembles. A simple approach, which has been found to often outperform more complex ones [113], is the use of a deep ensemble [111]. The core idea is to train multiple neural networks with different parameter initialization on the same data set. In this context, we denote the prediction result of the j-th neural network by $p(\hat{y}|x, \boldsymbol{\theta}_j)$. The final prediction is again given by the average over all J model $p(\hat{y}|x) = \frac{1}{J}\sum_{j=1}^{J} p(\hat{y}|x, \boldsymbol{\theta}_j)$.

3.3.2 Detecting Adversarial Examples

For the detection of the attack, i. e., the identification of adversarial examples, we describe the general attack setting and the different uncertainty measures that we employ.

Threat Model

We assume a white-box setting in which the attacker has full access to the model, including all parameters. Using this knowledge, the attacker generates adversarial examples offline. We only consider targeted attacks, where the adversary chooses the target transcription. Additionally, we assume that the trained ASR system remains unchanged over time.

Uncertainty Measures

For quantifying prediction uncertainty, we employ the following measures:

Entropy. To measure the uncertainty of the network over class predictions, we calculate the *entropy* over the K output classes as

$$\mathcal{H}[p(\hat{\boldsymbol{y}}|x)] = -\sum_{k=1}^{K} p(\hat{y}_k|x) \cdot \log p(\hat{y}_k|x). \tag{3.29}$$

This can be done for all network types, including the fNN with a softmax output layer. We calculate the entropy for each time step and use its maximum value as the uncertainty measure.

Mutual Information. To leverage the possible benefits of replacing the fNN with a BNN, MC dropout, or a deep ensemble, we evaluate the multiple predictions $p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_j)$ for $j = 1, ..., J$ of these networks. Note that these probabilities are derived differently for each network architecture, as described in Section 3.3.1. With this setup we can calculate the *Mutual Information* (MI), which is upper bounded by the entropy of the probabilities. The MI is a measure to tell how much the uncertainty of a random variable X is decreased given another random variable Y [114]. It is defined as

$$I(X;Y) = \mathcal{H}(X) - \mathcal{H}(X|Y). \tag{3.30}$$

In our case we measure the MI between the prediction $p(\hat{\boldsymbol{y}}|x)$ and the prediction given the parameters $\boldsymbol{\theta}_j$. For the final measure, the results are averaged over all $j = 1, \dots, J$

$$\mathrm{MI} = \frac{1}{J}\sum_{j=1}^{J} \mathcal{H}[p(\hat{\boldsymbol{y}}|x)] - \mathcal{H}[p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_j)]. \tag{3.31}$$

The MI indicates the inherent uncertainty of the model on the presented data [111].

Variance. Another measure that has been used by Feinman et al. [115] to detect adversarial examples for image recognition tasks is the *variance* of the different predictions:

$$\frac{1}{J}\sum_{j=1}^{J}[p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_j) - p(\hat{\boldsymbol{y}}|x)]^2. \tag{3.32}$$

Averaged Kullback-Leibler Divergence. To observe the variations of the distribution, we further introduce the *averaged Kullback-Leibler Divergence* (aKLD). The KLD describes the distance between two distributions $p(x)$ and $p(x)$ [114]. It is defined as

$$D(p||q) = \sum_{x\in\mathcal{X}} p(x) \cdot \log\frac{p(x)}{q(x)}, \tag{3.33}$$

where $\mathcal{X}$ describes the set of all possible values of the distributions $p(x)$ and $q(x)$. We use the aKLD between two predictions with two different instances of $\boldsymbol{\theta}_j$

$$\frac{1}{J-1}\sum_{j=1}^{J-1} p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_j) \cdot \log\frac{p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_j)}{p(\hat{\boldsymbol{y}}|x, \boldsymbol{\theta}_{j+1})}. \tag{3.34}$$

Because the samples $\boldsymbol{\theta}_j$ are drawn independently, we compare the first drawn example to the second, the second to the third, and so on, without any reordering.

3.3.3 Experiments

In the following, we describe implementation details and present the results of our experimental analysis.

Recognizer

We use a hybrid DNN-HMM ASR system. As a proof of concept for adversarial example detection, we focus on a simple recognizer for sequences of digits from 0 to 9.

Table 3.10: Accuracy on benign examples.

fNN	deep ensemble	MC dropout	BNN
0.991	0.994	0.973	0.981

We train the recognizer with the *TIDIGITS* training set, which includes approximately 8000 utterances of digit sequences. The feature extraction is integrated into the NNs via *torchaudio*. We use the first 13 MFCCs and their first and second derivatives as input features and train the NNs for 3 epochs followed by 3 additional epochs of Viterbi training to improve the ASR performance.

We use NNs with two hidden layers, each with 100 neurons, and a softmax output layer of size 95, corresponding to the number of states of the HMM. For the deep ensemble, we train $J = 5$ networks with different initialization; for the BNN, we draw $J = 5$ models from the posterior distribution and average the outputs to form the final prediction; and for dropout, we sample $J = 100$ sub-networks for the average prediction.[1]

The ASR accuracies are evaluated on a test set of 1000 benign utterances and are shown in Table 3.10, calculated as the sum over all substituted words S, inserted words I, and deleted words D in comparison to the original and the target label

$$\text{Accuracy} = \frac{N - D - I - S}{N} \quad , \tag{3.35}$$

where N is the total number of words of the reference text, which is either the original or the malicious target text.

All methods lead to a reasonable accuracy, with the deep ensemble models outperforming the fNN. At the same time, there is some loss of performance for the MC dropout model and the BNN model.

[1]Note, that we needed to increase the number of samples for dropout compared to the other methods, since using $J = 5$ for dropout led to worse recognition accuracy. Moreover, we also needed to estimate the average gradient over 10 sub-nets per training sample during training to observe increased robustness against adversarial examples.

Adversarial Attack

For the attack, we use a sequence of randomly chosen digits with a random length between 1 and 5. The corresponding targets for the attack have been calculated with the Montreal forced aligner [86]. To pass the targets through the NN we used the PGD attack [116]. For this purpose, we used cleverhans, a Python library to assess machine learning systems against adversarial examples [117].

During preliminary experiments, we found that using multiple samples for estimating the stochastic gradient for the estimation of adversarial examples decreases the strength of the attack. This result contradicts insights found for BNNs in image classification tasks, where the adversarial attacks become stronger when multiple samples are drawn for the gradient [118]. An explanation for this finding could be that for image classification, no hybrid system is used. In contrast to that, the Viterbi decoder in a hybrid ASR exerts an additional influence on the recognizer output and favors cross-temporal consistency.

Correspondingly, our empirical results indicate that sampling multiple times leads to unfavorable results for ASR from the attacker's perspective. Evaluating the averaged and the single adversarial examples separately shows that the averaged adversarial examples are more likely to return the original text due to the Viterbi decoding of the hybrid ASR system. Consequently, we have only used one sample to improve the attacker's performance and, thus, evaluate our defense mechanisms against a harder opponent. To validate the effectiveness of PGD, we investigate the word accuracy of the label predicted for the resulting adversarial example w.r.t. the target and the original transcription. These word accuracies are shown in Figure 3.14 for varying perturbation strength ($\epsilon = 0, \dots, 0.1$ with a step size of 0.01) of the PGD attack. Note that $\epsilon = 0$ corresponds to benign examples, as no perturbations are added to the original audio signal. We evaluated 100 adversarial examples for each ϵ and NN.

For all models, the accuracy w.r.t. the target transcription increases with increasing perturbation strength until approximately $\epsilon = 0.04$, and stagnates afterward. The attack has the most substantial impact on the fNN-based model, where the accuracy w.r.t. the malicious target transcription for $\epsilon \geq 0.05$ is almost 50 % higher than for the other models, for which the accuracy only reaches values between 0.4 and 0.7. This indicates that including NNs for uncertainty quantification into ASR systems makes it more challenging to calculate effective targeted adversarial attacks. Nevertheless, the

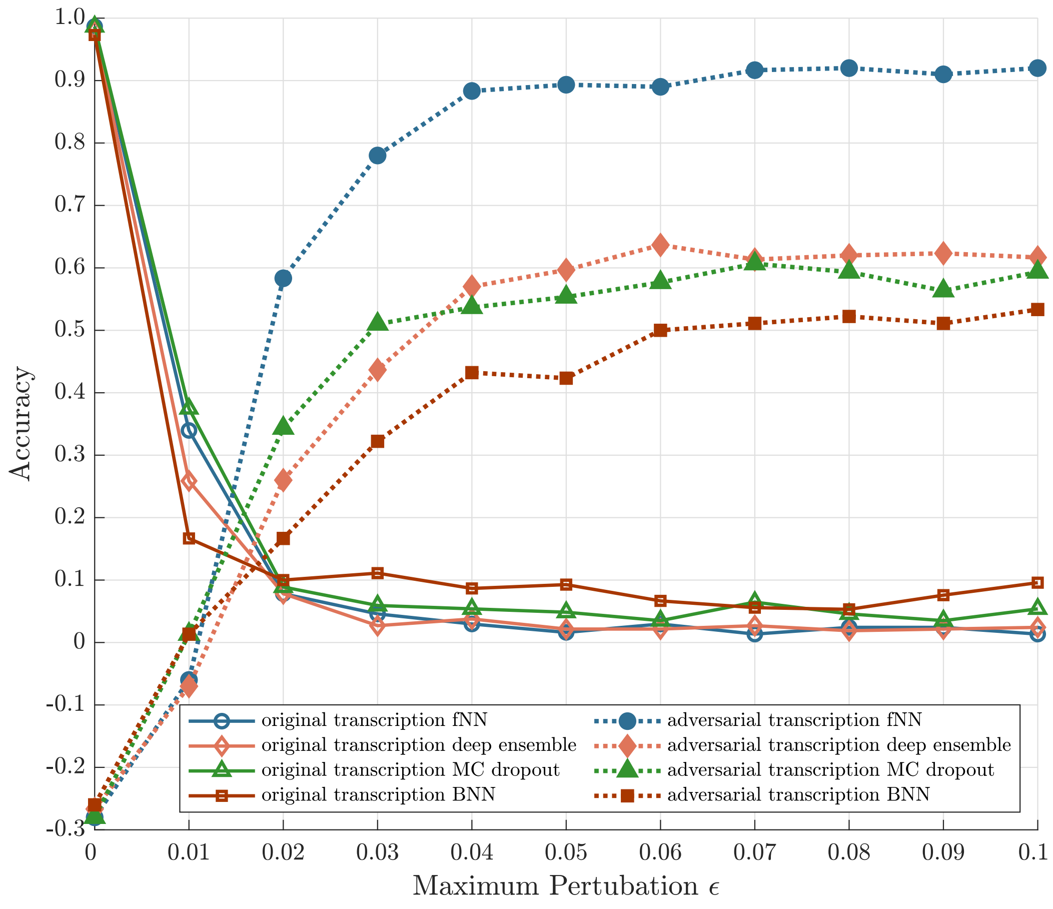


Figure 3.14: Accuracy with respect to the original and the target transcription plotted over ϵ for fNN, MC dropout, BNN, and deep ensemble, evaluated on 100 utterances each.

accuracy w.r.t. the original transcription is almost equally affected across all systems, indicating that for all of them, the original text is difficult to recover under attack.

Classifying Adversarial Examples

In order to detect adversarial examples, we calculate the measures described in Section 3.3.2 for 1000 benign and 1000 adversarial examples, estimated via PGD with $\epsilon = 0.05$. Figure 3.15 exemplarily shows histograms of the entropy values of the predictive distribution of the fNN over both sets of examples. Like the fNN, all other models also have a clear tendency to display higher uncertainty over classes for adver-

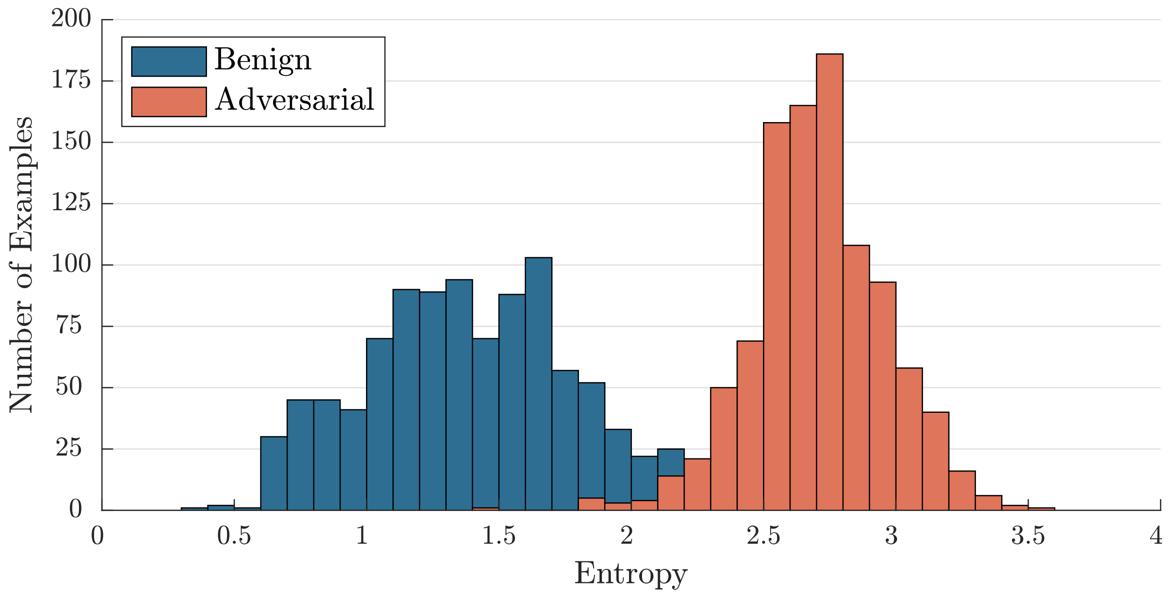


Figure 3.15: Histograms of predictive entropy values for an fNN for 1000 benign and 1000 adversarial examples.

sarial examples, with the largest difference between benign and adversarial examples was most severe for the entropy.

We build on this observation by constructing simple classifiers for the detection of adversarial examples: We fit a Gaussian distribution to the values of the corresponding measure over a held-out data set of 1000 benign examples for each network and measure. A new observation can then be classified as an attack if the value of the prediction uncertainty has low probability under the Gaussian model. We measure the *Receiver Operating Characteristic* (ROC) of these classifiers for each model type and uncertainty measure. The results are shown exemplarily for the BNN in Figure 3.16. Additionally, we display the *Area Under the Receiving Operator Curve* (AUROC) in Table 3.11. The results show that only the entropy has stable performance across all kinds of NNs and clearly outperforms the other measures (variance, aKLD, and MI). Note that the entropy is also the only measure that can be calculated for the fNN.

To verify the results for adversarial examples with low perturbations, which might be harder to detect, we followed the same approach for 1000 adversarial examples with a maximal perturbation of $\epsilon = 0.02$.

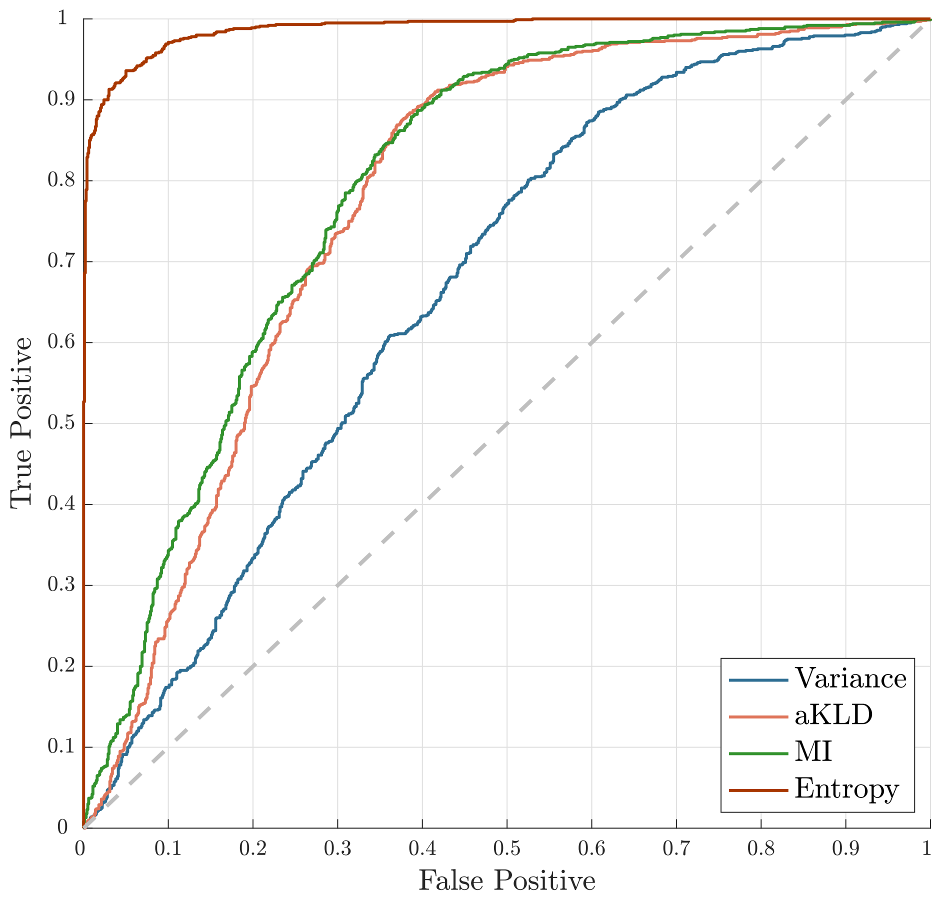


Figure 3.16: ROC curves of the different measures for the BNN with $\epsilon = 0.05$ on 1000 benign and adversarial examples each.

3.4 Related Work

The following summarizes related work on audio adversarial examples and their countermeasures.

Audio Adversarial Examples. By creating one of the first targeted attacks, Carlini et al. [78] have shown that these are possible against GMM-HMM ASR systems. In their paper, they use an inverse feature extraction to create adversarial audio samples. The resulting audio samples are not intelligible by humans in most of the cases and may be considered as noise, but may make thoughtful listeners suspicious. A different approach was shown by Vaidya et al. [77], where the authors changed an input signal to fit the target transcription by considering the features instead of the

Table 3.11: AUROC feature scores for 1000 adversarial examples with a perturbation strength $\epsilon = 0.05$. Best results for each network are shown in bold.

	Variance	aKLD	MI	Entropy
fNN	–	–	–	**0.989**
deep ensemble	0.455	0.892	**0.993**	0.990
MC dropout	0.637	0.443	0.498	**0.978**
BNN	0.667	0.777	0.794	**0.988**

Table 3.12: AUROC feature scores for 1000 adversarial examples with a perturbation strength $\epsilon = 0.02$. Best results for each network are shown in bold.

	Variance	aKLD	MI	Entropy
fNN	–	–	–	**0.997**
deep ensemble	0.461	0.624	0.964	**0.996**
MC dropout	0.937	0.578	0.411	**0.991**
BNN	0.489	0.448	0.462	**0.998**

output of the DNN. Nevertheless, the results show high distortions of the audio signal and can easily be detected by a human listener.

An approach to overcome this limitation was proposed by Zhang et al. [79] They have shown that an adversary can hide a transcription by utilizing non-linearities of microphones to modulate the baseband audio signals with ultrasound above 20 kHz. The main downside of the attack is the fact that the information of the necessary features needs to be retrieved from audio signals, recorded with the specific microphone, which is costly in practice. Furthermore, the modulation is tailored to a specific microphone an adversary wants to attack. As a result, the result may differ if another microphone is used.

Carlini and Wagner [75] published a technical report in which they introduce a general targeted attack on ASR systems using the CTC-loss [76]. The attack utilizes a gradient-descent-based minimization via the CTC-loss, which is designed for the recognition of time series data.

CommanderSong [82] is also evaluated against Kaldi and uses backpropagation to find adversarial examples. However, in order to minimize the noise, approaches from the image domain are borrowed. Therefore, the algorithm does not consider human auditory perception.

Alzantot et al. [119] proposed a black-box attack, which does not require knowledge about the model structure or parameters. For this purpose, the authors use a genetic algorithm to create their adversarial examples for a keyword spotting system, which differs from general speech recognition due to a much simpler architecture and far fewer possible recognition outcomes. For *DeepSpeech* [120] and *Kaldi*, Khare et al. [121] proposed a black-box attack based on evolutionary optimization, and Taori et al. [122] also present a similar approach in their paper.

The approach presented in Section 3.1 is different from all previous studies on adversarial perturbations for ASR, as we combine a targeted attack with the requirement that the added noise should be barely, if at all, perceptible. We use a modified backpropagation scheme, which has been very successful in creating adversarial perturbations for image classification and we initialize our optimization by forced-alignment to further minimize audible noise. Also, the psychoacoustics-based approach focuses on the human-perceptible frequency range, which is different from ultrasound-based approaches.

Adversarial Examples for Realistic Environments. Most these previous approaches have in common that the audio signal was fed directly into the ASR system. In the following, we outline works that present adversarial examples, which remain viable when played over the air:

Yakura and Sakuma [103] published a technical report, which describes an algorithm to create adversarial examples, which remain viable when played over the air, but with the limitation that it is necessary to have physical access to the room, where the attack takes place. Also, they did not evaluate their room-dependent examples for varying room conditions and were unable to create generic adversarial examples systematically.

Szuley and Kolter [104] also published a work on room-dependent robust adversarial examples, which worked under constraints given by a psychoacoustic model. However, their adversarial examples only work in an anechoic chamber, a room specifically designed to eliminate the effect of an RIR. A comparison with a real-world scenario is, therefore, not feasible, as the anechoic chamber effectively reproduces the effect of directly feeding the attack into the ASR system. Li et al. [123] published a work to obfuscate Amazon's Alexa *wake word* via specifically crafted music. However, their approach was not successful at creating *targeted* adversarial examples that work over the air.

Abdullah et al. [102] showed a black-box attack, in which psychoacoustics is used to calculate adversarial examples empirically. Their approach focuses on over-the-air attacks, but in many cases, humans can perceive the hidden message once they are alerted to its content. The attack presented in Section 3.2 is conceptually completely different, as we use a target audio file, where we embed the target transcription via backpropagation. The changes, therefore, sound like random noise. With Abdullah et al.'s approach, in contrast, an audio file with the spoken target text forms the starting point and is changed in such a way as to be unintelligible adversarial examples for realistic environments for unbiased human listeners, but not for humans aware of the target transcription. This is also the case for Chen et al.'s [124] black-box attack against several commercial devices, where humans can again perceive the target text.

As an extension of Carlini's and Wagner's attack [76], Qin et al. [125] introduced the first implementation of RIR-independent adversarial examples. Unfortunately, their approach only worked in a simulated environment and not for real over-the-air attacks, but the authors also utilize psychoacoustics to limit the perturbations.

The approach presented in Section 3.2 is the first targeted attack that provides room-independent, robust adversarial examples against a hybrid ASR system. We demonstrate how to generate adversarial examples that are mostly unaffected by the environment, as ascertained by verifying their success in a broad range of room characteristics. We utilize the same psychoacoustics-based approach proposed in Section 3.1 to limit the perturbations of the audio signal to remain below, or at least close to, the human thresholds of hearing, and we show that the examples remain robust to playback over the air. The perturbations that remain audible in the adversarial examples that we create are non-structured noise, so that human listeners cannot perceive any content related to the targeted recognition output. Hence, our attack can be successful in a broad range of possible rooms, without any physical access to the environment (e. g., by playback of inconspicuous media from the Internet), and where the target recognition output is not perceptible by human listeners. It shows the possibility and risk of a new attack vector, as no specialized hardware is needed for the playback and by being insensitive to the rooms in which the attacked systems are being operated.

Countermeasures. There have been numerous attempts to tackle the problem of adversarial examples in neural networks. However, it has been shown that the existence of these examples is a consequence of the high dimensionality of neural network architectures [126, 127]. To defend against adversarial attacks, several approaches aim e. g., at making their calculation harder by adding stochasticity and reporting prediction uncertainties [128, 129, 115]. Ideally, the model should display high uncertainties if and only if abnormal observations like adversarial examples or out-of-distribution data are fed to the system.

Akinwande et al. [130] and Samizade et al. [131] used anomaly detection, either in the network's activations or directly on raw audio, to detect adversarial examples. However, both methods are trained for defined attacks and are therefore easy to circumvent [132].

Zeng et al. [133] have combined the output of multiple ASR systems and calculated a *similarity score* between the transcriptions, but due to the transferability property of adversarial examples to other models, this countermeasure is not guaranteed to be successful [95].

Yang et al. [134] also utilize temporal dependencies of the input signal. For this, they compare the transcription of the entire utterance with a segment-wise transcription of the utterance. In case of a benign example, both transcriptions should be the same, which will typically not be the case for an adversarial example.

Liu and Ditzler [135] utilizing quantization error of the neural network's activations, which appear to be different for adversarial and benign audio examples.

Esmaeilpour et al. [136] used an approximation of the audio signal calculated by a *Generative Adversarial Network* (GAN) to defeat adversarial examples. However, their approach might easily be leveraged by fooling the GAN of the defense mechanism as these have shown to be vulnerable against adversarial examples, too.

Eisenhofer et al. [137] followed another approach; accepting that adversarial examples will always be possible, the target is to make adversarial audio perturbations perceptible. For this purpose, they remove semantically irrelevant information and use only these time-frequency ranges of the audio sample that humans can perceive. This forces a potential attacker to make audible changes and the perturbations, therefore, are much more conspicuous.

Other works leveraged uncertainty measures to improve the robustness of ASR systems in the absence of adversarial examples. Vyas et al. [138] and Jayashankar et al. [139]

used dropout and the respective transcriptions to measure the reliability of the ASR system's prediction. Abdelaziz et al. [140] and Huemmer et al. [141] have previously utilized the propagation of observation uncertainties through the layers of a neural network acoustic model via Monte Carlo sampling to increase the reliability of these systems under acoustic noise.

In the approach presented in Section 3.3, we do not focus on building a model that prevents adversarial examples but rely on outlier detection and fit the model in benign data only. This makes the system more robust against unknown attacks as it is not tailored to specific ones.

3.5 Summary

We have presented a new method for creating adversarial examples for ASR systems, which explicitly take dynamic human hearing thresholds into account. In this way, borrowing the mechanisms of MP3 encoding, the audibility of the added noise is clearly reduced. We have performed our attack against the state-of-the-art *Kaldi* ASR system and feed the adversarial input directly into the recognizer in order to show the general feasibility of psychoacoustics-based attacks.

By applying forced alignment and backpropagation to the DNN-HMM system, we were able to create inconspicuous adversarial perturbations very reliably. In general, it is possible to hide any target transcription within any audio file and, with the correct attack vectors, it was possible to hide the noise below the hearing threshold and make the changes psychophysically almost imperceptible. The choice of the original audio sample, an optimal phone rate, and forced alignment give the optimal starting point for the creation of adversarial examples. Additionally, we have evaluated different parameterizations, including the number of iterations and the allowed deviation from the hearing thresholds. The comparison with another approach by Yuan et al. [82], which is also able to create targeted adversarial examples, shows that our approach needs far lower distortions. Listening tests have proven that the target transcription was incomprehensible for human listeners. Furthermore, for some audio files, it was almost impossible for participants to distinguish between the original and adversarial sample, even with headphones and in a direct comparison.

In Section 3.2, we have demonstrated that ASR systems are vulnerable against adversarial examples played over the air, and we have introduced an algorithm for the calculation of robust adversarial examples. By simulating varying room setups, we can create highly robust adversarial examples that remain successful over the air in many environments.

We presented the results of empirical attacks for different room configurations. Our algorithm can be used with and without psychoacoustic hearing thresholds, limiting the perturbations to being less perceptible by humans. Furthermore, we have shown that it is possible to create targeted robust adversarial examples for varying rooms and varying audio content, even if no direct line-of-sight between the microphone and the speakers exists, and even if the test room characteristics are completely unknown during the creation of the example.

Finally, in Section 3.3 we introduce a mechanism to harden hybrid speech recognition systems by replacing the standard feed-forward neural network with a Bayesian neural network, Monte Carlo dropout, or deep ensemble networks. Our empirical results show that this increases the robustness against targeted adversarial examples tremendously. This can be seen in the low accuracy of the target transcription, which indicates a far lower vulnerability than that of standard hybrid speech recognition.

Another finding of this work is that the entropy serves as a good measure for identifying adversarial examples. In our experiments, we were able to discriminate between benign and adversarial examples with an AUROC score of up to 0.99 for all network architectures. Interestingly, the other measures, which are available when using approaches especially designed for uncertainty quantification, did not improve upon these results.

4 | Exploring Accidental Triggers

In the past few years, we have observed a huge growth in the popularity of voice assistants, especially in the form of smart speakers. Most major technology companies, among them Amazon, Baidu, Google, Apple, Tencent, and Xiaomi, have developed an assistant. Amazon is among the most popular brands on the market: the company reported in 2019 that it had sold more than 100 million devices with *Alexa* on board; there were more than 150 products that support this voice assistant (e. g., smart speakers, soundbars, headphones, etc.) [142]. Especially smart speakers are on their way to becoming a pervasive technology, with several security and privacy implications due to the way these devices operate: they continuously analyze every sound in their environment in an attempt to recognize a so-called *wake word* such as “Alexa,” “Echo,” “Hey Siri,” or “Xio dù xio dù.” If and only if a wake word is detected, the device starts to record the sound and uploads it to a remote server, where it is transcribed, and the detected word sequence is interpreted as a command. This mode of operation is mainly used due to privacy concerns, as the recording of all (potentially private) communication and processing this data in the cloud would be too invasive. Furthermore, the limited computing power and storage on the speaker prohibits a full analysis on the device itself. Hence, the recorded sound is sent to the cloud for analysis once a wake word is detected.

Unfortunately, the precise detection of wake words is a challenging task with a typical trade-off between usability and security: manufacturers aim for a low false acceptance and false rejection rate [143], which promotes a certain wiggle room for an adversary. As a result, it happens that these smart speakers trigger even if the wake word has not been uttered. First exploratory work on the error patterns of voice-driven user input has been done by Vaidya et al. [77]. In their 2015 paper, the authors explain how Google's voice assistant, running on a smartphone, *misinterprets* "cocaine noodles" as "OK Google" and they describe a way to exploit this behavior to execute unauthorized commands such as sending a text, calling a number, or opening a website. Later, Kumar et al. [144] presented an attack, called *skill squatting*, that leverages transcription errors of a list of words sounding similar to existing Alexa skills. Their attack exploits the *imperfect transcription* of the words by the Amazon API and routes users to malicious skills with similar-sounding names. A similar attack, in which the adversary exploits the way a skill is invoked, has been described by Zhang et al. [145].

Such research results utilize instances of what we call an *accidental trigger*: a sound that a voice assistant mistakes for its wake word. Privacy-wise, this can be fatal, as it will induce the voice assistant to start a recording and stream it to the cloud. Inadvertent triggering of smart speakers and the resulting accidentally captured conversations are seen by many as a privacy threat [146, 147, 148]. When the media reported in summer 2019 that employees of the manufacturer listen to voice recordings to transcribe and annotate them, this led to an uproar [149, 150]. As a result, many companies paused these programs and no longer manually analyze the recordings [151, 152, 153].

In this chapter, we perform a systematic and comprehensive analysis of accidental triggers to understand and elucidate this phenomenon in detail. To this end, we propose and implement an automated approach for systematically evaluating the resistance of smart speakers to such accidental triggers. We base this evaluation on candidate triggers carefully crafted from a pronouncing dictionary with a novel phonetic distance measure, as well as on available AV media content and bring it to bear on a range of current smart speakers. More specifically, in a first step, we analyze the vendor's protection mechanisms such as cloud-based wake word verification systems, used to limit the impact of accidental triggers. We carefully evaluate how a diverse set of 11 smart speakers from 8 manufacturers behaves in a simulated living-room-like scenario with different sound sources (e. g., TV shows, news, and professional audio datasets). We explore the feasibility of artificially crafting accidental triggers using a pronouncing dictionary and a weighted, phone-based Levenshtein distance metric and benchmark the robustness of the smart speakers against such crafted accidental triggers. We found that a distance measure that considers phone-dependent weights is more successful in describing potential accidental triggers. Based on this measure, we crafted 1-, 2-, and 3-grams as potential accidental triggers, using a TTS service and were able to find accidental triggers for all tested smart speakers in a fully automated way.

Finally, we give recommendations and discuss countermeasures to reduce the number of accidental triggers or limit their impact on users' privacy.

To summarize, we make the following key contributions:

- We develop a fully automated measurement setup that enables us to perform an extensive study of the prevalence of accidental triggers for 11 smart speakers from 8 manufacturers. We analyze a diverse set of audio sources, explore potential gender and language biases, and analyze the identified triggers' reproducibility.

- We introduce a method to synthesize accidental triggers with the help of a pronouncing dictionary and a weighted phone-based Levenshtein distance metric. We demonstrate that this method enables us to find new accidental triggers in a systematic way and argue that this method can benchmark the robustness of smart speakers.

4.1 Understanding Accidental Triggers

In this section, we provide the required background on wake word detection. Furthermore, we describe how Amazon deals with accidental triggers. Finally, we provide an overview of smart speaker privacy settings. In general, accidental triggers are the consequence of the trade-off between specificity and sensitivity, namely the false rejection and the false acceptance rate. Whenever a wake word recognizer is trained, the system aims to minimize both of these errors.

4.1.1 Wake Word Recognition

To enable natural communication between the user and the device, automatic speech recognition (ASR) systems built into smart speakers rely on a far-field voice-based activation. In contrast to a push-to-talk model, where speech recognition is only active after a physical button is pressed, smart speakers *continuously* record their surroundings to allow hands-free use. After detecting a specific *wake word*, also known as hotword or keyword, the smart speaker starts to respond. The wake word recognition system is often a lightweight DNN-based ASR system, limited to a few designated words [154, 155, 156]. To guarantee its responsiveness, the recognition runs locally and is therefore limited by the computational power and storage of the speaker. For example, we found the *Computer* wake word model for an Amazon Echo speaker to be less than 2 MB in size, running on a 1 GHz ARM Cortex-A8 processor. The speaker uses about 50% of its CPU time for the wake word recognition process. In addition to the wake word, the model also detects a stop signal ("Stop") to interrupt the currently running request. Especially when used in environments with ambient noise from external sources such as TVs, a low false acceptance and false rejection rate is much harder to achieve for these systems [143].

The device will only transmit data to the respective server *after* the wake word has been recognized locally. Hence, activating the wake word by an accidental trigger will lead to the upload of potentially sensitive and private audio data, and should, therefore, be avoided as far as possible.

In some cases, a speaker misinterprets another word or sound as its wake word. If the misinterpreted word is unrelated to the configured wake word, we refer to this event as an *accidental trigger*. To limit the consequences of such false wakes, vendors started to augment the local wake word recognition with a *cloud-based wake word verification*. We describe this mechanism in more detail in Section 4.1.3.

4.1.2 Voice Profiles and Sensitivity

Voice profiles, also referred to as a "Voice Match" or "Recognize My Voice" feature, are a convenience component of modern voice assistants [7]. The requisite voice training was introduced with iOS 9 (2015), and Android 8 (2017) to build context around questions and deliver personalized results. On smartphones, a voice profile helps to recognize the user better [8]. Vendors explain that without a profile, queries are simply considered to be coming from guests and thus will not include personal results [157].

In contrast to voice assistants on phones, smart speakers are intended to be activated by third parties, such as friends and visitors. Thus, voice profiles do not influence whether a smart speaker is activated or not when the wake word is recognized. In shared, multi-user environments, voice profiles enable voice assistants to tell users apart and deliver personalized search results, music playlists, and communication. The feature is also not meant for security, as a similar voice or recording can trick the system [158]. In our experiments, voice profiles were not enabled or used.

In April 2020, Google introduced a new feature that allows users to adjust the wake word's responsiveness to limit the number of accidental activations [159]. In our experiments, we used the "Default" sensitivity.

4.1.3 Cloud-Based Wake Word Verification

Next, we focus on cloud-based wake word verification that vendors deploy to prevent or recover from accidental triggers.

The local speech recognition engine is limited by the speaker's resources. Thus, in May 2017, Amazon deployed a two-stage system [160], where a low-power ASR on the Echo is supported by a more powerful ASR engine in the cloud.

Accordingly, accidental triggers can be divided into two categories: (i) *local triggers* that overcome the local classifier, but get rejected by the cloud-based ASR engine, and (ii) *local + cloud triggers* that overcome both. While a local trigger switches the LED indicator on, a subsequent question "*{accidental local trigger}*, will it rain today?" will not be answered. In cases where the cloud does not confirm the wake word's presence, it sends a command to the Echo to stop the audio stream. Surprisingly, the entire process from the local recognition of the wake word to the moment where Echo stops the stream and switches off the LED indicator only takes about $1-2$ seconds. In our tests, we observe that during this process, Echo uploads at least $1-2$ seconds of voice data, approx. 0.5 seconds of audio before the detected wake word occurs, plus the time required to utter the wake word (approx. another second). In cases where the cloud-based ASR system also detects the wake word's presence, the accidental trigger can easily result in the upload of 10 or more seconds of voice data. During

Table 4.1: Smart Speaker Privacy Settings

	Voice Recordings				Local
Vendor	**Opt-Out**	**Retention**	**Delete**	**Report**	**Trigger**
Amazon	Yes	3, 18 months	A, R, I	Yes	Yes
Apple	Yes	6, 24 months	A	N/A	N/A
Google	Yes	3, 18 months	A, R, I	No	Yes
Microsoft	Yes*	Unspecified	A, I	No	No

*Cannot speak to Cortana anymore; A=All, R=Range, I=Individual.

our experiments, we found that all major smart speaker vendors use a cloud-based verification system, including Amazon, Apple, Google, and Microsoft.

4.1.4 Smart Speaker Privacy Settings

To learn more about how vendors handle their users' data, we requested the voice assistant interaction history from Amazon, Apple, Google, and Microsoft using their respective web forms. Among the tested vendors, Apple is the only manufacturer that does not provide access to the voice data but allows users to request their complete deletion.

In Table 4.1, we analyze whether a user is able to opt-out of the automatic storing of their voice data, how long the recordings will be retained, the possibility to request the deletion of the recordings, and whether recordings can be reported as problematic. Furthermore, we checked if false activations through accidental triggers, i. e., local triggers, are visible to the user ("Audio was not intended for Alexa"). Apple reports storing the voice recordings using a device-generated random identifier for up to 24 months but promises to disassociate the recordings from related request data (location, contact details, and app data) [161], after six months. In contrast, customers of Amazon and Google can choose between two different voice data retention options. According to Google, the two time frames of 3 and 18 months are owed to *recency* and *seasonality* [162]. Microsoft's retention policy is more vague, but they promise to comply with legal obligations and to only store the voice data "as long as necessary."

Table 4.2: Evaluated Smart Speakers

ID	Assistant, Release	Wake Word(s)	Language†	Smart Speaker
VA1	Amazon: Alexa, 2014	*Alexa*	en_us, de_de	Amazon: Echo Dot (v3)
VA2	Amazon: Alexa, 2014	*Computer*	en_us, de_de	Amazon: Echo Dot (v3)
VA3	Amazon: Alexa, 2014	*Echo*	en_us, de_de	Amazon: Echo Dot (v3)
VA4	Amazon: Alexa, 2014	*Amazon*	en_us, de_de	Amazon: Echo Dot (v3)
VA5	Google: Assistant, 2012	*OK/Hey Google*	en_us, de_de	Google: Home Mini
VA6	Apple: Siri, 2011	*Hey Siri*	en_us, de_de	Apple: HomePod
VA7	Microsoft: Cortana, 2014	*Hey/- Cortana*	en_us	Harman Kardon: Invoke
VA8	Xiaomi: Xiao AI, 2017	*Xio ài tóngxué*	zh_cn	Xiaomi: Mi AI Speaker
VA9	Tencent: Xiaowei, 2017	*Jisì'èr líng*	zh_cn	Tencent: Tngtng TS-T1
VA10	Baidu: DuerOS, 2015	*Xio dù xio dù*	zh_cn	Baidu: NV6101 (1C)
VA11	SoundHound: Houndify, 2015	*Hallo/Hey/Hi Magenta*	de_de	Telekom: Magenta Speaker

†: In our experiments, we only considered English (US), German (DE), and Standard Chinese (ZH).

4.2 Evaluation Setup

In this section, we describe our evaluated smart speakers and the datasets we used for our measurement study.

4.2.1 Evaluated Smart Speakers

In our experiments, we evaluate 11 smart speakers as listed in Table 4.2. The smart speakers have been selected based on their market shares and availability [163, 164]. In the following, with the term *smart speaker*, we refer to the hardware component. At the same time, we use the term *voice assistant* to refer to cloud-assisted ASR and the conversational intelligence built into the speaker.

Since its introduction in 2014, the Amazon Echo is one of the most popular speakers. It enables users to choose between four different wake words ("Alexa," "Computer," "Echo," and "Amazon"). In our experiments, we used four Echo Dot (3rd Gen.) and configured each to a different wake word. Similarly, for the Google Assistant, we used a Home Mini speaker, which listens to the wake words "OK Google" and "Hey Google." From Apple, we evaluated a HomePod speaker with "Hey Siri" as its wake word. To test Microsoft's Cortana, we bought the official Invoke smart speaker developed by Harman Kardon that recognizes "Cortana" and "Hey Cortana."

Moreover, we expanded the set by including non-English (US) speaking assistants from Europe and Asia. We bought three Standard Chinese (ZH) and one German (DE) speaking smart speaker. The Xiaomi speaker listens to "Xio ài tóngxué" (), which literately translates to "little classmate." The Tencent speaker listens to "Jisì'èr líng" (), which literately translates to the digit sequence 9-4-2-0. The wake word is a phonetic replacement of "Jiùshì ài n," which translates to "just love you." The Baidu speaker listens to "Xio dù xio dù" (), which literately translates to "small degree," but is related to the smart device product line Xiaodu (little "du" as in Bai*du*). Finally, we ordered the Magenta Speaker from the German telecommunications operator Deutsche Telekom, which listens to "Hallo," "Hey," and "Hi Magenta." In this case, magenta refers to a product line and also represents the company's primary brand color. Deutsche Telekom has not developed the voice assistant in-house. Instead, they chose to integrate a third-party white-label solution developed by SoundHound [165]. While the speaker also allows accessing Amazon Alexa, we have not enabled this feature for our measurements. The Magenta Speaker is technically identical to the Djingo speaker [166], which was co-developed by the French operator Orange.

4.2.2 Evaluated Datasets

In the following, we provide an overview of the datasets used to evaluate the prevalence of accidental triggers. We included media to resemble content that is likely played in a typical US household to simulate an environment with ambient noise from external sources such as TVs [143]. Moreover, we considered professional audio datasets used by the machine learning community.

TV Shows. The first category of media is TV shows. We considered a variety of different genres to be most representative. Our list comprises popular shows from the last 10 years and includes animated series and a family sitcom, a fantasy drama, and a political thriller. Our English (US) TV show dataset includes *Game of Thrones*, *House of Cards*, *Modern Family*, *New Girl*, and *The Simpsons*.

News. The second category is newscasts. As newscasts tend to be repetitive, we used one broadcast per day and television network only. The analyzed time frame covers news broadcast between August and October 2019. Our English (US) newscasts dataset includes *ABC World News*, *CBS Evening News*, *NBC Nightly News*, and *PBS NewsHour*.

Professional Datasets. The third category is professional audio datasets. Due to the costly process of collecting appropriate training datasets and the accessibility of extensive and well-analyzed datasets, we considered professional audio datasets commonly used by the speech recognition community.

- *LibriSpeech* [167]: An audio dataset created by volunteers who read and record public domain texts to create audiobooks. It contains $1,000$ hours of speech. The corpus has been built in 2015 and is publicly available; it is a widely used benchmark for automatic speech recognition.

- *Mozilla Common Voice* [168]: The dataset is based on an ongoing crowdsourcing project headed by Mozilla to create a free speech database. At the time of writing, the project includes a collection of 48 languages. Our English (US) version of the dataset contains $1,200$ hours of speech and has been downloaded in August 2019. As neither the environment nor the equipment for the audio recordings is controlled, the quality of the recordings differs widely.

- *Wall Street Journal* [169]: A corpus developed to support research on large-vocabulary, continuous speech recognition systems containing read English text. The dataset was recorded in 1993 in a controlled environment and comprises 400 hours of speech.

- *CHiME*: The *Computational Hearing in Multisource Environment* (CHiME) dataset is intended to train models to recognize speech from recordings made by distant microphones in noisy environments. The *5th* CHiME challenge dataset includes recordings from group dinners of four participants each, with two acting as hosts and two as guests [170]. Audio signals were recorded at 20 parties, each in a different home, via six Kinect microphone arrays and four binaural microphone pairs. This dataset thus provides multi-channel recordings of highly realistic, distant-talking speech with natural background noise. In total, the dataset consists of 50 hours of recording time.

Noise. We used noise recordings as a special category to test the sensitivity of the voice assistants against audio data other than speech. For this purpose, we used the noise partition of the *MUSAN* dataset [171], containing approximately 6 hours of many kinds of environmental noise (excluding speech and music).

Non-English Media. To test for linguistic differences, e. g., biases between different languages, we tested one Standard Chinese (ZH) and four German (DE) TV shows. We analyzed the Chinese TV show *All Is Well* and the German-dubbed version of the TV show *Modern Family* for easy comparison. Additionally, we tested the German-dubbed version of *The Big Bang Theory*, as well as *Polizeiruf 110* and *Tatort* as examples for undubbed German TV shows. Moreover, we evaluated three shorter (12 hours each) samples of the Chinese newscast *CCTV Xinwen Lianbo* and the German newscasts *ARD Tagesschau* and *ZDF Heute Journal*.

Female vs. Male Speakers. To explore potential gender biases in accidental triggers of voice assistants, we also included two sets of randomly chosen voice data from the *LibriSpeech* dataset. Every set consisted of a female and a male 24-hour sample. Every sample was built from multiple 20-minute sequences, which themselves were made of 100 different 12-seconds audio snippets.

4.3 Prevalence of Accidental Triggers

Based on the datasets described above, we now explore the prevalence of accidental triggers in various media such as TV shows, newscasts, and professional audio datasets.

4.3.1 Approach

We start by describing our technical setup to measure the prevalence of accidental triggers across 11 smart speakers (cf. Section 4.2.1) using 24-hour samples of various datasets (cf. Section 4.2.2). The basic idea is to simulate a common living room-like

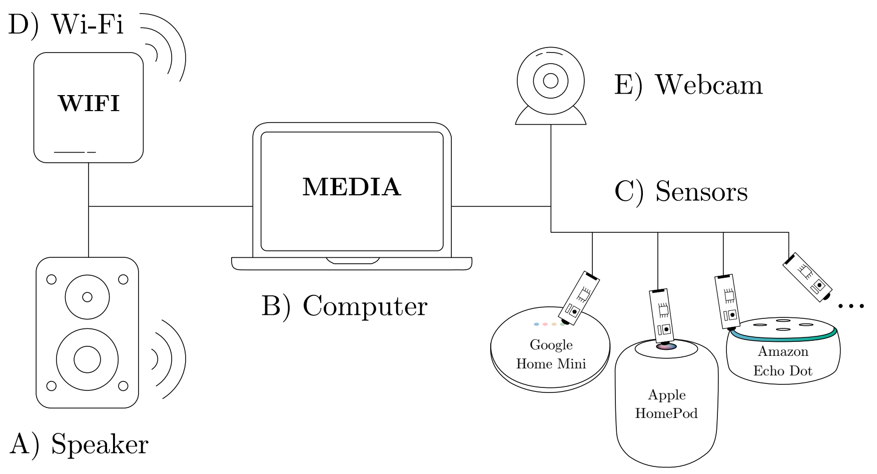


Figure 4.1: Setup: A loudspeaker (A) is playing media files from a computer (B). The LED activity indicators of a group of smart speakers are monitored using light sensors (C). All speakers are connected to the Internet over Wi-Fi (D). A webcam (E) is used to record a video of each measurement.

scenario, where a smart speaker is in close proximity to an external audio source like a TV [143, 172].

Measurement Setup

In the following, we describe the used hard- and software used to measure accidental triggers automatically.

Hardware. The measurement setup consists of five components, as depicted in Figure 4.1. To rule out any external interference, all experiments are conducted in a sound-proof chamber. We positioned 11 smart speakers at a distance of approx. 1 meter to a loudspeaker (A) and play media files from a computer (B). To detect any activity of the smart speakers, we attach photoresistors (C) (i. e., light sensors) on the LED activity indicator of each speaker, as one can see in Figure 4.2. In the case of any voice assistant activity, the light sensor detects the quick change in brightness and emits a signal to the computer (B). To prevent interference from external light sources, the photoresistors are covered by a small snippet of reusable adhesive tape. All smart speakers are connected to the Internet using a WiFi network (D). During all measurements, we record network traces using `tcpdump` to be able to analyze their

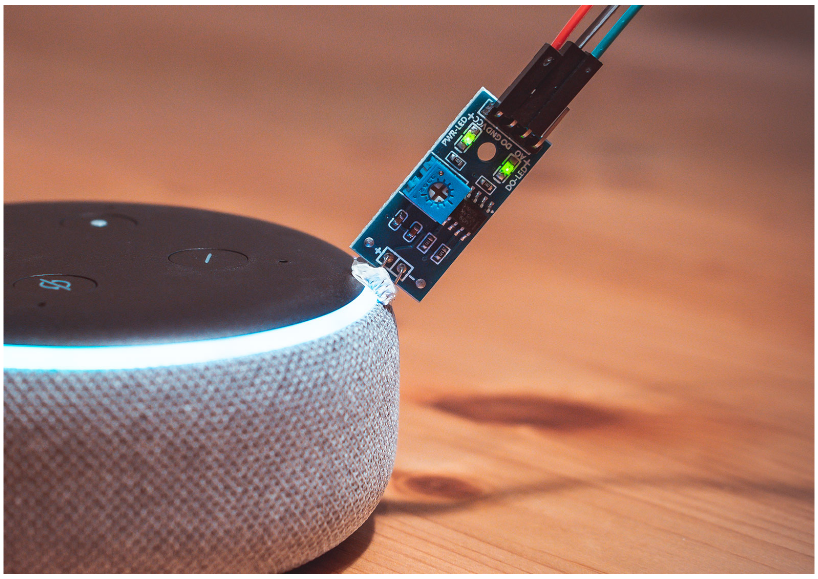


Figure 4.2: Photoresistor attached to the LED indicator of a smart speaker. The sensitivity of the sensor can be adjusted via a potentiometer. Any activity is recognized and logged.

activity on a network level. To verify the measurement results, we record a video of each measurement via a webcam with a built-in microphone (E). The entire setup is connected to a network-controllable power socket that we use to power cycle the speakers in case of failures or non-responsiveness.

Software. To verify the functionality and responsiveness of the measurement setup, we periodically play a test signal, which consists of the wake word (e. g., “Alexa”) and the stop word (e. g., “Stop”) of each voice assistant (in its configured language) and a small pause between them. Overall, the test signal for all 11 speakers is approximately 2m 30s long. During the measurements, we verify that each voice assistant triggers to its respective test signal. In the case of no response, multiple or prolonged responses, all voice assistants are automatically rebooted and rechecked. As a side effect, the test signal ensures that each assistant stops any previous activity (like playing music or telling a joke) that might have been accidentally triggered by a previous measurement run. Using this setup, we obtain a highly reliable and fully automated accidental trigger search system.

Trigger Detection

The process of measuring the prevalence of accidental triggers consists of three parts, as depicted in Figure 4.3. First, a 24-hour search is executed twice per dataset. Second, a ten-fold verification of a *potential* trigger is done to confirm the existence of the trigger and measure its reproducibility. Third, a manual classification of *verified* triggers is performed to ensure the absence of the wake word or related words. In the following, we describe these steps in more detail.

I. Search: In a first step, we prepare a 24-hour audio sample consisting of multiple episodes/broadcasts of approximately 20 minutes each (with slightly different lengths depending on the source material) for each of the datasets introduced in Section 4.2.2. We play each of the 24-hour samples twice and log any smart speaker LED activity as an indicator for a potential trigger. The logfile includes a timestamp, the currently played media, the playback progress in seconds, and the triggered smart speaker's name.

We played each audio file twice due to some changes in results that were observed when we played the same sample multiple times. These changes do not come as a great surprise, due to different components that we cannot control, e. g., the internal framing of the recorded audio. Therefore, each time the same audio file is played, the system will get a slightly different signal, with slightly shifted windows and possibly small changes in the additive noise that cannot be fully prevented but is (strongly) attenuated in our test environment. Also, there may be further indeterminacies up in the chain of trigger processing, as was also noted by others [144, 148].

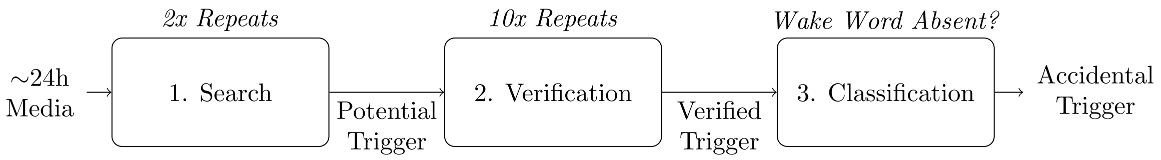


Figure 4.3: Trigger detection workflow: Every approx. 24-hour dataset is played twice. Subsequently, the existence of every *potential* trigger is confirmed. Finally, every *verified* trigger is classified as *accidental*, if the wake word or a related word is not present in the identified scene.

II. Verification: In a second step, we extract a list of *potential* triggers from the logfile and verify these triggers by replaying a 10-second snippet containing the identified scene. From the potential trigger location within the media, i. e., the playback progress when the trigger occurred, we rewind 7 seconds and replay the scene until 3 seconds after the documented trigger location. This playback is repeated ten times to confirm the existence and to measure the reproducibility of the trigger.

III. Classification: In a third step, every verified trigger is classified by reviewing a 30-second snippet of the webcam recording at the time of the trigger. Here, two independently working reviewers need to confirm the accidental trigger by verifying the correct wake word's absence. If a trigger is caused by the respective wake word or a related word such as Alexander ("Alexa"), computerized ("Computer"), echoing ("Echo"), Amazonian ("Amazon"), etc., we discard the trigger and exclude it from further analysis. Where available, the analysis is assisted by the transcriptions/subtitles of the respective dataset.

To determine the approximate distribution between *local* and *cloud*-based triggers, we expand our classification step. Instead of only determining the mere presence of the wake word or a related word, two members of our team also classify the triggers into local or local + cloud triggers. As noted in Section 4.1.4, not all smart speaker vendors provide access or report local triggers in their voice assistant interaction history. Thus, we use the internal processes, especially the LED timings and patterns, to classify triggers. The heuristic for that classification is based on the time the LED indicator of the speaker remains on. Based on preliminary tests, where we verified if the triggers passed the cloud model, we choose a threshold of 2 seconds of speaker activity to classify the trigger as local + cloud. Moreover, we use voice responses and certain LED patterns as obvious signals for a local + cloud trigger. The inter-rater reliability between our reviewers, measured by Cohen's kappa, is $\kappa \geq 0.89$ across all evaluated datasets.

4.3.2 Results

An overview of our results can be found in Table 4.3 and Table 4.4. We report the absolute counts of observed accidental triggers and actual instances of spoken wake words.

Comparison Across Speakers. Looking at the four VA1-4 Amazon Echo wake words, we can see that "Amazon" (57) and "Echo" (43) trigger less often than "Alexa"

Table 4.3: Prevalence of Accidental Triggers for English wake words.

		Alexa		**Computer**		**Echo**		**Amazon**		**Ok Google**		**Hey Siri**		**Hey Cortana**	
		A	W	A	W	A	W	A	W	A	W	A	W	A	W
		en_us		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**	
TV Shows	**Time**	**31**	**0**	**31**	**6**	**18**	**2**	**38**	**2**	**3**	**0**	**2**	**0**	**94**	**0**
Game of Thrones	24h	6	-	6	-	5	-	3	-	-	-	-	-	14	-
House of Cards	24h	2	-	11	-	2	2	15	-	-	-	1	-	14	-
Modern Family	24h	6	-	9	4	4	-	12	1	1	-	1	-	23	-
New Girl	24h	4	-	5	1	4	-	6	-	2	-	-	-	29	-
The Simpsons	24h	13	-	-	1	3	-	2	1	-	-	-	-	14	-
News	**Time**	**22**	**5**	**9**	**2**	**4**	**4**	**12**	**62**	**2**	**0**	**4**	**2**	**44**	**0**
ABC World News	24h	-	-	3	-	-	-	2	9	1	-	1	-	11	-
CBS Evening News	24h	12	1	1	1	-	-	7	24	-	-	-	-	13	-
NBC Nightly News	24h	2	4	-	-	2	1	-	23	1	-	2	2	6	-
PBS NewsHour	24h	8	-	5	1	2	3	3	6	-	-	1	-	14	-
Professional	**Time**	**46**	**1**	**37**	**32**	**21**	**3**	**7**	**1**	**11**	**0**	**2**	**0**	**59**	**0**
LibriSpeech	24h	14	-	9	-	6	2	5	-	-	-	-	-	17	-
Moz. CommonVoice	24h	10	1	21	5	14	1	2	1	11	-	2	-	18	-
WallStreetJournal	24h	22	-	7	27	1	-	-	-	-	-	-	-	24	-
CHiME	24h	1	-	3	3	-	-	10	2	7	-	1	-	1	-
Sum	13d	**100**	**6**	**80**	**43**	**43**	**9**	**67**	**67**	**23**	**0**	**9**	**2**	**198**	**0**

A: Accidental triggers; *W*: Wake word said; Gray cells: Mismatch between played audio and wake word model language.

(99) and "Computer" (77). Moreover, we observe that the VA5 Google Home (16) and the VA6 Apple HomePod (8) seem to be the most robust speakers of all English (US) speakers across all played datasets, and we discuss potential reasons for that in Section 4.5.1. Another noteworthy observation is that VA7 Microsoft Cortana triggered far more often (197) than the other speakers across all kinds of audio data.

From a qualitative perspective, the identified triggers are often confusions with similar-sounding words or sequences, such as, "a lesson" (Alexa), "commuter" (Computer), "OK, cool" (OK Google), "a city" (Hey Siri). Another category are proper names that are unknown or likely infrequently included in the training data. Examples include names of persons and states such as "Peter" and "Utah" (Computer),

Table 4.4: Prevalence of Accidental Triggers and Wake Words for Chinese and German wake words.

		Xio ài tóngxué		**Jisì' èr líng**		**Xio dù xio dù**		**Hallo Magenta**	
		A	W	A	W	A	W	A	W
		zh_cn		zh_cn		zh_cn		de_de	
TV Shows	**Time**	**1**	**0**	**7**	**0**	**0**	**0**	**3**	**0**
Game of Thrones	24h	1	-	1	-	-	-	1	-
House of Cards	24h	-	-	3	-	-	-	1	-
Modern Family	24h	-	-	1	-	-	-	1	-
New Girl	24h	-	-	1	-	-	-	-	-
The Simpsons	24h	-	-	1	-	-	-	-	-
News	**Time**	**1**	**0**	**4**	**0**	**0**	**0**	**1**	**0**
ABC World News	24h	-	-	1	-	-	-	-	-
CBS Evening News	24h	-	-	1	-	-	-	1	-
NBC Nightly News	24h	-	-	2	-	-	-	-	-
PBS NewsHour	24h	1	-	-	-	-	-	-	-
Professional	**Time**	**2**	**0**	**3**	**0**	**0**	**0**	**1**	**0**
LibriSpeech	24h	-	-	-	-	-	-	-	-
Moz. CommonVoice	24h	1	-	1	-	-	-	-	-
WallStreetJournal	24h	1	-	2	-	-	-	1	-
CHiME	24h	-	-	1	-	-	-	-	-
Sum	13d	**4**	**0**	**15**	**0**	**0**	**0**	**5**	**0**

A: Accidental triggers; *W*: Wake word said; Gray cells: Mismatch between played audio and wake word model language.

"Eddard" (Echo), "Montana" (Cortana), but also uncommon old English phrases such as "Alas!" (Alexa). Finally, we observed a few cases of triggers that include fictional language (*Dothraki*) or unintelligible language (*gibberish*) and two occasions of *non-speech* accidental triggers: A ringing phone triggering "Amazon" in the TV show *New Girl* and a honk made by a car horn triggering "Alexa" in the TV show *The Simpsons*.

Comparison Across Datasets. When comparing across datasets, one must keep in mind that the total playback time differs across categories. While every dataset

(i. e., every row in the table) consisted of 24 hours of audio data, the number of datasets per category differs.

In general, we cannot observe any noteworthy differences in accidental triggers (A) across the three dataset categories. In contrast, if we have a look at the cases where the wake word was actually said (W), we see that this was very often the case for "Computer" in the professional Wall Street Journal dataset caused by an article about the computer hardware company IBM and for "Amazon" across the news datasets. In this case, the 62 instances of "Amazon" referred 13 times to the 2019 Amazon rainforest wildfires and 49 times to the company.

If we look at the professional datasets, the number of triggers is within the same range or even increases compared to TV shows and news. As such, we have not found a speaker that triggered less often, because it might have been specifically trained on one of the professional datasets. In contrast to the other professional audio datasets, the CHiME dataset consists of recordings of group dinner scenarios resulting in comparatively less spoken words, explaining the overall lower number of accidental activations. Not presented in Table 4.3 or Table 4.4 is the *MUSAN* noise dataset, because we have not observed any triggers across the different speakers. This suggests that accidental triggers are less likely to occur for non-speech audio signals.

Comparison Between Local and Cloud-Based Triggers. An overview of the distribution can be seen in Figure 4.4. Depending on the wake word, we find that the cloud ASR engine also misrecognizes about half of our accidental triggers. Fortunately for Cortana, only a small number of triggers (8 out of 197) are able to trick Microsoft's cloud verification.

Comparison Between Female and Male Speakers.. We performed an experiment designed to study a potential model bias in smart speakers, a common problem for machine learning systems [173, 174, 175]. Fortunately, across our tested datasets, we cannot find any noteworthy difference in the number of accidental triggers for female and male speakers; no bias can be observed in our experiments. The detailed numbers are shown in Table 4.5.

Comparison Across Languages. Somewhat expected is the result that the three Chinese and the German smart speaker do not trigger very often on English (US)

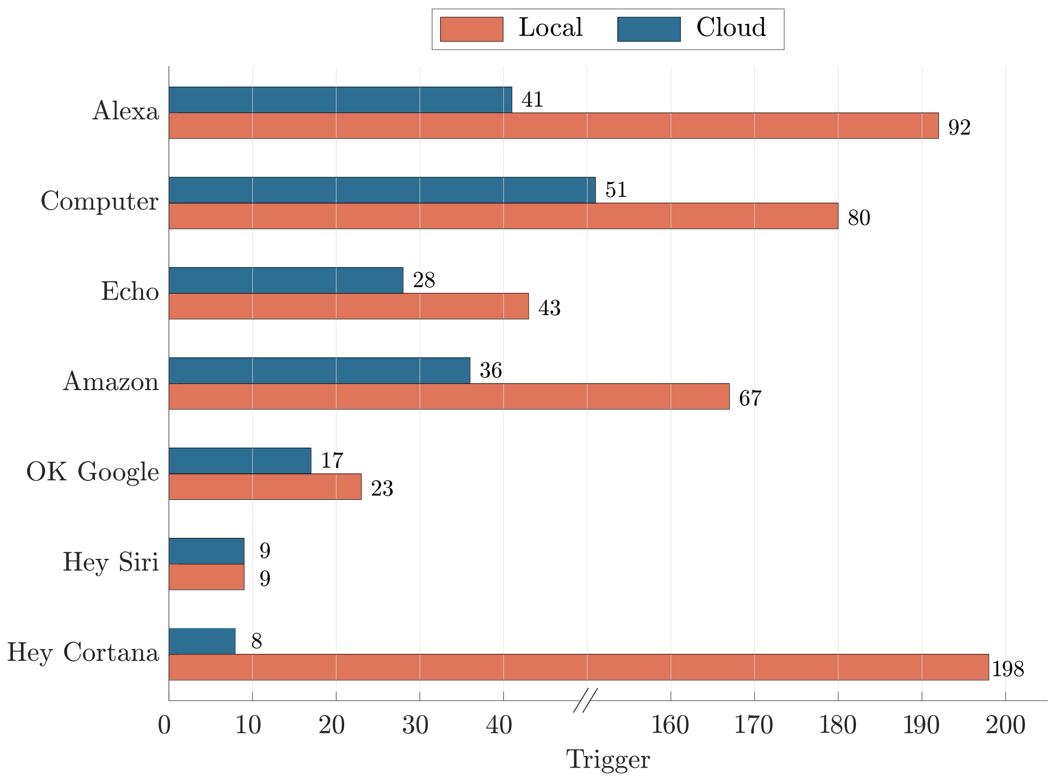


Figure 4.4: The number of accidental triggers that are incorrectly recognized by the local and the cloud-based ASR engine. Local triggers are triggers that are recognized as the wake word by the local model only. Cloud triggers are recognized by both the local and the cloud model.

content (cf. right part of Table 4.3 and Table 4.4). In Table 4.6, we report the results for the differences across languages to explore another potential model bias of the evaluated systems. Even though we only tested a small number of datasets per language, the number of triggers of VA5 Google and VA6 Apple is very low and comparable to their English performance. Given the fact that we played the very same episodes of the TV show *Modern Family* in English (US) and German, we find the wake word "Computer" to be more resistant to accidental triggers in German (1) than in English (9). A similar but less pronounced behavior can be seen with "Alexa." Moreover, we found that "big brother" in Standard Chinese dàg () is often confused with the wake word "Echo", which is hence not the best wake word choice for this language. Similarly, the German words "Am Sonntag" ("On Sunday"), with a high prevalence notably in weather forecasts, are likely to be confused with "Amazon."

Table 4.5: Differences Between Female and Male Speakers.

		Alexa		**Computer**		**Echo**		**Amazon**		**Ok Google**		**Hey Siri**		**Hey Cortana**	
		A	W	A	W	A	W	A	W	A	W	A	W	A	W
		en_us		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**	
Female	**Time**	**31**	**3**	**9**	**0**	**4**	**6**	**10**	**0**	**0**	**0**	**0**	**0**	**41**	**0**
LibriSpeech I (F)	24h	9	2	8	-	2	4	4	-	-	-	-	-	19	-
LibriSpeech II (F)	24h	22	1	1	-	2	2	6	-	-	-	-	-	22	-
Male	**Time**	**33**	**0**	**8**	**0**	**0**	**8**	**8**	**2**	**1**	**0**	**0**	**0**	**46**	**0**
LibriSpeech I (M)	24h	19	-	3	-	-	4	5	2	-	-	-	-	20	-
LibriSpeech II (M)	24h	14	-	5	-	-	4	3	-	1	-	-	-	26	-

		Xio ài tóngxué		**Jisì' èr líng**		**Xio dù xio dù**		**Hallo Magenta**	
		A	W	A	W	A	W	A	W
		zh_cn		zh_cn		zh_cn		de_de	
Female	**Time**	**1**	**0**	**1**	**0**	**0**	**0**	**2**	**0**
LibriSpeech I (F)	24h	-	-	1	-	-	-	1	-
LibriSpeech II (F)	24h	1	-	-	-	-	-	1	-
Male	**Time**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
LibriSpeech I (M)	24h	1	-	-	-	-	-	-	-
LibriSpeech II (M)	24h	-	-	-	-	-	-	-	-

A: Accidental triggers; *W*: Wake word said; Gray cells: Mismatch between played audio and wake word model language.

4.3.3 Reproducibility

During the verification step of our accidental trigger search, we replayed every trigger 10 times to measure its reproducibility. This experiment is designed based on the insight that accidental triggers likely represent samples near the decision thresholds of the machine learning model. Furthermore, we cannot control all potential parameters during the empirical experiments, and thus we want to study if, and to which extent, a trigger is actually repeatable.

Table 4.6: Differences in Languages.

		Alexa		**Computer**		**Echo**		**Amazon**		**Ok Google**		**Hey Siri**		**Hey Cortana**	
		A	W	A	W	A	W	A	W	A	W	A	W	A	W
English	**Time**	**en_us**		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**		**en_us**	
Modern Family	24h	6	-	9	4	4	-	12	1	1	-	1	-	23	-
German	**Time**	**de_de**		**de_de**		**de_de**		**de_de**		**de_de**		**de_de**		en_us	
Modern Family	24h	1	1	1	13	3	-	13	1	2	-	2	-	17	-
Big Bang Theory	24h	-	-	1	9	9	-	3	2	2	1	1	1	12	-
Polizeiruf 110	24h	3	-	4	7	3	-	13	-	-	-	-	-	18	-
Tatort	24h	-	-	-	8	4	-	15	1	2	-	-	-	6	-
ARD Tagesschau	12h	3	-	1	1	-	-	10	13	1	-	-	1	29	-
ZDF Heute Journal	12h	-	-	-	4	-	-	5	3	-	-	-	-	8	-
Standard Chinese	**Time**	en_us		en_us		en_us		en_us		en_us		en_us		en_us	
All Is Well	24h	1	-	1	-	9	-	6	-	-	-	-	-	28	-
CCTV X. Lianbo	12h	3	-	1	-	1	-	-	-	-	-	1	-	38	-

		Xio ài tóngxué		**Jisì' èr líng**		**Xio dù xio dù**		**Hallo Magenta**	
		A	W	A	W	A	W	A	W
English	**Time**	zh_cn		zh_cn		zh_cn		de_de	
Modern Family	24h	-	-	1	-	-	-	1	-
German	**Time**	zh_cn		zh_cn		zh_cn		**de_de**	
Modern Family	24h	-	-	1	-	-	-	-	-
Big Bang Theory	24h	-	-	-	-	-	-	1	-
Polizeiruf 110	24h	-	-	-	-	-	-	-	-
Tatort	24h	-	-	-	-	-	-	-	-
ARD Tagesschau	12h	1	-	-	-	-	-	-	-
ZDF Heute Journal	12h	-	-	1	-	-	-	-	-
Standard Chinese	**Time**	**zh_cn**		**zh_cn**		**zh_cn**		de_de	
All Is Well	24h	-	-	-	-	2	-	-	-
CCTV X. Lianbo	12h	-	-	3	-	-	-	-	-

A: Accidental triggers; *W*: Wake word said; Gray cells: Mismatch between played audio and wake word model language.

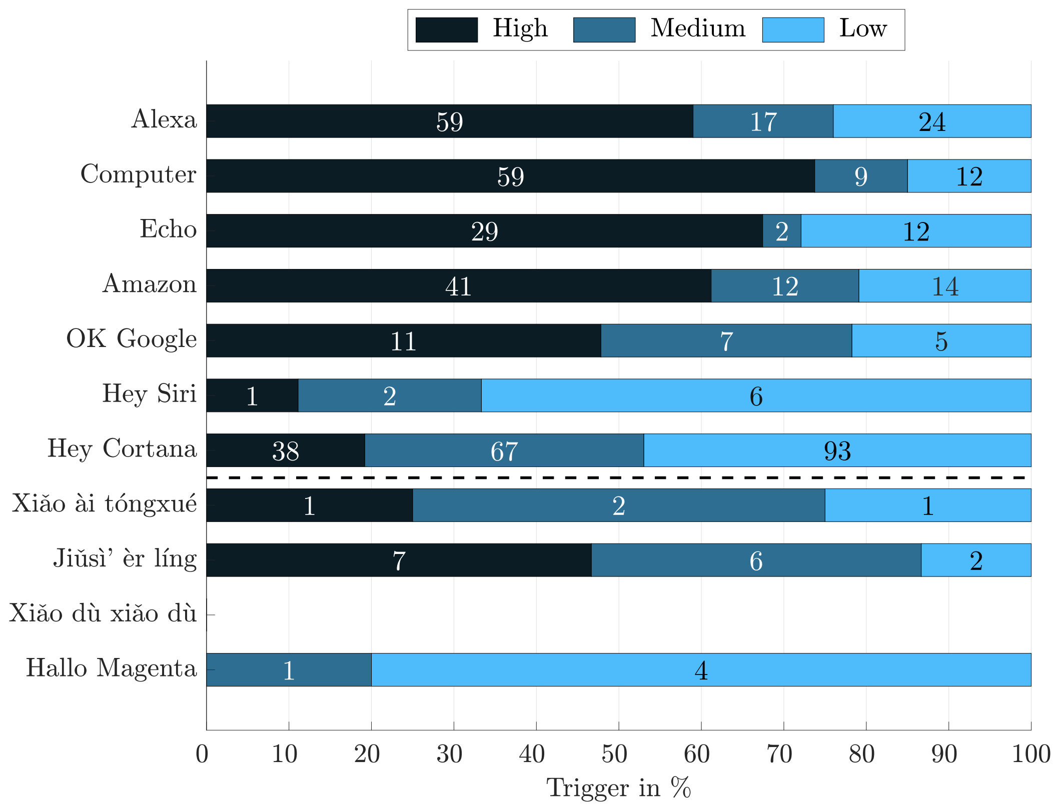


Figure 4.5: Accidental trigger reproducibility. Note that the four speakers below the dashed line do not use wake words in English (US); "Xio dù xio dù" did not have any triggers.

We binned the triggers into three categories: *low*, *medium*, and *high* reproducibility. Audio snippets that triggered the respective assistant 1–3 times are considered as *low*, 4–7 times as *medium*, and 8–10 times as *high*. In Figure 4.5, we visualize these results. We observe that across the Amazon and Google speakers, around 75 % of our found triggers are medium to highly reproducible. This indicates that most of the identified triggers are indeed reliable and represent examples where the wake word recognition fails. For the Apple and Microsoft speakers, the triggers are less reliable in our experiments. One caveat of the results is that the Chinese and German speakers' data are rather sparse and do not allow a meaningful observation and interpretation of the results.

4.4 Crafting Accidental Triggers

The previous experiments raise the question of whether it is possible to specifically forge accidental triggers in a systematic and fully automated way. We hypothesize that words with a similar pronunciation as the wake word, i. e., based on similar phones (the linguistically smallest unit of speech production) are promising candidates. In this section, we are interested in crafting accidental triggers that are likely caused by the wake word's phonetic similarity.

4.4.1 Speech Synthesis

To systematically test candidates, we utilize Google's TTS API. To provide a variety across different voices and genders, we synthesize 10 different TTS versions, one for each US English voice in the TTS API. Four of the voices are standard TTS voices; six are Google *WaveNet* voices [176]. In both cases, the female-male-split is half and half.

Note that some words have more than one possible pronunciation (e. g., T AH M **EY** T OW vs. T AH M **AA** T OW). Unfortunately, we cannot control how Google's TTS service pronounces these words. Nevertheless, we are able to show how, in principle, one can find accidental triggers, and we use 10 different voices for the synthesis to limit this effect.

4.4.2 Levenshtein Distance

To compare the wake words with other words, we use the Fisher corpus [177] version of the Carnegie Mellon University pronouncing dictionary [178], an open-source pronunciation dictionary for North American English listing the phone sequences of more than $130,000$ words. We propose two versions of a weighted phone-based Levenshtein distance [100] to measure the distance of the phonetic description of a candidate to the phonetic description of the respective wake word in order to find potential triggers in a fully automated way. Using dynamic programming, we can compute the minimal distance $\mathcal{L}$ (under an optimal alignment of the wake word and the trigger word). Formally, we calculate

$$\mathcal{L} = \frac{s \cdot S + d \cdot D + i \cdot I}{N} \tag{4.1}$$

with the number of *substituted* phones S, *inserted* phones I, *deleted* phones D, and the total number of phones N, describing the weighted *edit distance* to transform one word into another. The parameters s, d, and i describe scale factors for the different kinds of errors.

In the following, we motivate our different scale factors: During the decoding step of the recognition pipeline, a path search through all possible phone combinations is conducted by the ASR system. In general, for the recognition, the path with the least cost is selected as the designated output of the recognition (i. e., wake word or not wake word). Considering these principles of wake word recognition, we assume that the different kinds of errors have different impacts on the wake word recognition, as e. g., utterances with deletions of relevant phones will hardly act as a wake word.

To find the optimal scale factors, we conducted a *hyperparameter search* where we tested different combinations of weights. For this purpose, we played all different TTS versions of 50,000 English words and tracked which of the voice assistants triggered at least once. In total, we were able to find 826 triggers. In a second, more advanced, version of this distance measure, we considered phone-dependent weights for the different kinds of errors. A more detailed description of this version of the distance measure is presented in Section 4.4.3.

We do not include the wakeword itself and ignore words that are pronounced like parts of the wake word (e. g., “Hay” is blocklisted for “Hey” or “computed” for “computer”). The blocklist of the wake words contains a minimum of 2 words (*Cortana*) and up to 6 words (*Computer*). For the optimization, we used a ranked-based assessment: We sorted all $50,000$ words by their distance $\mathcal{L}$ and used the rank of the triggered word with the largest distance as a metric to compare the different weighted Levenshtein distances. With this metric, we performed a grid search for s, d, i over the interval $[0, 1]$ with a step width of 0.05.

4.4.3 Phone-Dependent Weights

For a more advanced version of the weighted Levenshtein distance, we utilized information about how costly it is to substitute, delete, and insert specific phones (i. e., intuitively it should be less costly to replace one vowel with another vowel in comparison to replacing a vowel with a consonant). For this purpose, we calculated phone-dependent weights as described in the following: We used a trained ASR system and employed *forced alignment*, which is usually used during the training of an ASR system to avoid the need for a detailed alignment of the transcription to the audio file. We can use this algorithm to systematically change the phonetic description of an audio file's transcription and measure the costs of these specific changes.

To measure the impact of such changes, we distinguish between deletions, substitutions, and insertions. To assess the cost of the deletion of specific phones, we randomly draw 100 words that contain that specific phone and synthesize 10 versions of this word via Google's TTS API. We use the difference of the log-likelihood scores of the forced-alignment output with and without this specific phone for all TTS versions of the word. For example, we use the word *little* with the phonetic description L IH T AH L for the phone AH in *Alexa* and measure the score of the forced alignment algorithm for L IH T AH L and L IH T L. The difference in these two scores describes the cost of deleting the sound 'AH' in this specific context. Note that we only calculated the weights of phones that occur in the wake words. For the final weights, we use the average over all 100 words and 10 TTS versions and finally normalize the values of all averaged phone costs $\boldsymbol{d}$ to have a mean value of 1.0. The resulting deletion weights $\hat{\boldsymbol{d}}$ are shown in Figure 4.6.

Similarly, to determine the cost of all possible substitutions, we replace the phone-under-test with all other phones for all 100 words and 10 TTS versions. We followed the same approach as for the deletion costs previously and averaged and normalized the log-likelihood scores to define the final weights. The matrix of the substitution weights is shown in Figure 4.7. Therefore, the rows in the figure do not show all theoretically possible phones. Finally, we compare the scores between the original transcription and the transcription with an inserted phone for the insertion weights. These weights are also normalized to have an average value of 1.0. The insertion weights are shown in Figure 4.8. All weights are then used in Equation (4.1) along with the scale factors.

4.4.4 Cross-Validation

We performed a leave-one-out cross-validation to measure the performance of Equation (4.1) in predicting whether words are potential accidental triggers. For this purpose, we compared three different versions of Equation (4.1): a version, with all scale factors set to 1 (Unweighted), a scaled version where we optimized the scale fac-

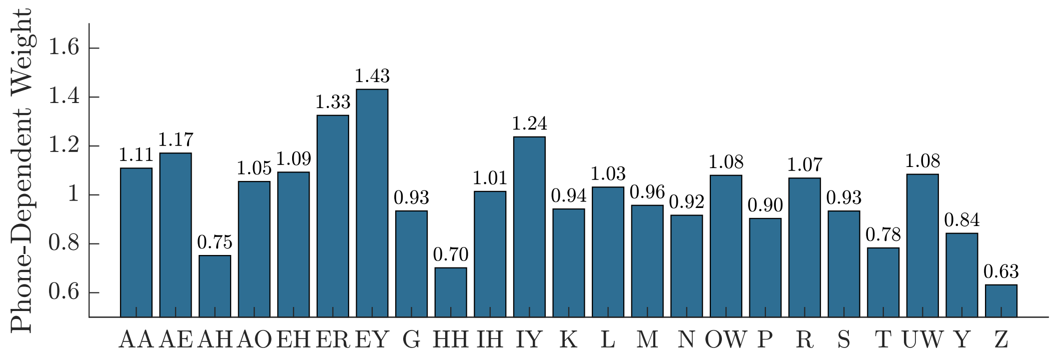


Figure 4.6: Deletion weights used for the advanced version of the weighted Levenshtein distance. The higher the value, the higher the costs if this phone is *removed*.

tors (Simple), and a version with our optimized scale factors and the phone-dependent weights (Advanced).

We have run a hyperparameter search for the simple and the advanced version of eight wake-words triggers for each fold and tested the resulting scale factors on the remaining 8 wake word. Table 4.7 shows the number of triggers we found within the 100 words with the smallest distance for all three versions of the Levenshtein distance and all wake words. Note that the distances tend to cluster words into same distances due to the fixed length of each wake word and, therefore, the same total number of phones N, especially for the unweighted and the simple version.

For cases where it is not possible to clearly determine the closest 100 words, we use all words with a smaller distance than the 100^{th} word and draw randomly out of the words with the next largest distance until we obtain a list of 100 words to ensure a fair comparison in Table 4.7.

In the third column (Total), we show the total number of words that triggered the respective wake words, out of the 50,000 words, after filtering. Note that the Google wake words had only 1 or 2 triggers in total, which is an upper bound for the top 100 results.

The different versions of the Levenshtein distance generally show better results for the simple and the advanced version compared to the unweighted version, especially for all Amazon wake words. Only for the two wake words from Microsoft, this is not the case. Nevertheless, the advanced version shows the best results on average and is, therefore, the version we use in the following experiments. Notably, for e. g.,

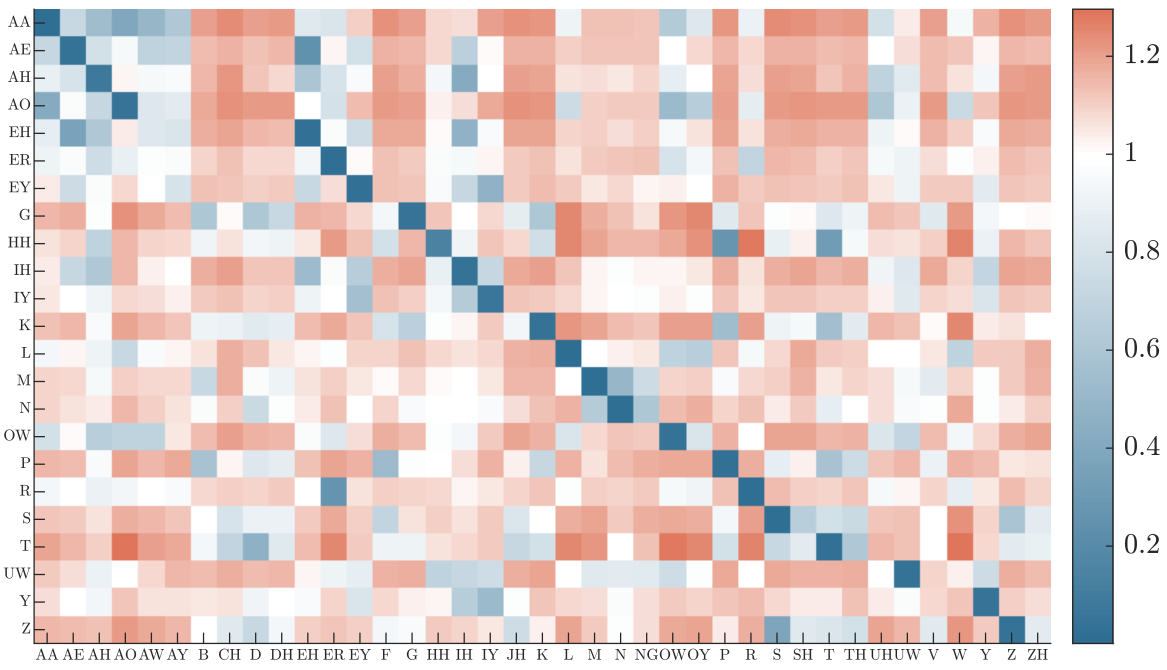


Figure 4.7: Substitution weights used for the advanced version of the weighted Levenshtein distance plotted as a matrix describing the cost to *replace* the phone in the row with a phone of the columns.

Computer, approximately one third (32/100) of the words with the smallest distance actually triggered the smart speaker and for many of the wake words, more than, or almost half of all possible triggers can be found within the 100 words with the smallest distance.

4.4.5 Performance on Real-World Data

With the optimized scale factors and weights, we evaluate the distance measure on the transcriptions of the CHiME dataset to assess the performance of the optimized distance measure on real-world voice data. For this purpose, we consider 1-, 2-, and 3-grams to also test sequences of words that occur in the CHiME transcriptions.

We perform a hyperparameter search for the advanced version of the Levenshtein distance (scale factors and phone-dependent weights) on the triggers of all 9 wake words on the data set used in Section 4.4.4. For these, the optimal scale factors are $s = 1.46$, $d = 1.30$, and $i = 0.24$, which we use in the following experiment. We select the 100 n-grams with the smallest distance to the respective wake word from

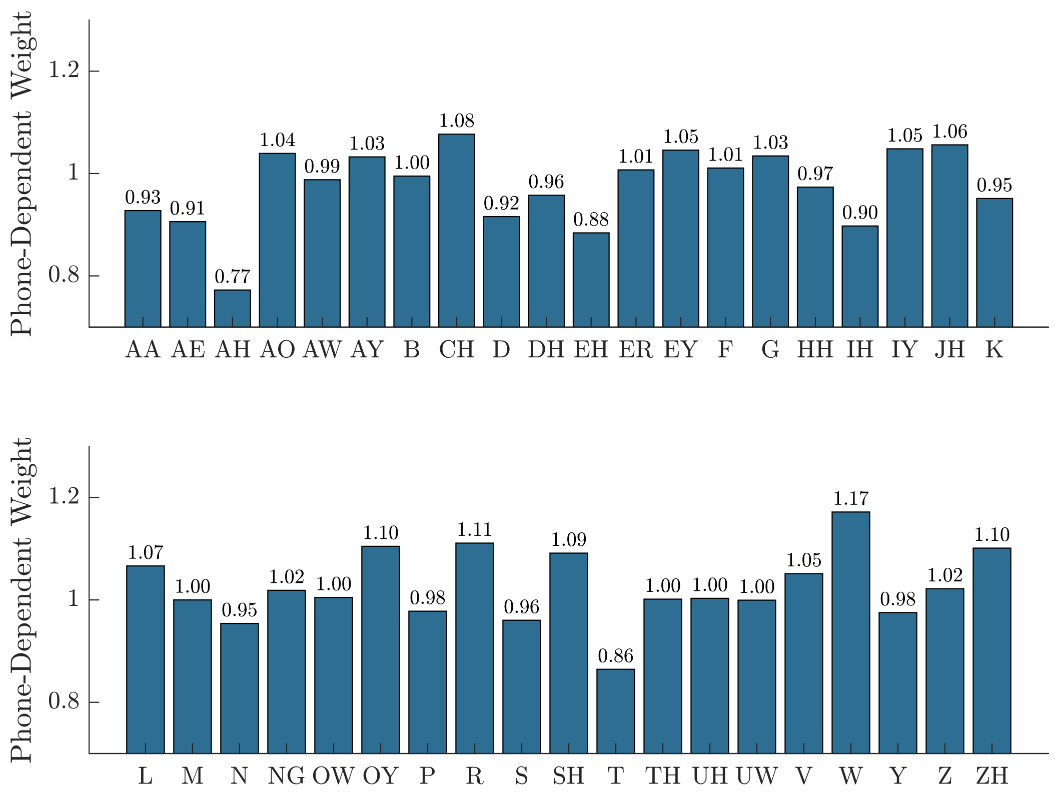


Figure 4.8: Insertion weights used for the advanced version of the weighted Levenshtein distance. The higher the value, the higher the costs if this phone is *inserted*.

all 1-, 2-, and 3-grams, in total, 300 n-grams for each wake word. All these n-grams are synthesized with Google's TTS API. We then play these crafted triggers against all smart speakers. The results of the CHiME n-grams are shown in Table 4.8. We were able to find a number of triggers for almost all of the wake words, like "fresh parmesan" for *Amazon*, "my cereal" for *Hey Siri*, and "all acts of" for *Alexa*. A manual analysis revealed that some of the crafted triggers were also found, and also caused triggers, in our previously tested real-world data. Examples include: (Alexa) "*Election* day is" in Modern Family, (Computer) "*compared to* the millions" in PBS NewsHour, and (Cortana) "*Montana*'s bag limits are" in LibriSpeech.

Table 4.7: Results of the leave-one-out cross-validation. We report the number of triggers within the 100 words with the smallest distance to the respective wake word.

ID	Wake Word	Total	Unweighted	Simple	Advanced
VA1	*Alexa*	52	9	17	24
VA2	*Computer*	75	17	21	32
VA3	*Echo*	23	4	5	12
VA4	*Amazon*	12	1	1	7
VA5a	*OK Google*	2	0	0	0
VA5b	*Hey Google*	1	0	0	0
VA6	*Hey Siri*	7	3	3	5
VA7a	*Hey Cortana*	38	9	9	6
VA7b	*Cortana*	45	13	14	10

4.5 Discussion

The results of our experiments suggest possible reasons for the differences across wake words and raise the question of why their vendors have chosen them in the first place. The trade-off between a low false acceptance and false rejection rate is hard to balance and we discuss potential measures that can help to reduce the impact of accidental triggers on the user's privacy.

4.5.1 Wake Word

Properties of Robust Wake Words. Looking at the number of words in a wake word, one would assume a clear benefit using two words. This observation is supported by the results in Table 4.7, where "Cortana" leads to more triggers than "Hey Cortana." On the contrary, the shortest wake word "Echo" has fewer triggers than "Hey Cortana," suggesting that not only the number of words (and phones) itself is important, but the average distance to common words in the respective language. These results suggest that increasing the number of words in a wake word has the same effect as increasing the distance to common words. If we consider the differences in the prevalence of accidental triggers, and that adding an additional word (e. g., "Hey")

Table 4.8: To craft realistic word *combinations*, we construct word sequences based on n-grams from the CHiME transcriptions. We report the numbers of triggers within the 100 n-grams with the smallest distance to the respective wake word.

ID	**Wake Word**	**1-gram**	**2-gram**	**3-gram**
VA1	*Alexa*	7	10	5
VA2	*Computer*	16	12	10
VA3	*Echo*	1	8	3
VA4	*Amazon*	2	11	4
VA5a	*OK Google*	0	1	0
VA5b	*Hey Google*	0	0	0
VA6	*Hey Siri*	2	2	0
VA7a	*Hey Cortana*	8	8	4
VA7b	*Cortana*	7	5	6

comes at close to no cost for the user, we recommend that vendors deploy wake words consisting of two words.

Word Selection. Amazon shared some details about why they have chosen "Alexa" as their wake word [179]: The development was inspired by the LCARS, the Star Trek computer, which is activated by saying "Computer." Moreover, they wanted a word that people do not ordinarily use in everyday life. In the end, Amazon decided on "Alexa" because it sounded unique and used soft vowels and an "x." The co-founder of Apple's voice assistant chose the name "Siri" after a co-worker in Norway [180]. Later, when Apple turned Siri from a push-to-talk into a wake word-based voice assistant, the phrase "Hey Siri" was chosen because they wanted the wake word to sound as natural as possible [8]. Based on those examples we can see that the wake word choice in practice is not always a rational, technically founded decision, but driven by other factors like marketing as in "OK Google," "Amazon," "Xio dù xio dù," or "Hallo Magenta," or based on other motivations such as in the case of "Siri" or "Computer." Another issue can arise when trying to port a wake word across languages. An example of that is the confusion of dàg (“big brother”) and “Echo” described in Section 4.3.2, and it gets even more complicated in multilingual households [181].

4.5.2 Countermeasures

Local On-Device Speech Recognition. In 2019, Google deployed an on-device speech recognizer on a smartphone that can transcribe spoken audio in real-time without an Internet connection [182, 183]. We find such an approach to be promising, as it can help to reduce the impact of accidental triggers by limiting the upload of sensitive voice data. After the local ASR detects the wake word, one can imagine a speaker that transcribes the audio input and only after being certain to have detected a user command/question, uploads the short wake word sequence for cloud verification. When both ASR engines agree about the wake word’s presence, the command/question is forwarded to the cloud in text or audio form. Coucke et al. has described a smart speaker that runs completely offline and is thus private-by-design [184].

Device-Directed Queries and Visual Cues. Amazon presented a classifier for distinguishing device-directed queries from background speech in the context of follow-up queries [185, 186]. While follow-up queries are a convenience feature, one can imagine a similar system that can reduce the number of accidental triggers. Mhaidli et al. [187] explored the feasibility of only selectively activating a voice assistant using gaze direction and voice volume level by integrating a depth-camera to recognize a user’s head orientation. While this approach constitutes a slight change in how users interact with a smart speaker, it effectively reduces the risk of accidental triggers, by requiring a direct line-of-sight between the user and the device. However, their participants also expressed privacy concerns due to the presence of the camera.

Privacy Mode and Safewords. Previous work [188] has documented the ineffectiveness of current privacy controls, such as the mute button, given the inability to use the speaker hands-free when muted. We imagine a method similar to a *safeword* as a possible workaround for this problem. For this, the speaker implements a *privacy mode* that is activated by a user saying, “Alexa, please start ignoring me,” but could, for example, also be activated based on other events such as the time of the day. In the privacy mode, the speaker disables all cloud functionality, including cloud-based wake word verification and question answering.

The speaker's normal operation is then re-enabled by a user saying, "Alexa, Alexa, Alexa." Due to the requirement to speak the somewhat lengthy safeword, accidental triggers will only happen very rarely. We imagine this privacy control to be more usable than a mute button, as the hands-free operation is still possible. As only the wake word is repeated multiple times, we think that vendors can implement this functionality using the local ASR engine.

Increased Transparency. Another option is to increase transparency and control over the retention periods and individual recordings. In particular, our experience with Microsoft's *Privacy Dashboard* made it clear that vendors need to implement features to better control, sort, filter, and delete voice recordings. Amazon's and Google's web interface already allow a user to filter interactions by date or device easily. In particular, we imagine a view that shows potential accidental triggers, e. g., because the assistant could not detect a question. Currently, accidental triggers are (intentionally) not very present, and are easy to miss in the majority of legitimate requests. If accidental triggers were to become more visible, we hope that users would start to more frequently use privacy controls such as safewords, the mute button, or to request the deletion of the last interaction via a voice command, e. g., "Hey Google, that wasn't for you." At first, integrating such a functionality seems unfavorable to vendors, but it can easily be turned into a privacy feature that can be seen as an advantage over competitors.

4.5.3 Limitations

We have neither evaluated nor explored triggers for varying rooms and acoustic environments, e. g., distances or volumes. Even if this might influence the reproducibility, this was not part of our study, as we focused on the general number of accidental triggers in a comparable setup across all experiments. This also implies that our results are somewhat tied to the hard- and software version of the evaluated smart speakers.

Our results are subject to change due to model updates for the local ASR or updates of the cloud model.

Furthermore, we are dealing with a system that is not entirely deterministic, as others already noted [144]. Accidental triggers we mark as local triggers, sometimes overcome the cloud-based recognizer and vice versa. The findings are mainly based on the English (US) language; even though we also played a limited set of German and Standard Chinese media, our results are not applicable to other languages or ASR models.

4.6 Related Work

There is an increasing amount of work focusing on the security and privacy of smart speakers that motivate and guide our research, as discussed in the following.

Malkin et al. [146] studied the privacy attitudes of 116 smart speaker users. Almost half of their respondents did not know that their voice recordings are stored in the cloud, and only a few had ever deleted any of their recordings. Moreover, they reported that their participants were particularly protective about other people's recordings, such as guests. Besides conversations that include children, financial, sexual, or medical information, *accidentally captured conversations* were named information that should automatically be screened out and not stored. Lau et al. [188] have studied privacy perceptions and concerns around smart speakers. They found an incomplete understanding of the resulting privacy risks and document problems with incidental smart speaker users. For example, they described that two of their participants used the audio logs to surveil or monitor incidental users. They noted that current privacy controls are rarely used. For example, they studied why users do not make use of the mute button on the smart speaker. Most of their participants preferred to simply unplug the device and give trust issues and the inability to use the speaker hands-free as reasons not to press the mute button.

Similarly, Abdi et al. [189] explored mental models of where smart speaker data is stored, processed, and shared. Ammari et al. [190] studied how people use their voice assistants and found users being concerned about *random activations* and documented how they deal with them.

Huang et al. [191] studied users' concerns about shared smart speakers. Their participants expressed worries regarding *voice match false positives*, unauthorized access of personal information, and the misuse of the device by unintended users such as visitors. They confirmed that users perceive external entities, such as smart speaker vendors, collecting voice recordings as a major privacy threat.

Chung et al. [147] named *unintentional voice recordings* a significant privacy threat and warned about entities with legitimate voice data access and commercial interests, as well as helpless users not in control of their voice data.

Tabassum et al. [192] studied *always-listening* voice assistants that do not require any wake word. Zeng et al. [193] studied security- and privacy-related attitudes of people living in smart homes. Their participants mentioned privacy violations and concerns, particularly around audio recordings. Recently and concurrently, Dubois et al. [148] published a paper where they played 134 hours of TV shows to measure the prevalence of accidental triggers. Their setup relied on a combination of a webcam, computer vision, and a network-traffic-based heuristic. Their work confirms our results in Section 4.3. In contrast to our work, the authors focused only on a comparatively small English TV show dataset and English-speaking smart speakers. They did not consider speakers from other countries, other languages, or other audio datasets.

4.7 Summary

We conduct a comprehensive analysis of accidental triggers in voice assistants and explore their impact on the user's privacy. We explain how current smart speakers try to limit the impact of accidental triggers using cloud-based verification systems and analyze how these systems affect users' privacy. More specifically, we automate the process of finding accidental triggers and measure their prevalence across 11 smart speakers. We describe a method to artificially craft such triggers using a pronouncing dictionary and a weighted phone-based Levenshtein distance metric that can be used to benchmark smart speakers. As the underlying problem of accidental triggers, the trade-off between a low false acceptance and false rejection rate is hard to balance we discuss countermeasures that can help to reduce the number and impact of accidental triggers.

5 | Conclusion

This thesis evaluated the robustness of speech and speaker recognition systems, including spoofing-resistant speaker recognition, adversarial examples for hybrid speech recognition systems, and an analysis of the sensitivity of popular smart speakers to accidental triggers. This chapter summarizes the main findings and presents potential directions for future work.

5.1 Summary

In Chapter 2, we showed that a speaker recognition augmented with a facial recognition can deal with various distortions added to the audio and video data. We trained an i-vector-based speaker recognition system and a face recognition based on LBP and utilized uncertainty measures to estimate an optimal weight for a combined recognition. Our approach achieved at least the same recognition rate one would get when only relying on the best single modality and notably exceeds this performance in cases where at least one modality is distorted.

Additionally, in Chapter 2 we proposed a text-dependent audio-visual spoofing detection for speaker verification. The detection mechanism utilizes CHMM to measure the transcription and synchronicity simultaneously. For its evaluation, we considered different spoofing scenarios that are known from real-world attacks. The results have shown that the system successfully recognizes different attack scenarios, where either the wrong text has been uttered, one modality is missing, or the audio and video streams do not match.

The second part of this thesis analyzed adversarial examples for hybrid speech recognition systems that exploit psychoacoustic hearing thresholds. We have shown that it is possible to hide any target transcription within virtually any audio file. Hiding the noise below the dynamic hearing thresholds makes the changes almost imperceptible. This was confirmed with two user studies, which assessed the audibility as well as the intelligibility with human listeners.

To investigate the practical implications and the real-world impact of audio adversarial examples, the attack was extended to situations where the audio signal is played via a loudspeaker. We considered the room characteristics as a random variable, utilizing RIRs during the optimization of the adversarial examples. We have shown that no prior knowledge about the attack setup is required. An attacker can calculate generic adversarial examples for varying rooms, which do not need to be tailored to a specific setup. Furthermore, it is possible to create targeted robust adversarial examples for varying audio content, even if no direct line-of-sight between the microphone and the speakers exists.

To conclude Chapter 3, we introduced a defense mechanism for hybrid speech recognition systems by replacing the acoustic model DNN with neural networks capable of uncertainty quantification. Our empirical results show that this significantly increased the robustness against targeted adversarial examples. Additionally, we have shown that entropy serves as a good measure for identifying adversarial examples. We were able to discriminate between benign and adversarial examples with a one-class classifier that detects adversarial examples as outliers in our experiments.

In the final chapter, we performed a systematic analysis of the sensitivity of popular smart speakers to accidental triggers. We automated the process of finding accidental triggers and measured their prevalence across 11 smart speakers. We described a method to artificially craft such triggers using a pronouncing dictionary and a weighted phone-based Levenshtein distance metric that can be used to benchmark smart speakers.

5.2 Future Research Directions

This thesis investigated threats against speech and speaker recognition systems, as well as proposed countermeasures and methods to quantify weaknesses to improve their robustness. Based on the results of this thesis, the following research questions are interesting directions for future work.

Advances of Deep Learning. Recent advances in the field of deep learning require new countermeasures that withstand attacks utilizing methods such as *deepfakes*, artificially created images, audio signals, and videos [194]. Deepfakes have become so astonishingly realistic that they are hard to detect even for human observers. Therefore, biometric authentication based on voice and images needs to consider attacks with such artificially created content, e. g., by detecting artifacts that are unavoidable consequences of the construction of deepfakes [195].

Privacy. Voice assistants are primarily used in smartphones or at home, where they may capture very private information. Privacy-preserving speech and speaker recognition is a neglected research direction but can be extremely important considering that the data is in general processed in the cloud. To further improve speech recognition, training data from the domain it operates in needs to be collected. Wake-word-based speech recognition systems constantly capture their environment, and accidental triggers are labeled and used to improve the system. This raises the question of how users' privacy can be preserved without losing the utility of hands-free voice assistants, for example via privacy-preserving machine learning.

In the field of machine learning, privacy-preserving mechanisms are a separate line of research. This includes *differential privacy*, which follows a more formal definition of privacy; privacy is guaranteed if an attacker cannot tell whether a specific sample is part of a training set. *Federated learning* follows a different approach. Here, the idea is to distribute the training data to separated devices, e. g., every client uses its own data, and apply collaborative learning to avoid processing the training data at a central, probably not trustworthy, instance. Both techniques have barely been applied for speech-based systems and offer a promising direction for future work.

Different Attack Vectors. Attacks against machine learning systems are not limited to adversarial examples. In general, different components of the machine learning chain are vulnerable to different kinds of attacks. Adversarial examples can be understood as attacks during the run-time of a system. An orthogonal line of research is *data poisoning* or *backdoor* attacks that aim to manipulate the training data to get a system to learn something it should not. For example, a system may learn to recognize a speed limit sign whenever a particular pattern, e. g., in the form of a sticker, is shown in the image. The stealthiness of data poisoning arises from the fact that poisoned data is hard to detect, as it is infeasible to verify all training data. Besides, the poisons themselves are not obvious; e. g., in clean-label poison attacks, the poisoned samples retain their original label.

Parameter stealing attacks try to construct a surrogate model that rebuilds the learned parameters of an existing black-box model. For this purpose, the attacker sends a request to the model under attack, and the request and the returned output are used to obtain a model that closely resembles the target model. The surrogate model can then be used for a white-box attack that transfers to the stolen black-box model. Consequently, it can never be guaranteed that a model that is deployed as a black-box remains a secret and models also need to be resistant against attacks in which they are used as a white-box model.

Data poisoning and parameter stealing attacks have primarily been shown to be applicable to systems for image with image-based inputs. If and to what degree speech and speaker recognition systems are vulnerable to such attacks has not been thoroughly investigated yet [119, 122, 196].

Different Perspectives on Adversarial Examples. Adversarial examples can be used to understand models and their limitations better. Such approaches would not aim to prevent adversarial examples completely but use them to investigate the difference between human and machine processing of real-world data. Ilyas et al. [127] have shown that adversarial perturbations are consequences of well-generalizing, yet brittle, features and that neural networks utilize —besides the features that are also used by human observers—so-called *non-robust features*. Understanding the nature of adversarial examples better enables the design of models that are easier to interpret and makes it harder to design inconspicuous attacks.

5.3 Concluding Remarks

Speech-based systems, such as voice assistants, have become more and more important in our everyday life. For example, in smart speakers, they constantly capture and analyze their surrounding environment and are used for various security- and privacy-sensitive tasks. This thesis investigated real-world threats against speech and speaker recognition systems. We have developed methods to quantify voice assistants' weaknesses and countermeasures to improve the robustness of speech and speaker recognition. Future work should investigate other attack vectors and privacy-preserving speech and speaker recognition. Additionally, existing countermeasure strategies should be reassessed to withstand future advances in the rapidly growing field of deep learning. In the meantime, it continues to be our responsibility to develop new secure speaker and speech recognition systems.

A | Over-the-Air Recording Setups

Table A.1: Microphone and speaker positions and the reverberation times for each room in Table 3.9.

	T_{60}	Microphone	Speaker	
			w/ line-of-sight	w/o line-of-sight
Lecture Room	0.80 s	$\mathbf{r} = [8.1\,\text{m}, 3.4\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [11.0\,\text{m}, 3.4\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [8.9\,\text{m}, 2.2\,\text{m}, 0.0\,\text{m}]$
Meeting Room	0.74 s	$\mathbf{r} = [3.7\,\text{m}, 5.7\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [1.8\,\text{m}, 5.7\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [3.7\,\text{m}, 4.9\,\text{m}, 0.0\,\text{m}]$
Office	0.64 s	$\mathbf{r} = [3.8\,\text{m}, 1.8\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [1.4\,\text{m}, 4.6\,\text{m}, 1.2\,\text{m}]$	$\mathbf{s} = [-0.5\,\text{m}, 2.0\,\text{m}, 1.2\,\text{m}]$

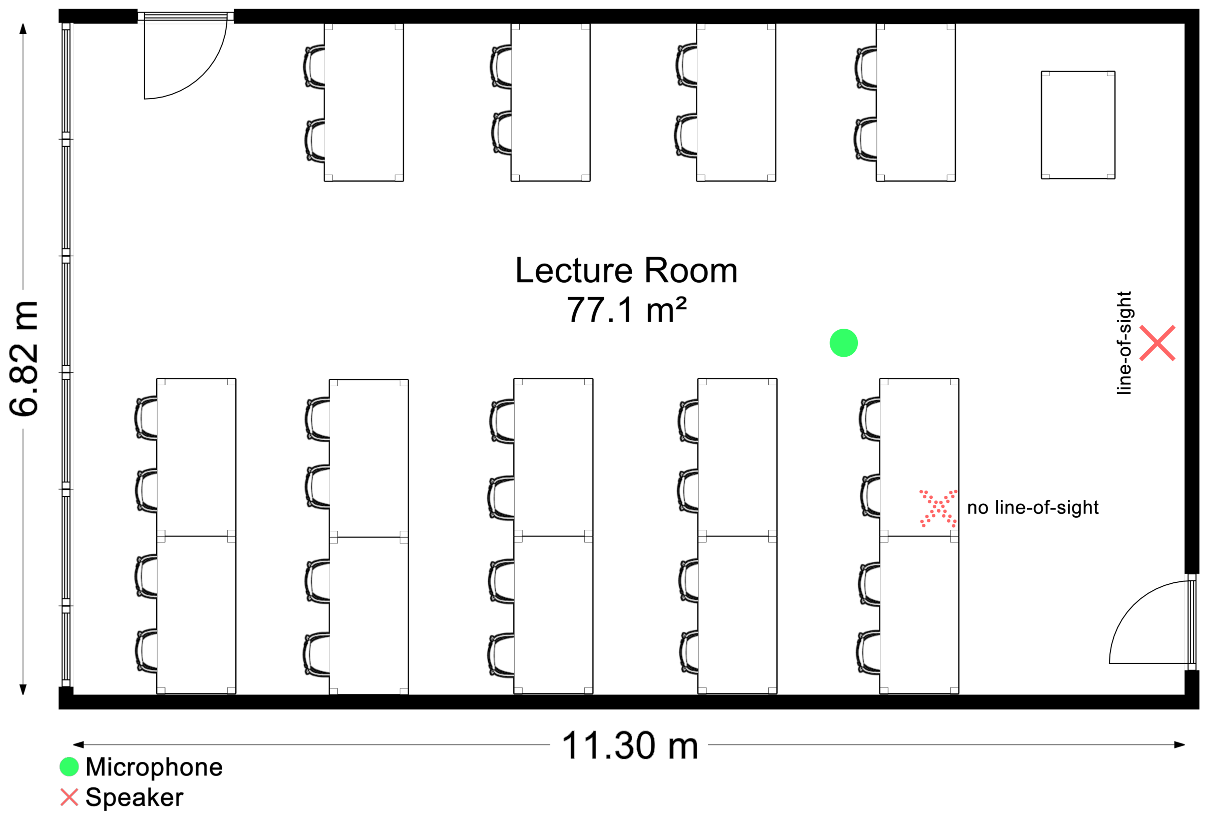


Figure A.1: Room layout of the lecture room.

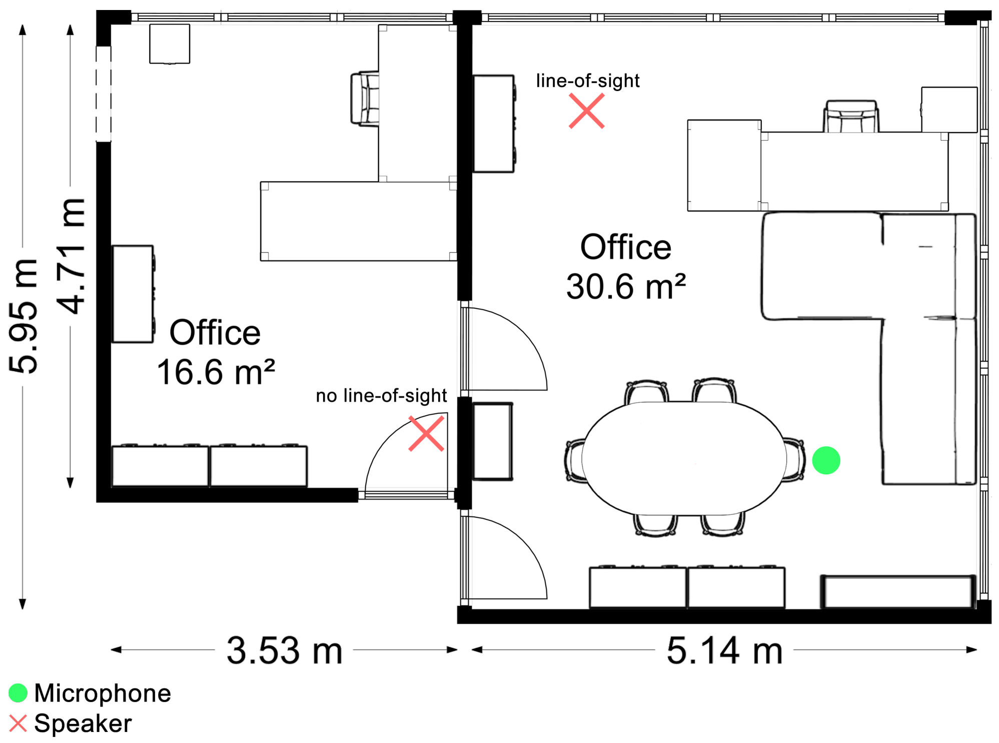


Figure A.2: Room layout of the office room.

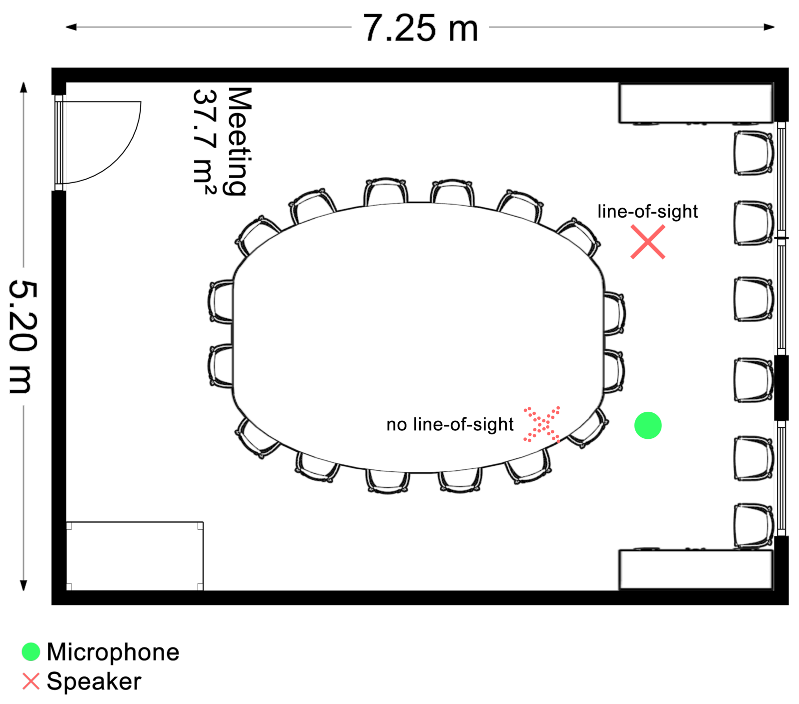


Figure A.3: Room layout of the meeting room.

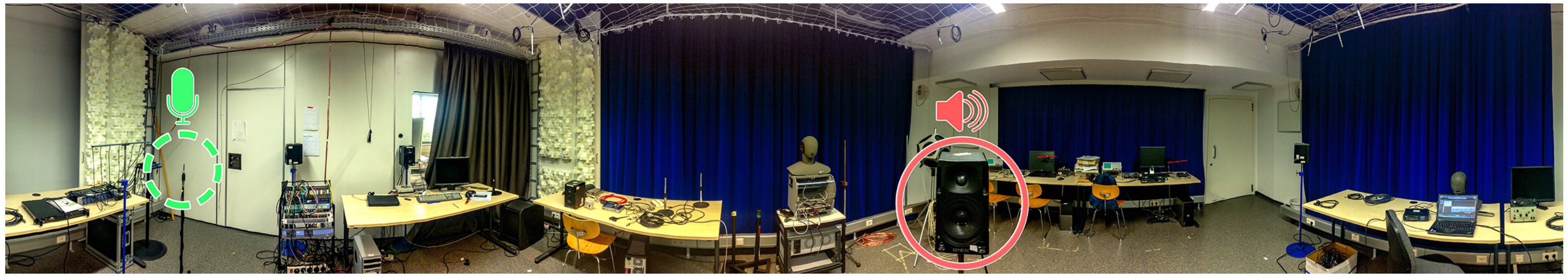


Figure A.4: 360 degree panorama shot of the lab setup used for the over-the-air recordings. The green dashed circle shows the microphone position and the red solid circle shows the position of the loudspeaker.

Bibliography

[1] T. Moynihan, "How to keep Amazon Echo and Google Home from responding to your TV," Feb. 2017. https://www.wired.com/2017/02/keep-amazon-echo-google-home-responding-tv/, as of April 12, 2021.

[2] V. Wong, "Burger King's new ad will hijack your Google Home," Apr. 2017. https://www.cnbc.com/2017/04/12/burger-kings-new-ad-will-hijack-your-google-home.html, as of April 12, 2021.

[3] A. Tilley, "How a few words to Apple's Siri unlocked a man's front door," Sept. 2016. https://www.forbes.com/sites/aarontilley/2016/09/21/apple-homekit-siri-security, as of April 12, 2021.

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, ICLR '14, (Banff, Alberta, Canada), Apr. 2014.

[5] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, pp. 2805–2824, Jan. 2019.

[6] A. Kerckhoffs, "La cryptographie militaire," *Journal des Sciences Militaires*, vol. 9, pp. 5–38, Jan. 1883.

[7] C. Welch, "Amazon's Alexa can now recognize different voices and give personalized responses," Oct. 2017. `https://www.theverge.com/circuitbreaker/2017/10/11/16460120/`, as of April 12, 2021.

[8] I. Apple, "Personalized Hey Siri," Apr. 2018. `https://machinelearning.apple.com/2018/04/16/personalized-hey-siri.html`, as of April 12, 2021.

[9] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, "Spoofing and countermeasures for speaker verification: A survey," *Speech Communication*, vol. 66, pp. 130–153, Feb. 2015.

[10] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models.* Heidelberg, Germany: Springer, 2 ed., 2007.

[11] ISO Central Secretary, "Information Technology – Coding of moving pictures and associated audio for digital storage media at up to 1.5 Mbits/s – Part3: Audio," Standard ISO/IEC 11172-3:1993, International Organization for Standardization, Geneva, Switzerland, 1993.

[12] H. Fletcher, "Auditory patterns," *Reviews of Modern Physics*, vol. 12, pp. 47–65, Jan. 1940.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, Nov. 1997.

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations*, ICLR '15, (San Diego, California, USA), May 2015.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Conference on Neural Information Processing Systems*, NIPS '17, (Long Beach, California, USA), pp. 5998–6008, Curran Associates, Dec. 2017.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* Cambridge, Massachusetts, USA: MIT Press, 1 ed., 2016.

[17] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference on Learning Representations*, ICLR ’15, (San Diego, California, USA), May 2015.

[18] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, “Can machine learning be secure?,” in *Symposium on Information, Computer and Communications Security*, ASIA CCS ’06, (Taipei, Taiwan), pp. 16–25, ACM, Mar. 2006.

[19] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, “The security of machine learning,” *Machine Learning*, vol. 81, pp. 121–148, May 2010.

[20] D. Lowd and C. Meek, “Adversarial learning,” in *International Conference on Knowledge Discovery and Data Mining*, KDD ’05, (Chicago, Illinois, USA), pp. 641–647, ACM, Aug. 2005.

[21] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in *European Symposium on Security and Privacy*, EuroSP ’16, (Saarbrücken, Germany), pp. 372–387, IEEE, Mar. 2016.

[22] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *International Conference on Learning Representations*, ICLR ’15, (San Diego, California, USA), May 2015.

[23] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in *International Conference on Machine Learning*, ICML ’18, (Stockholm, Sweden), pp. 274–283, PMLR, July 2018.

[24] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted Gaussian mixture models,” *Digital Signal Processing*, vol. 10, pp. 19–41, Jan. 2000.

[25] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, pp. 788–798, May 2011.

[26] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, “X-vectors: Robust DNN embeddings for speaker recognition,” in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP ’18, (Calgary, Alberta, Canada), pp. 5329–5333, IEEE, Apr. 2018.

[27] V. Bataev, M. Korenevsky, I. Medennikov, and A. Zatvornitskiy, “Exploring end-to-end techniques for low-resource speech recognition,” in *International Conference on Speech and Computer*, SPECOM ’18, (Leipzig, Germany), pp. 32–41, Springer, Sept. 2018.

[28] D. Wang, X. Wang, and S. Lv, “An overview of end-to-end automatic speech recognition,” *Symmetry*, vol. 11, pp. 1018:1–1018:26, Aug. 2019.

[29] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *International Conference on Machine Learning*, ICML ’06, (Pittsburgh, Pennsylvania, USA), pp. 369–376, ACM, June 2006.

[30] A. L. Maas, Z. Xie, D. Jurafsky, and A. Y. Ng, “Lexicon-free conversational speech recognition with neural networks,” in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT ’15, (Denver, Colorado, USA), pp. 345–354, ACL, May 2015.

[31] A. Graves, “Sequence transduction with recurrent neural networks,” in *Representation Learning Workshop*, RLW ’12, (Edinburgh, Scotland, United Kingdom), July 2012.

[32] L. Dong, S. Xu, and B. Xu, “Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition,” in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP ’18, (Calgary, Alberta, Canada), pp. 5884–5888, IEEE, Apr. 2018.

[33] J. Du, Y.-H. Tu, L. Sun, F. Ma, H.-K. Wang, J. Pan, C. Liu, J.-D. Chen, and C.-H. Lee, “The USTC-iFlytek system for CHiME-4 challenge,” in *Workshop on Speech Processing in Everyday Environments*, CHiME ’16, (San Francisco, California, USA), pp. 36–38, ISCA, Sept. 2016.

[34] N. Kanda, R. Ikeshita, S. Horiguchi, Y. Fujita, K. Nagamatsu, X. Wang, V. Manohar, N. E. Y. Soplin, M. Maciejewski, S.-J. Chen, A. S. Subramanian, R. Li, Z. Wang, J. Naradowsky, L. P. Garcia-Perera, and G. Sell, "The Hitachi/JHU CHiME-5 system: Advances in speech recognition for everyday home environments using multiple microphone arrays," in *Workshop on Speech Processing in Everyday Environments*, CHiME '18, (Hyderabad, India), pp. 6–10, ISCA, Sept. 2018.

[35] I. Medennikov, I. Sorokin, A. Romanenko, D. Popov, Y. Khokhlov, T. Prisyach, N. Malkovskii, V. Bataev, S. Astapov, M. Korenevsky, and A. Zatvornitskiy, "The STC system for the CHiME 2018 challenge," in *Workshop on Speech Processing in Everyday Environments*, CHiME '18, (Hyderabad, India), pp. 1–5, ISCA, Sept. 2018.

[36] J. Kollewe, "HSBC rolls out voice and Touch ID security for bank customers," Feb. 2016. `https://www.theguardian.com/business/2016/feb/19/hsbc-rolls-out-voice-touch-id-security-bank-customers`, as of April 12, 2021.

[37] G. Chollet, P. Perrot, W. Karam, C. Mokbel, S. Kanade, and D. Petrovska-Delacrétaz, "Identities, forgeries and disguises," *International Journal of Information Technology and Management*, vol. 11, pp. 138–152, Dec. 2011.

[38] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli, "Multimodal fusion for multimedia analysis: A survey," *Multimedia Systems*, vol. 16, pp. 345–379, Apr. 2010.

[39] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, Mar. 1998.

[40] A. H. Abdelaziz, S. Zeiler, and D. Kolossa, "Learning dynamic stream weights for coupled-HMM-based audio-visual speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, pp. 863–876, May 2015.

[41] M. Heckmann, F. Berthommier, and K. Kroschel, "Noise adaptive stream weighting in audio-visual speech recognition," *EURASIP Journal on Advances in Signal Processing*, vol. 2002, pp. 1260–1273, Nov. 2002.

[42] R. Schlüter and W. Macherey, “Comparison of discriminative training criteria,” in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP ’98, (Seattle, Washington, USA), pp. 493–496, IEEE, May 1998.

[43] D. A. Reynolds and R. C. Rose, “Robust text-independent speaker identification using Gaussian mixture speaker models,” *IEEE Transactions on Speech and Audio Processing*, vol. 3, pp. 72–83, Jan. 1995.

[44] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel, “Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification,” in *Conference of the International Speech Communication Association*, INTERSPEECH ’09, (Brighton, United Kingdom), pp. 1559–1562, ISCA, Sept. 2009.

[45] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, “Local binary patterns and its application to facial image analysis: A survey,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, pp. 765–781, Nov. 2011.

[46] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 2037–2041, Oct. 2006.

[47] S. O. Sadjadi, M. Slaney, and L. Heck, “MSR Identity Toolbox v1.0: A MATLAB toolbox for speaker-recognition research,” Technical Report MSR-TR-2013-133, Microsoft, Sept. 2013.

[48] Y. Ephraim and D. Malah, “Speech enhancement using a minimum mean-square error log-spectral amplitude estimator,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, pp. 443–445, Apr. 1985.

[49] P. C. Loizou, *Speech Enhancement: Theory and Practice*. Boca Raton, Florida, USA: CRC Press, 2 ed., 2017.

[50] P. S. Aleksic and A. K. Katsaggelos, “Audio-visual biometrics,” *Proceedings of the IEEE*, vol. 94, pp. 2025–2044, Nov. 2006.

[51] Z. Akhtar, *Security of Multimodal Biometric Systems Against Spoof Attacks.* PhD thesis, University of Cagliari, 2012.

[52] T. Kraft, "Analyzing state-of-the-art audio-visual authentication systems on the Web," Jan. 2017. Bachelor Thesis. Department of Electrical Engineering and Information Technology, Ruhr-Universität Bochum, Germany.

[53] W. Karam, H. Bredin, H. Greige, G. Chollet, and C. Mokbel, "Talking-face identity verification, audiovisual forgery, and robustness issues," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 746481:1–746481:15, May 2009.

[54] F. Verdet and J. Hennebert, "Impostures of talking face systems using automatic face animation," in *International Conference on Biometrics: Theory, Applications and Systems*, BTAS '08, (Washington, District of Columbia, USA), pp. 1–4, IEEE, Sept. 2008.

[55] D. Schabus, M. Pucher, and G. Hofer, "Joint audiovisual hidden semi-Markov model-based speech synthesis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, pp. 336–347, Apr. 2014.

[56] A. V. Nefian, L. Liang, X. Pi, X. Liu, C. Mao, and K. P. Murphy, "A coupled HMM for audio-visual speech recognition," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '02, (Orlando, Florida, USA), pp. 2013–2016, IEEE, May 2002.

[57] E. Argones-Rúa, H. Bredin, C. García-Mateo, G. Chollet, and D. González-Jiménez, "Audio-visual speech asynchrony detection using co-inertia analysis and coupled hidden Markov models," *Pattern Analysis and Applications*, vol. 12, pp. 271–284, Sept. 2009.

[58] E. Boutellaa, Z. Boulkenafet, J. Komulainen, and A. Hadid, "Audiovisual synchrony assessment for replay attack detection in talking face biometrics," *Multimedia Tools and Applications*, vol. 75, pp. 5329–5343, Aug. 2016.

[59] D. R. Hardoon, S. Szedmák, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, pp. 2639–2664, Dec. 2004.

[60] T. Kobayashi and N. Otsu, “Motion recognition using local auto-correlation of space-time gradients,” *Pattern Recognition Letters*, vol. 33, pp. 1188–1195, July 2012.

[61] M. R. Alam, R. Togneri, F. Sohel, M. Bennamoun, and I. Naseem, “Linear regression-based classifier for audio visual person identification,” in *International Conference on Communications, Signal Processing, and their Applications*, ICCSPA ’13, (Sharjah, United Arab Emirates), pp. 1–5, IEEE, Feb. 2013.

[62] C. Yu and L. Huang, “Biometric recognition by using audio and visual feature fusion,” in *International Conference on System Science and Engineering*, ICSSE ’12, (Dalian, China), pp. 173–178, IEEE, June 2012.

[63] M. R. Alam, M. Bennamoun, R. Togneri, and F. Sohel, “A deep neural network for audio-visual person recognition,” in *International Conference on Biometrics Theory, Applications and Systems*, BTAS ’15, (Arlington, Virginia, USA), pp. 1–6, IEEE, Sept. 2015.

[64] M. Todisco, H. Delgado, and N. W. D. Evans, “A new feature for automatic speaker verification anti-spoofing: Constant Q cepstral coefficients,” in *Odyssey: The Speaker and Language Recognition Workshop*, Odyssey ’16, (Bilbao, Spain), pp. 283–290, ISCA, June 2016.

[65] A. Aides and H. Aronowitz, “Text-dependent audiovisual synchrony detection for spoofing detection in mobile person recognition,” in *Conference of the International Speech Communication Association*, INTERSPEECH ’16, (San Francisco, California, USA), pp. 2125–2129, ISCA, Sept. 2016.

[66] J. Komulainen, I. Anina, J. Holappa, E. Boutellaa, and A. Hadid, “On the robustness of audiovisual liveness detection to visual speech animation,” in *International Conference on Biometrics Theory, Applications and Systems*, BTAS ’16, (Niagara Falls, New York, USA), pp. 1–8, IEEE, Sept. 2016.

[67] H. Bredin and G. Chollet, “Making talking-face authentication robust to deliberate imposture,” in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP ’08, (Las Vegas, Nevada, USA), pp. 1693–1696, IEEE, Mar. 2008.

[68] H. Bredin and G. Chollet, "Audiovisual speech synchrony measure: Application to biometrics," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, pp. 70186:1–70186:11, Dec. 2007.

[69] D. Dean, S. Sridharan, and T. Wark, "Audio-visual speaker verification using continuous fused HMMs," in *HCSNet Workshop on Use of Vision in Human-Computer Interaction*, VisHCI '06, (Canberra, Australia), pp. 87–92, Australian Computer Society, Nov. 2006.

[70] T. Fu, X. Liu, L. Liang, X. Pi, and A. V. Nefian, "Audio-visual speaker identification using coupled hidden Markov models," in *International Conference on Image Processing*, ICIP '03, (Barcelona, Spain), pp. 29–32, IEEE, Sept. 2003.

[71] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Toward human parity in conversational speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, pp. 2410–2423, Sept. 2017.

[72] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L. Lim, B. Roomi, and P. Hall, "English conversational telephone speech recognition by humans and machines," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 132–136, ISCA, Aug. 2017.

[73] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* Hoboken, New Jersey, USA: Prentice-Hall, 1 ed., 1993.

[74] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, ECML PKDD '13, (Prague, Czech Republic), pp. 387–402, Springer, Sept. 2013.

[75] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Symposium on Security and Privacy*, SP '17, (San Jose, California, USA), pp. 39–57, IEEE, May 2017.

[76] N. Carlini and D. Wagner, “Audio adversarial examples: Targeted attacks on speech-to-text,” in *Deep Learning and Security Workshop*, DLS ’18, (San Francisco, California, USA), pp. 1–7, IEEE, May 2018.

[77] T. Vaidya, Y. Zhang, M. Sherr, and C. Shields, “Cocaine Noodles: Exploiting the gap between human and machine speech recognition,” in *Workshop on Offensive Technologies*, WOOT ’15, (Washington, District of Columbia, USA), USENIX, Aug. 2015.

[78] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, “Hidden voice commands,” in *USENIX Security Symposium*, SSYM ’16, (Austin, Texas, USA), pp. 513–530, USENIX, Aug. 2016.

[79] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, “DolphinAttack: Inaudible voice commands,” in *Conference on Computer and Communications Security*, CCS ’17, (Dallas, Texas, USA), pp. 103–117, ACM, Oct. 2017.

[80] L. Song and P. Mittal, “POSTER: Inaudible voice commands,” in *ACM Conference on Computer and Communications Security*, CCS ’17, (Dallas, Texas, USA), pp. 2583–2585, ACM, Oct. 2017.

[81] N. Roy, H. Hassanieh, and R. R. Choudhury, “BackDoor: Making microphones hear inaudible sounds,” in *Conference on Mobile Systems, Applications, and Services*, MobiSys ’17, (Niagara Falls, New York, USA), pp. 2–14, ACM, June 2017.

[82] X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter, “CommanderSong: A systematic approach for practical adversarial voice recognition,” in *USENIX Security Symposium*, SSYM ’18, (Baltimore, Maryland, USA), pp. 49–64, USENIX, Aug. 2018.

[83] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The Kaldi speech recognition toolkit,” in *Workshop on Automatic Speech Recognition and Understanding*, ASRU ’11, (Waikoloa, Hawaii, USA), IEEE, Dec. 2011.

[84] M. Fujimoto, "Factored deep convolutional neural networks for noise robust speech recognition," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 3837–3841, ISCA, Aug. 2017.

[85] V. Manohar, D. Povey, and S. Khudanpur, "JHU Kaldi system for Arabic MGB-3 ASR challenge using diarization, audio-transcript alignment and transfer learning," in *Workshop on Automatic Speech Recognition and Understanding*, ASRU '17, (Okinawa, Japan), pp. 346–352, IEEE, Dec. 2017.

[86] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal Forced Aligner: Trainable text-speech alignment using Kaldi," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 498–502, ISCA, Aug. 2017.

[87] S. Ranjan and J. H. L. Hansen, "Improved gender independent speaker recognition using convolutional neural network based bottleneck features," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 1009–1013, ISCA, Aug. 2017.

[88] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "A network of deep neural networks for distant speech recognition," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '17, (New Orleans, Louisiana, USA), pp. 4880–4884, IEEE, Mar. 2017.

[89] J. Trmal, M. Wiesner, V. Peddinti, X. Zhang, P. Ghahremani, Y. Wang, V. Manohar, H. Xu, D. Povey, and S. Khudanpur, "The Kaldi OpenKWS system: Improving low resource keyword search," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 3597–3601, ISCA, Aug. 2017.

[90] P. Upadhyaya, O. Farooq, M. R. Abidi, and Y. V. Varshney, "Continuous Hindi speech recognition model based on Kaldi ASR toolkit," in *International Conference on Wireless Communications, Signal Processing and Networking*, WiSPNET '17, (Chennai, India), pp. 786–789, IEEE, Mar. 2017.

[91] J. Villalba, N. Brümmer, and N. Dehak, "Tied variational autoencoder backends for i-vector speaker recognition," in *Conference of the International Speech Communication Association*, INTERSPEECH '17, (Stockholm, Sweden), pp. 1004–1008, ISCA, Aug. 2017.

[92] K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny, "Building competitive direct acoustics-to-word models for English conversational speech recognition," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '18, (Calgary, Alberta, Canada), pp. 4759–4763, IEEE, Apr. 2018.

[93] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "The Microsoft 2016 conversational speech recognition system," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '17, (New Orleans, Louisiana, USA), pp. 5255–5259, IEEE, Mar. 2017.

[94] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *International Conference on Machine Learning*, ICML '18, (Stockholm, Sweden), pp. 2142–2151, PMLR, July 2018.

[95] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," *CoRR*, vol. abs/1605.07277, pp. 1–13, May 2016.

[96] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Security Symposium*, SSYM '16, (Austin, Texas, USA), pp. 601–618, USENIX, Aug. 2016.

[97] N. Papernot, P. D. McDaniel, I. J. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *ACM Asia Conference on Computer and Communications Security*, ASIA CCS '17, (Abu Dhabi, United Arab Emirates), pp. 506–519, ACM, Apr. 2017.

[98] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," in *Symposium on Security and Privacy*, SP '18, (San Francisco, California, USA), pp. 36–52, IEEE, May 2018.

[99] N. Schinkel-Bielefeld, N. Lotze, and F. Nagel, “Audio quality evaluation by experienced and inexperienced listeners,” in *International Congress on Acoustics*, ICA ’13, (Montréal, Quebec, Canada), pp. 060016:1–060016:8, ASA, June 2013.

[100] G. Navarro, “A guided tour to approximate string matching,” *ACM Computing Surveys*, vol. 33, pp. 31–88, Mar. 2001.

[101] M. Schoeffler, S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, “webMUSHRA – A comprehensive framework for Web-based listening tests,” *Journal of Open Research Software*, vol. 6, pp. 1–8, Feb. 2018.

[102] H. Abdullah, W. Garcia, C. Peeters, P. Traynor, K. R. B. Butler, and J. Wilson, “Practical hidden voice attacks against speech and speaker recognition systems,” in *Symposium on Network and Distributed System Security*, NDSS ’19, (San Diego, California, USA), ISOC, Feb. 2019.

[103] H. Yakura and J. Sakuma, “Robust audio adversarial example for a physical attack,” in *International Joint Conference on Artificial Intelligence*, IJCAI ’19, (Macao, China), pp. 5334–5341, IJCAI, Aug. 2019.

[104] J. Szurley and J. Z. Kolter, “Perceptual based adversarial audio attacks,” *CoRR*, vol. abs/1906.06355, pp. 1–10, June 2019.

[105] J. B. Allen and D. A. Berkley, “Image method for efficiently simulating smallroom acoustics,” *Journal of the Acoustical Society of America*, vol. 65, pp. 943–950, Apr. 1979.

[106] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, “Synthesizing robust adversarial examples,” in *International Conference on Machine Learning*, ICML ’18, (Stockholm, Sweden), pp. 284–293, PMLR, July 2018.

[107] S. Voran and C. Sholl, “Perception-based objective estimators of speech,” in *Workshop on Speech Coding for Telecommunications: Speech Coding for Interoperable Global Colmmunications*, SCFT ’95, (Annapolis, Maryland, USA), pp. 13–14, IEEE, Sept. 1995.

[108] W. Yang, *Enhanced Modified Bark Spectral Distortion (EMBSD): An Objective Speech Quality Measure Based on Audible Distortion and Cognition Model.* PhD thesis, Temple University, 1999.

[109] R. M. Neal, *Bayesian Learning for Neural Networks.* PhD thesis, University of Toronto, 1995.

[110] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, ICML '16, (New York City, New York, USA), pp. 1050–1059, JMLR, June 2016.

[111] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Conference on Neural Information Processing Systems*, NIPS '17, (Long Beach, California, USA), pp. 6402–6413, Curran Associates, Dec. 2017.

[112] C. Louizos and M. Welling, "Structured and efficient variational deep learning with matrix Gaussian posteriors," in *International Conference on Machine Learning*, ICML '16, (New York City, New York, USA), pp. 1708–1716, JMLR, June 2016.

[113] J. Snoek, Y. Ovadia, E. Fertig, B. Lakshminarayanan, S. Nowozin, D. Sculley, J. V. Dillon, J. Ren, and Z. Nado, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Conference on Neural Information Processing Systems*, NIPS '19, (Vancouver, British Columbia, Canada), pp. 13969–13980, Curran Associates, Dec. 2019.

[114] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Hoboken, New Jersey, USA: Wiley, 2 ed., 2006.

[115] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," *CoRR*, vol. abs/1703.00410, pp. 1–9, Mar. 2017.

[116] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, ICLR '18, (Vancouver, British Columbia, Canada), OpenReview.net, Apr. 2018.

[117] N. Papernot, F. Faghri, N. Carlini, I. J. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato,

W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, R. Long, and P. D. McDaniel, “Technical report on the CleverHans v2.1.0 adversarial examples library,” *CoRR*, vol. abs/1610.00768, pp. 1–12, Oct. 2016.

[118] R. S. Zimmermann, “Comment on ‘Adv-BNN: Improved adversarial defense through robust Bayesian neural network’,” *CoRR*, vol. abs/1907.00895, pp. 1–3, July 2019.

[119] M. Alzantot, B. Balaji, and M. Srivastava, “Did you hear that? Adversarial examples against automatic speech recognition,” in *Machine Deception Workshop*, NIPS ’17, (Long Beach, California, United States), Curran Associates, Dec. 2017.

[120] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, “Deep Speech: Scaling up end-to-end speech recognition,” *CoRR*, vol. abs/1412.5567, pp. 1–12, Dec. 2014.

[121] S. Khare, R. Aralikatte, and S. Mani, “Adversarial black-box attacks on automatic speech recognition systems using multi-objective evolutionary optimization,” in *Conference of the International Speech Communication Association*, INTERSPEECH ’19, (Graz, Austria), pp. 3208–3212, ISCA, Sept. 2019.

[122] R. Taori, A. Kamsetty, B. Chu, and N. Vemuri, “Targeted adversarial examples for black box audio systems,” in *Deep Learning and Security Workshop*, DLS ’19, (San Francisco, California, USA), pp. 15–20, IEEE, Sept. 2019.

[123] J. Li, S. Qu, X. Li, J. Szurley, J. Z. Kolter, and F. Metze, “Adversarial music: Real world audio adversary against wake-word detection system,” in *Conference on Neural Information Processing Systems*, NIPS ’19, (Vancouver, British Columbia, Canada), pp. 11908–11918, Curran Associates, Dec. 2019.

[124] Y. Chen, X. Yuan, J. Zhang, Y. Zhao, S. Zhang, K. Chen, and X. Wang, “Devil’s Whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices,” in *USENIX Security Symposium*, SSYM ’20, (Virtual Conference), pp. 2667–2684, USENIX, Aug. 2020.

[125] Y. Qin, N. Carlini, G. W. Cottrell, I. J. Goodfellow, and C. Raffel, “Imperceptible, robust, and targeted adversarial examples for automatic speech recog-

nition," in *International Conference on Machine Learning*, ICML '19, (Long Beach, California, USA), pp. 5231–5240, PMLR, June 2019.

[126] A. Shamir, I. Safran, E. Ronen, and O. Dunkelman, "A simple explanation for the existence of adversarial examples with small Hamming distance," *CoRR*, vol. abs/1901.10861, pp. 1–19, Jan. 2019.

[127] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Conference on Neural Information Processing Systems*, NIPS '19, (Vancouver, British Columbia, Canada), pp. 125–136, Curran Associates, Dec. 2019.

[128] L. Smith and Y. Gal, "Understanding measures of uncertainty for adversarial example detection," in *Conference on Uncertainty in Artificial Intelligence*, UAI '18, (Monterey, California, USA), pp. 560–569, AUAI Press, Aug. 2018.

[129] C. Louizos and M. Welling, "Multiplicative normalizing flows for variational Bayesian neural networks," in *International Conference on Machine Learning*, ICML '17, (Sydney, New South Wales, Australia), pp. 2218–2227, PMLR, Aug. 2017.

[130] V. Akinwande, C. Cintas, S. Speakman, and S. Sridharan, "Identifying audio adversarial examples via anomalous pattern detection," in *Workshop on Adversarial Learning Methods for Machine Learning and Data Mining*, AdvML '20, (San Diego, California, USA), ACM, Aug. 2020.

[131] S. Samizade, Z. Tan, C. Shen, and X. Guan, "Adversarial example detection by classification for deep speech recognition," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '20, (Barcelona, Spain), pp. 3102–3106, IEEE, May 2020.

[132] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Workshop on Artificial Intelligence and Security*, AISec '17, (Dallas, Texas, USA), pp. 3–14, ACM, Nov. 2017.

[133] Q. Zeng, J. Su, C. Fu, G. Kayas, L. Luo, X. Du, C. C. Tan, and J. Wu, "A multiversion programming inspired approach to detecting audio adversarial exam-

ples," in *Conference on Dependable Systems and Networks*, DSN '19, (Portland, Oregon, USA), pp. 39–51, IEEE, June 2019.

[134] Z. Yang, B. Li, P. Chen, and D. Song, "Characterizing audio adversarial examples using temporal dependency," in *International Conference on Learning Representations*, ICLR '19, (New Orleans, Louisiana , USA), OpenReview.net, May 2019.

[135] H. Liu and G. Ditzler, "Detecting adversarial audio via activation quantization error," in *International Joint Conference on Neural Networks*, IJCNN '20, (Glasgow, United Kingdom), pp. 1–7, IEEE, July 2020.

[136] M. Esmaeilpour, P. Cardinal, and A. L. Koerich, "Class-conditional defense GAN against end-to-end speech attacks," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '21, (Toronto, Ontario, Canada), IEEE, June 2021.

[137] T. Eisenhofer, L. Schönherr, J. Frank, L. Speckemeier, D. Kolossa, and T. Holz, "Dompteur: Taming audio adversarial examples," *CoRR*, vol. abs/2102.05431, pp. 1–17, Feb. 2021.

[138] A. Vyas, P. Dighe, S. Tong, and H. Bourlard, "Analyzing uncertainties in speech recognition using dropout," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '19, (Brighton, United Kingdom), pp. 6730–6734, IEEE, May 2019.

[139] T. Jayashankar, J. Le Roux, and P. Moulin, "Detecting audio attacks on ASR systems with dropout uncertainty," in *Conference of the International Speech Communication Association*, INTERSPEECH '20, (Shanghai, China), pp. 4671–4675, ISCA, Oct. 2020.

[140] A. H. Abdelaziz, S. Watanabe, J. R. Hershey, E. Vincent, and D. Kolossa, "Uncertainty propagation through deep neural networks," in *Conference of the International Speech Communication Association*, INTERSPEECH '15, (Dresden, Germany), pp. 3561–3565, ISCA, Sept. 2015.

[141] C. Huemmer, R. Maas, A. Schwarz, R. F. Astudillo, and W. Kellermann, "Uncertainty decoding for DNN-HMM hybrid systems based on numerical sam-

pling," in *Conference of the International Speech Communication Association*, INTERSPEECH '15, (Dresden, Germany), pp. 3556–3560, ISCA, Sept. 2015.

[142] D. Bohn, "Amazon says 100 million Alexa devices have been sold," Jan. 2019. `https://www.theverge.com/2019/1/4/18168565/`, as of April 12, 2021.

[143] A. Raju, S. Panchapagesan, X. Liu, A. Mandal, and N. Ström, "Data augmentation for robust keyword spotting under playback interference," *CoRR*, vol. abs/1808.00563, pp. 1–6, Aug. 2018.

[144] D. Kumar, R. Paccagnella, P. Murley, E. Hennenfent, J. Mason, A. Bates, and M. Bailey, "Skill squatting attacks on Amazon Alexa," in *Security Symposium*, SSYM '19, (Santa Clara, California, USA), pp. 33–47, USENIX, Aug. 2019.

[145] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian, "Dangerous Skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems," in *Symposium on Security and Privacy*, SP '19, (San Francisco, California, USA), pp. 1381–1396, IEEE, May 2019.

[146] N. Malkin, J. Deatrick, A. Tong, P. Wijesekera, S. Egelman, and D. Wagner, "Privacy attitudes of smart speaker users," *Privacy Enhancing Technologies*, pp. 250–271, July 2019.

[147] H. Chung, M. Iorga, J. Voas, and S. Lee, "Alexa, can I trust you?," *IEEE Computer*, vol. 50, pp. 100–104, Sept. 2017.

[148] D. J. Dubois, R. Kolcun, A. M. Mandalari, M. T. Paracha, D. Choffnes, and H. Haddadi, "When speakers are all ears: Characterizing misactivations of IoT smart speakers," in *Privacy Enhancing Technologies Symposium*, PETS '20, (Virtual Conference), pp. 255–276, Sciendo, July 2020.

[149] M. Day, G. Turner, and N. Drozdiak, "Amazon workers are listening to what you tell Alexa," Apr. 2019. `https://www.bloomberg.com/news/articles/2019-04-10/is-anyone-listening-to-you-on-alexa-a-global-team-reviews-audio`, as of April 12, 2021.

[150] L. Van Hee, D. Baert, T. Verheyden, and R. Van Den Heuvel, "Google employees are eavesdropping, even in your living room," July 2019. `https://www.vrt.be/vrtnws/en/2019/07/10/google-employees-are-eavesdropping-even-in-flemish-living-rooms/`, as of April 12, 2021.

[151] C. Lecher, "Google will pause listening to EU voice recordings while regulators investigate," Aug. 2019. `https://www.theverge.com/2019/8/1/20750327/`, as of April 12, 2021.

[152] A. Hern, "Alexa users can now disable human review of voice recordings," Aug. 2019. `https://www.theguardian.com/technology/2019/aug/05/alexa-allows-users-to-disable-human-review-of-voice-recordings`, as of April 12, 2021.

[153] C. Gartenberg, "Apple apologizes for Siri audio recordings, announces privacy changes going forward," Aug. 2019. `https://www.theverge.com/2019/8/28/20836760/`, as of April 12, 2021.

[154] J. Guo, K. Kumatani, M. Sun, M. Wu, A. Raju, N. Ström, and A. Mandal, "Time-delayed bottleneck highway networks using a DFT feature for keyword spotting," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '18, (Calgary, Alberta, Canada), pp. 5489–5493, IEEE, Apr. 2018.

[155] M. Wu, S. Panchapagesan, M. Sun, J. Gu, R. Thomas, S. N. P. Vitaladevuni, B. Hoffmeister, and A. Mandal, "Monophone-based background modeling for two-stage on-device wake word detection," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '18, (Calgary, Alberta, Canada), pp. 5494–5498, IEEE, Apr. 2018.

[156] I. Apple, "Hey Siri: An on-device DNN-powered voice trigger for Apple's personal assistant," Oct. 2017. `https://machinelearning.apple.com/2017/10/01/hey-siri.html`, as of April 12, 2021.

[157] I. Google, "Google Assistant with Voice Match – Upgraded Voice Match," Feb. 2020. `https://support.google.com/assistant/answer/9071681`, as of April 12, 2021.

[158] A. Gebhart, “Is Google Home good at voice recognition?,” Apr. 2017. https://www.cnet.com/news/is-google-home-good-at-voice-recognition/, as of April 12, 2021.

[159] I. Google, “Google Assistant sensitivity,” Apr. 2020. https://www.blog.google/products/assistant/more-ways-fine-tune-google-assistant-you/, as of April 12, 2021.

[160] T. Karczewski, “Cloud-based wake word verification improves “Alexa” wake word accuracy,” May 2017. https://developer.amazon.com/en-US/docs/alexa/alexa-voice-service/enable-cloud-based-wake-word-verification.html, as of April 12, 2021.

[161] A. Hern, “Apple contractors ‘regularly hear confidential details’ on Siri recordings,” July 2019. https://www.theguardian.com/technology/2019/jul/26/apple-contractors-regularly-hear-confidential-details-on-siri-recordings, as of April 12, 2021.

[162] D. Pierce, “Google’s new magic number for storing personal data: 18 months,” June 2020. https://www.protocol.com/google-delete-data-18-months, as of April 12, 2021.

[163] T. Scott, “Smart speakers statistics: Report 2021,” Dec. 2020. https://speakergy.com/smart-speakers-statistics/, as of April 12, 2021.

[164] K. Lyons, “Amazon is still crushing Google and Apple in the smart speaker market,” Feb. 2020. https://www.theverge.com/2020/2/10/21131988/, as of April 12, 2021.

[165] B. Kinsella, “Deutsche Telekom and SoundHound make their partnership public with Houndify supporting Magenta,” Oct. 2019. https://voicebot.ai/2019/10/31/deutsche-telekom-and-soundhound-make-their-partnership-public-with-houndify-supporting-magenta/, as of April 12, 2021.

[166] Orange, S.A., “Orange launches the voice assistant Djingo,” Nov. 2019. https://djingo.orange.fr/, as of April 12, 2021.

[167] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '15, (Brisbane, Queensland, Australia), pp. 5206–5210, IEEE, Apr. 2015.

[168] M. Foundation and Community, "Mozilla: Common Voice," June 2017. `https://voice.mozilla.org`, as of April 12, 2021.

[169] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Workshop on Speech and Natural Language*, HLT '92, (Harriman, New York, USA), pp. 357–362, Morgan Kaufmann, Feb. 1992.

[170] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Conference of the International Speech Communication Association*, INTERSPEECH '18, (Hyderabad, India), pp. 1561–1565, ISCA, Sept. 2018.

[171] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," *CoRR*, vol. abs/1510.08484, pp. 1–4, Oct. 2015.

[172] S. Sigtia, E. Marchi, S. Kajarekar, D. Naik, and J. Bridle, "Multi-task learning for speaker verification and voice trigger detection," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '20, (Barcelona, Spain), pp. 6844–6848, IEEE, May 2020.

[173] L. Dixon, J. Li, J. Sorensen, N. Thain, and L. Vasserman, "Measuring and mitigating unintended bias in text classification," in *AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, (New Orleans, Louisiana, USA), pp. 67–73, ACM, Feb. 2018.

[174] R. Tatman, "Gender and dialect bias in YouTube's automatic captions," in *Workshop on Ethics in Natural Language Processing*, EthNLP@EACL '17, (Valencia, Spain), pp. 53–59, ACL, Apr. 2017.

[175] S. Kiritchenko and S. M. Mohammad, "Examining gender and race bias in two hundred sentiment analysis systems," in *Conference on Lexical and Computational Semantics*, *SEM@NAACL-HLT '18, (New Orleans, Louisiana, USA), pp. 43–53, ACL, June 2018.

[176] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *Speech Synthesis Workshop*, SSW '16, (Sunnyvale, California, USA), ISCA, Sept. 2016.

[177] C. Cieri, D. Miller, and K. Walker, "The Fisher Corpus: A resource for the next generations of speech-to-text," in *International Conference on Language Resources and Evaluation*, LREC '04, (Lisbon, Portugal), pp. 69–71, ELRA, May 2004.

[178] K. A. Lenzo, "Carnegie Mellon pronouncing dictionary (CMUdict) - version 0.7b," Nov. 2014. `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`, as of April 12, 2021.

[179] J. Bort, "Amazon engineers had one good reason and one geeky reason for choosing the name Alexa," July 2016. `https://www.businessinsider.com/why-amazon-called-it-alexa-2016-7`, as of April 12, 2021.

[180] Y. Heisler, "Steve Jobs wasn't a fan of the Siri name," Mar. 2012. `https://www.networkworld.com/article/2221246/`, as of April 12, 2021.

[181] M. Singh, "Amazon's Alexa now speaks Hindi," Sept. 2019. `https://techcrunch.com/2019/09/18/amazon-alexa-hindi-india/`, as of April 12, 2021.

[182] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *International Conference on Acoustics, Speech and Signal Processing*, ICASSP '19, (Brighton, United Kingdom), pp. 6381–6385, IEEE, May 2019.

[183] J. Kastrenakes, "Pixel 4 recorder app can transcribe speech in real-time without an internet connection," Oct. 2019. `https://www.theverge.com/2019/10/15/20915452/`, as of April 12, 2021.

[184] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril, M. Primet, and J. Dureau, "Snips Voice Platform: An embedded spoken language understanding system for

private-by-design voice interfaces," *CoRR*, vol. abs/1805.10190, pp. 1–29, May 2018.

[185] S. H. Mallidi, R. Maas, K. Goehner, A. Rastrow, S. Matsoukas, and B. Hoffmeister, "Device-directed utterance detection," in *Conference of the International Speech Communication Association*, INTERSPEECH '18, (Hyderabad, India), pp. 1225–1228, ISCA, Sept. 2018.

[186] I. Amazon, "Alexa: Turn on Follow-Up mode," Mar. 2018. `https://www.amazon.com/gp/help/customer/display.html?nodeId=202201630`, as of April 12, 2021.

[187] A. H. Mhaidli, M. K. Venkatesh, Y. Zou, and F. Schaub, "Listen Only When Spoken To: Interpersonal communication cues as smart speaker privacy controls," *Privacy Enhancing Technologies*, vol. 2020, pp. 251–270, Apr. 2020.

[188] J. Lau, B. Zimmerman, and F. Schaub, "Alexa, are you listening? Privacy perceptions, concerns and privacy-seeking behaviors with smart speakers," in *Conference on Computer-Supported Cooperative Work and Social Computing*, CSCW '18, (New York City, New York, USA), pp. 102:1–102:31, ACM, Nov. 2018.

[189] N. Abdi, K. M. Ramokapane, and J. M. Such, "More than smart speakers: Security and privacy perceptions of smart home personal assistants," in *Symposium on Usable Privacy and Security*, SOUPS '19, (Santa Clara, California, USA), pp. 451–466, USENIX, Aug. 2019.

[190] T. Ammari, J. Kaye, J. Y. Tsai, and F. Bentley, "Music, search, and IoT: How people (really) use voice assistants," *ACM Transactions on Computer-Human Interaction*, vol. 26, pp. 17:1–17:28, Apr. 2019.

[191] Y. Huang, B. Obada-Obieh, and K. Beznosov, "Amazon vs. My Brother: How users of shared smart speakers perceive and cope with privacy risks," in *ACM Conference on Human Factors in Computing Systems*, CHI '20, (Honolulu, Hawaii, USA), pp. 402:1–402:13, ACM, Apr. 2020.

[192] M. Tabassum, T. Kosiundefinedski, A. Frik, N. Malkin, P. Wijesekera, S. Egelman, and H. R. Lipford, "Investigating users' preferences and expectations for

always-listening voice assistants," *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, pp. 153:1–153:23, Dec. 2019.

[193] E. Zeng, S. Mare, and F. Roesner, "End user security and privacy concerns with smart homes," in *Symposium on Usable Privacy and Security*, SOUPS '17, (Santa Clara, California, USA), pp. 65–80, USENIX, July 2017.

[194] R. Tolosana, R. Vera-Rodríguez, J. Fiérrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, vol. 64, pp. 131–148, Dec. 2020.

[195] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz, "Leveraging frequency analysis for deep fake image recognition," in *International Conference on Machine Learning*, ICML '20, (Virtual Conference), pp. 3247–3258, PMLR, July 2020.

[196] H. Aghakhani, T. Eisenhofer, L. Schönherr, D. Kolossa, T. Holz, C. Kruegel, and G. Vigna, "VENOMAVE: Clean-label poisoning against speech recognition," *CoRR*, vol. abs/2010.10682, pp. 1–13, Oct. 2020.