mgr inż. Piotr Kawa

On Generalization and Robustness of Audio DeepFake Detection

Doctoral dissertation

Supervisor:
prof. dr hab. inż. Marek Klonowski

Co-supervisor:
dr inż. Piotr Syga

Wrocław University of Science and Technology

Wrocław 2024

Streszczenie rozprawy doktorskiej „On Generalization and Robustness of Audio DeepFake Detection”

Piotr Kawa

W niniejszej rozprawie przedstawiono rozwiązania mające na celu poprawę metod detekcji wygenerowanej komputerowo mowy tzw. *dźwiękowych manipulacji DeepFake.* Zaproponowane usprawnienia dotyczą problemów dziedzinowych: poprawy generalizacji detektorów, zwiększenia wydajności algorytmów, poprawy odporności detektorów na ataki oraz eksploracji metod reprezentacji danych w oparciu o głębokie sieci neuronowe.

Rozprawa doktorska oparta jest na następujących publikacjach:

1. Kawa, P., Plata, M., Syga, P. *Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection*, Interspeech 2022,

2. Kawa, P., Plata, M., Syga, P. *SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection*, TrustCom 2022,

3. Kawa, P., Plata, M., Syga, P. *Defense Against Adversarial Attacks on Audio DeepFake Detection*, Interspeech 2023,

4. Kawa, P., Plata, M., Czuba, M., Szymański, P., Syga, P. *Improved DeepFake Detection Using Whisper Features*, Interspeech 2023.

Generalizacja detektorów DeepFake jest powszechnym i wciąż nierozwiązanym problem niższej skuteczności modeli na danych testowych, których rozkład różni się od danych treningowych. Jednym z powodów jest nadmierne dopasowanie (ang. overfitting) — modele zamiast nauczyć się ogólnych wzorców charakteryzujących próbki DeepFake, uczą się artefaktów poszczególnych generatorów. Ponadto, aktualnie wykorzystywane zbiory treningowe, często nie zawierają próbek stworzonych z użyciem najnowszych generatorów. W celu poprawy generalizacji i stabilności metod detekcji, określanych przez efektywność na danych spoza zbioru treningowego, zaproponowano podejście *Attack Agnostic Dataset.* Metoda ta polega na rozłącznym podziale ataków pomiędzy zbiory treningowe, walidacyjne i testowe. Wyniki omawiane w rozprawie oparte są na trzech zbiorach: WaveFake, FakeAVCeleb oraz ASVspoof 2019 LA. Metody podziału ataków pomiędzy podzbiory mają na celu symulowanie różnych scenariuszy np. trening na zbiorze opartym tylko na podobnych metodach generowania DeepFake. Analiza wyników poszczególnych podziałów pozwala wybrać modele o najlepszej generalizacji na danych spoza zbioru treningowego. Metoda *Attack Agnostic Dataset* może być w łatwy sposób rozszerzona o kolejne zbiory i podziały. W ramach prac zaproponowano nową reprezentację danych opartą na połączeniu spektrogramu melowego oraz parametrów liniowo-cepstralnych w wyniku czego uzyskano wyniki EER o 5% niższe niż w przypadku pojedynczych reprezentacji.

Globalna natura problemu manipulacji DeepFake, której wynikiem jest duża ilość wygenerowanej treści wymaga istnienia nisko-kosztowych metod detekcji zapewniających wysoki poziom efektywności. Pozwala to zarówno na użycie wyżej wymienionych metod w celu analizy treści umieszczanej np. na portalach społecznościowych, jak również, zwiększa dostępność tych rozwiązań pozwalając zwykłym obywatelom na samodzielną weryfikację treści. W tym celu stworzono sieć neuronową *SpecRNet.* Jest to rekurencyjny model przetwarzający dwuwymiarowe reprezentacje sygnału dźwiękowego. Architektura rozwiązania zainspirowana została popularnym modelem anti-spoofingowym RawNet2 przetwarzającym sygnał dźwiękowy przy użyciu 60 razy większej liczby parametrów. Architektura *SpecRNet* pozwala na 40% obniżenie czasu przetwarzania próbek w porównaniu do modelu LCNN uważanego ówcześnie za jeden z najszybszych i najskuteczniejszych metod detekcji. *SpecRNet* odznacza się porównywalnymi wynikami, co zostało dodatkowo potwierdzone na podstawie zaproponowanych przez nas testów dotyczących m.in. scenariuszy małej ilości danych treningowych czy krótkich próbek dźwiękowych.

W celu zmniejszenia szansy wykrycia wygenerowanej próbki, adwersarz jest w stanie wykorzystać różne rodzaje modyfikacji by ukryć artefakty charakteryzujące sztuczne dane. Jedną z najbardziej efektywnych metod są ataki adwersarialne, które poprzez niezauważalne modyfikacje danych wejściowych są w stanie w skuteczny sposób zaburzyć działanie sieci neuronowych. W rozprawie przedstawiono pierwszą analizę wpływu wyżej wymienionych ataków na detektory DeepFake. W ramach badań przeprowadzono analizę wpływu dwóch typów scenariuszy — white-box oraz transferability różniących się zakresem wiedzy adwersarza na temat atakowanego systemu. Wykorzystując trzy różne rodzaje ataków (FGSM, PGD oraz FAB) zwiększono EER od wyjściowej wartości 0.0221, do 0.9905 w przypadku scenariusza white-box oraz do 0.4867 w przypadku scenariusza transferability. Następnie zaproponowano nową metodę adaptatywnego treningu adwersarialnego mającego na celu zwiększenie odporności sieci na ataki. Wykorzystanie zaproponowanej metody pozwala na obniżenie EER do 0.0982 w scenariuszu white-box oraz 0.1091 w scenariuszu transferability.

Reprezentacja sygnału dźwiękowego przetwarzanego przez klasyfikatory DeepFake ma istotny wpływ na ich efektywność. Coraz większa liczba proponowanych rozwiązań oparta jest na danych uzyskiwanych z sieci neuronowych trenowanych w ramach metod uczenia self-supervised (SSL). Reprezentacje te pozwalają na osiągnięcie lepszych wyników w porównaniu do standardowych algorytmów przetwarzania sygnałów. W ramach rozprawy jako alternatywę dla rozwiązań SSL zaproponowano wykorzystanie modelu rozpoznawania mowy Whisper. Architektura ta została wytrenowana na największym zbiorze przetwarzania mowy składającym się z ponad 860.000 godzin nagrań. Przeprowadzono analizę wykorzystania modelu zarówno jako samodzielny ekstraktor cech, jak również jako integralną część detektorów używaną w procesie dostrajania. Zaproponowana reprezentacja danych składająca się z danych z modelu Whisper połączonych z parametrami mel-cepstralnymi pozwala na poprawę wyjściowego wyniku na zbiorze In-The-Wild o 26%.

Abstract of the doctoral dissertation ”On Generalization and Robustness of Audio DeepFake Detection”

Piotr Kawa

In the following dissertation, we present solutions to improve methods for detecting computer-generated speech — a problem referred to as *audio DeepFake detection*. The proposed improvements address the following domain problems: improving the generalization of the detectors, increasing the speed of the solutions, improving detector robustness to the attacks, and exploring methods of data representation based on deep neural networks.

The dissertation is based on the following scientific publications:

1. Kawa, P., Plata, M., Syga, P. *Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection*, Interspeech 2022,

2. Kawa, P., Plata, M., Syga, P. *SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection*, TrustCom 2022,

3. Kawa, P., Plata, M., Syga, P. *Defense Against Adversarial Attacks on Audio DeepFake Detection*, Interspeech 2023,

4. Kawa, P., Plata, M., Czuba, M., Szymański, P., Syga, P. *Improved DeepFake Detection Using Whisper Features*, Interspeech 2023.

Generalization of DeepFake detectors is a common, and still unsolved, problem of lower model performance on test data with the distribution differing from training data. One of the reasons is overfitting — models, instead of learning the general patterns characterizing DeepFake samples, learn the artifacts of the individual generating methods. Moreover, the currently used training datasets often do not contain samples created with the latest synthesis methods. To improve the generalization and stability of detectors (determined by efficiency on data outside the training dataset), we propose the *Attack Agnostic Dataset* framework. This method is based on the disjoint division of attacks between training, validation, and test subsets. The results discussed in the dissertation are based on three datasets: WaveFake, FakeAVCeleb and ASVspoof 2019 LA. The presented techniques of splitting the attacks between subsets are designed to simulate different scenarios, e.g. training using a dataset based only on similar DeepFake generation methods. Analyzing the results of the particular splits allows us to select models with the best generalization on data outside the training set. *Attack Agnostic Dataset* can be easily extended to include additional datasets and splits. We propose a new representation based on a combination of mel spectrogram and linear-frequency cepstral coefficients, and as a result, we obtain EER 5% lower than in the case of single representations.

The global nature of the DeepFake phenomenon, resulting in a large volume of artificially generated content, requires low-cost, high-efficiency detection methods. Such solutions enable not only the analysis of the content posted on social networks but also increase the availability of such methods, allowing ordinary citizens to verify the content themselves. We present the neural network *SpecRNet*. It is a recurrent model that processes two-dimensional representations of a raw audio signal. Its architecture was inspired by the popular RawNet2 anti-spoofing model that processes the audio signal using 60 times more parameters. *SpecRNet's* inference speed is 40% faster in comparison to the LCNN model, considered one of the fastest and best-performing detectors at the time. *SpecRNet* provides similar performance, which we confirm by introducing and performing benchmarks covering scenarios such as small volumes of training data or short utterances.

To reduce the possibility of detecting a generated sample, an adversary can utilize various modifications to hide artifacts characterizing the fake samples. One of the most effective methods is adversarial attacks, which, through imperceptible modifications of the input data, can effectively decrease the performance of neural networks. In the dissertation, we present the first analysis of the impact of the attacks mentioned above on the DeepFake detectors. We analyze the impact of two types of scenarios — white-box and transferability differing in the degree of the adversary's knowledge of the system under attack. Using three different types of attacks (FGSM, PGD and FAB), we can increase the EER from the initial value of 0.0221 to 0.9905 for the white-box scenario and to 0.4867 for the transferability scenario. Next, we propose a new method of adaptive adversarial training aimed at increasing the network's robustness to the attacks. Using the proposed method, the EER can be decreased to 0.0982 in the white-box scenario and 0.1091 in the transferability scenario.

The representation of the audio signal processed by DeepFake classifiers significantly impacts their effectiveness. An increasing number of proposed solutions are based on data from neural networks trained by self-supervised learning (SSL) methods. These representations provide better performance compared to standard signal processing algorithms. As part of the dissertation, we propose using the Whisper speech recognition model as an alternative to SSL methods. This model has been trained on the largest speech-processing dataset consisting of more than 860,000 hours of content. We are analyzing using the model both as a standalone feature extractor and as an integral part of the detectors used in the fine-tuning process. Our proposed data representation consisting of the hidden state of the Whisper model combined with mel-frequency cepstral coefficients improves the baseline results on the In-The-Wild dataset by 26%.

Contents

Contents **I**

1 Introduction **1**

2 Related work **7**

2.1 Speech synthesis 7
2.1.1 History 7
2.1.2 Applications 8
2.1.3 Products and toolkits 9
2.1.4 Taxonomy 9
2.2 Audio Representations 11
2.2.1 Waveform representation 11
2.2.2 Spectrograms 12
2.2.3 Linear and mel frequency cepstral coefficients 13
2.3 DeepFakes 14
2.3.1 Human perception of audio DeepFakes 17
2.3.2 DeepFake detection 19
2.3.3 Performance metrics 20
2.3.4 Voice authenticity taxonomy 22
2.3.5 Detection methods 22
2.4 DeepFake detection datasets 25
2.4.1 ASVspoof 25
2.4.2 WaveFake 27
2.4.3 FakeAVCeleb 27
2.4.4 In-The-Wild 28
2.4.5 Other datasets 29
2.5 Adversarial Attacks 32
2.5.1 Fast Gradient Sign Method 33
2.5.2 Projected Gradient Descent 34

2.5.3 Fast Adaptive Boundary 34
2.6 Research goals 34

3 Evaluating generalization of detection models 37
3.1 Attack Agnostic Dataset 37
3.2 Preprocessing procedure 40
3.3 Evaluated architectures 40
3.4 Benchmark 41
3.5 Front-ends comparison 43
3.6 Ablation study 44
3.7 Conclusion 45

4 Fast and reliable detection using SpecRNet 47
4.1 SpecRNet 48
4.2 Pipeline description 49
4.3 Baseline comparison on a full dataset 50
4.4 Time benchmark 53
4.5 Limited attacks benchmark 55
4.6 Short utterances scenario 56
4.7 Data scarcity scenario 58
4.8 Conclusions 60

5 Adversarial attacks and defense against them 61
5.1 Included scenarios 62
5.2 Adversarial attacks 62
5.3 Models 64
5.4 Datasets and preprocessing 66
5.5 White–box benchmark 66
5.6 Transferability benchmark 68
5.7 Adversarial Training 70
5.8 Ablation study 74
5.9 Conclusions 75

6 Using Whisper in detecting DeepFakes 77
6.1 Whisper ASR model 78
6.2 Models 80
6.3 Front-ends 80
6.4 Setup 82
6.5 Preprocessing 82
6.6 Baseline comparison 83

6.7 Concatenated features 85
6.8 Fine-tuning 86
6.9 ASVspoof 2021 DF tests 88
6.10 Features comparison 89
6.11 Conclusions 92

7 Summary 95

Bibliography 101

Chapter 1

Introduction

This dissertation focuses on the implementation of artificial intelligence (AI) solutions to address specific problems of impersonating someone's voice or attempting to denigrate a victim by putting false utterances into circulation. In particular, we address the problem of *audio DeepFake manipulations* — utterances created using AI generative models. Due to the high quality of these materials and the ease with which they can be produced, they pose a significant threat [1]. The most prominent examples include money extortion [2, 3], or spreading fake news [4]. The following dissertation focuses on methods of determining the authenticity of utterances — a task referred to as *audio DeepFake detection*.

The subsequent chapters of the dissertation cover various aspects of detecting audio DeepFake manipulations. Chapter 2 describes the definitions and motivations for the problems presented in the dissertation. It also contains information about the current state of the field of *audio DeepFake detection* and covers the research questions addressed in the dissertation. Chapter 3 focuses on the generalization of the DeepFake detection solutions. In this chapter, we propose a novel benchmark called the *Attack Agnostic Dataset*, which allows in-depth evaluation of the generalization and stability properties of DeepFake detectors and, therefore, choosing the detector that might perform best on out-of-domain data. Chapter 4 introduces a novel detection neural network — *SpecRNet*. The model achieves comparable results in detecting audio DeepFake manipulations while providing a 40% improvement in processing speed compared to related work. Additionally, we introduce three novel benchmarks which are later used to evaluate the detection system. Chapter 5 covers adversarial attacks on audio DeepFake detection methods, where we first show that these attacks can successfully decrease the performance of the detectors, making them ineffectual, and later, we introduce a novel training technique that significantly increases the robustness against such attacks without compromising much of performance on non-adversarial data. Chapter 6 explores the topic of the audio representations used in detecting DeepFakes. We are first to extract information stored in the hidden layers of the state-of-the-art Whisper automatic speech recognition (ASR) model [5] and use it in tasks other than ASR. Thanks to this, we obtain a 26% reduction of the Equal Error Rate metric (refer to Section 2.3.3) on one of the most demanding DeepFake detection datasets. Chapter 7 provides a concise summary of the findings presented in this thesis.

The majority of the content in this dissertation derives from previously published scientific articles, which were later expanded by additional experiments. The following list comprises these manuscripts, accompanied by a brief description of their most significant outcomes:

1. *Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection* — [6] was written jointly with Marcin Plata and Piotr Syga and presented at the INTERSPEECH 2022 conference. The main contributions of this work include introducing a benchmark to examine the generalization of DeepFake detection models and proposing a novel audio representation leading to more generalized and stable results. In this research, the dissertation's author was responsible for preparing source code, conducting experiments and was involved in writing the paper.

2. *SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection* — [7] was written jointly with Marcin Plata and Piotr Syga and presented at The 21st IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2022). In this work, we introduced a novel architecture of a DeepFake detector called SpecRNet, which provided a significant speed enhancement with no significant decrease in performance. We backed the results based on the three novel benchmarks we introduced, in particular low-resource and short utterances scenarios and limited attacks benchmark providing in-depth analysis of the influence of particular attacks on the models' final performance. The author's contribution included introducing the SpecRNet's architecture, conducting the experiments, and co-writing the article.

3. *Defense Against Adversarial Attacks on Audio DeepFake Detection* — [8] was written jointly with Marcin Plata and Piotr Syga and presented at the INTERSPEECH 2023 conference. In this work, we conducted the first-ever analysis of the influence of adversarial attacks on audio DeepFake detectors. We later increased their robustness against the attacks thanks to our new method for adaptive adversarial training. In this research, the thesis author was responsible for preparing training pipelines, conducting adversarial attacks, and related training and was involved in writing the manuscript.

4. *Improved DeepFake Detection Using Whisper Features* — [9] was written jointly with Marcin Plata, Michał Czuba, Piotr Szymański and Piotr Syga and presented at the INTERSPEECH 2023 conference. Our main contribution was enhancing the performance on the In-The-Wild dataset by 26%. We achieved that by utilizing the Whisper ASR model, which, in this work, was the first-ever used for a task other than automatic speech recognition. In this research, the dissertation's author was responsible for introducing the concept of using Whisper as a front-end, was involved in designing and conducting experiments, and co-wrote the manuscript.

In addition, during the Ph.D., the dissertation's author (co-)authored the following manuscripts:

1. *Verify It Yourself: A Note on Activation Functions' Influence on Fast DeepFake Detection.* [10] was written jointly with Piotr Syga and presented at the 18th International Conference on Security and Cryptography (SECRYPT 2021). This work focused on enhancing computationally undemanding methods for visual DeepFake detection. We improved the performance of MesoNet neural network [11] by replacing its activation functions and achieving over 1% performance improvement along with increased stability. The work introduced a novel activation function — Pish, which provided enhanced performance on specific datasets at the cost of a slight time overhead. In this research, the author was responsible for preparing and conducting the experiments and was involved in the writing process.

2. *MLAAD: The Multi-Language Audio Anti-Spoofing Dataset* [12] was written jointly with Nicolas M. Müller, Wei Herng Choong, Edresson Casanova, Eren Gölge, Thorsten Müller, Piotr Syga, Philip Sperl and Konstantin Böttinger and was accepted at the International Joint

Conference on Neural Networks 2024 (IJCNN 2024) conference. The main contribution was presenting the first-ever multi-language DeepFake detection dataset, solving one of the most prominent research gaps in the field — the scarcity of differently distributed datasets and lack of datasets in languages other than English, Chinese, and Japanese. The dataset comprised over 160 hours of synthetic speech in 23 languages created using 52 Text-To-Speech models. The author's contribution to this work included preparing the TTS generation pipelines and co-writing the article.

3. *A New Approach to Voice Authenticity* [13] was written jointly with Nicolas M. Müller, Shen Hu, Matthias Neu, Jennifer Williams, Philip Sperl and Konstantin Böttinger and was accepted at the INTERSPEECH 2024 conference. We proposed a conceptual shift away from the binary paradigm of audio being either *real* (genuine human speech) or *fake* (produced by TTS or VC systems). Instead, we introduced a paradigm of pinpointing particular *voice edits*. Apart from TTS and VC speech manipulations, the edits include various digital signal processing manipulations like speed or pitch alteration, which were reported to be used for deceiving the audiences [14, 15, 16]. We introduced a novel benchmark dataset based on the M-AILABS dataset [17] comprising 20 voice edits grouped into six categories. The author's contribution to this work covered preparing the voice edits creation pipelines and co-writing the article.

During the Ph.D. studies the author was supported under the grant funded by Polish National Science Center (pol. Narodowe Centrum Nauki) *Information concealment and privacy in (mostly) distributed systems* (pol. *Ukrywanie informacji i prywatność w systemach (głównie) rozproszonych*) from 01.06.2022 to 30.11.2022 (project number 2018/29/B/ST6/02969).

Moreover, the author took part in the following R&D projects funded by The National Centre for Research and Development (pol. Narodowe Centrum Badań i Rozwoju) and The Polish Agency for Enterprise Development (pol. Polska Agencja Rozwoju i Przedsiębiorczości) and was a part of the scientific teams working on various computer vision problems including image watermarking, biometrics (facial anti-spoofing) and virtual object placement:

- *Vestigium - a platform using intelligent video content watermarking algorithms enabling fast identification and blocking of the source of* *illegal transmissions* (pol. *Vestigium — platforma wykorzystująca inteligentne algorytmy znakowania treści wideo umożliwiające szybką identyfikację i blokowanie źródła nielegalnych transmisji*) from 05.06.2020 to 30.11.2020, project number: POIR.01.01.01-00-1032/18,

- *Adspective - The platform automating the creation of high-quality advertising in multimedia content using AI* (pol. *Adspective - Platforma automatyzująca tworzenie wysokiej jakości reklamy w treściach multimedialnych z wykorzystaniem AI*) from 05.01.2022 to 30.06.2023, project number: POIR.01.01.01-00-1531/20,

- *Vestigit Watermark on the Edge - a platform to secure streaming in multi-CDN environments with the usage of Content Adaptive Fast Watermark (CAFW)* (pol. *Vestigit Watermark on Edge - Platforma do zabezpieczeń streamingu w środowiskach multi-CDN z wykorzystaniem Content Adaptive Fast Watermark (CAFW)*), from 01.03.2023 to 30.06.2023, project number: POIR.01.01.01-00-0090/22,

- *Development and real-world verification of a biometric liveness verification system for remote identity verification processes.* (pol. *Opracowanie i weryfikacja w warunkach rzeczywistych systemu biometrycznej weryfikacji żywotności obiektu (ang. liveness) w ramach zdalnych procesów weryfikacji tożsamości.*), from 01.07.2023 to 30.11.2023, project number: POIR.01.01.01-00-0169/22,

- *Development of a system for automatic remote verification of identity documents, providing multi-faceted information security* (pol. *Opracowanie systemu do automatycznej, zdalnej weryfikacji dokumentów tożsamości, zapewniającego wieloaspektowe bezpieczeństwo informacyjne.*), from 01.01.2024 to 18.06.2024, project number: FENG.01.01-IP.02-0761/23.

Chapter 2

Related work

The following chapter contains definitions and motivations related to the problem addressed in the dissertation.

2.1 Speech synthesis

Speech synthesis is a field of study that focuses on the exploration of algorithms, models, and techniques aimed at generating artificial speech.

2.1.1 History

We trace the early development of speech synthesis back to the late 18th century [18]. In 1779, Christian Gottlieb Kratzenstein created a mechanical device composed of acoustic resonators that mimicked the human vocal tract by activating the resonators with vibrating reeds. His apparatus could artificially produce the vowels /a/, /e/, /i/, /o/, and /u/. Some of the earliest attempts at electrical-based speech synthesis were made in the 1930s when Homer W. Dudley invented a mechanical device called ”Voder” [19]. This keyboard-operated electronic mechanism was able to synthesize intelligible speech. The first computer-based speech generation techniques emerged in the 1950s and 60s — in 1961, John Larry Kelly Jr. and his team in Bell Labs used an IBM 704 computer to synthesize speech in the song *Daisy Bell*. It was later used in the screenplay of *2001: A Space Oddysey* [20]. The advancements in the 1980s and 1990s were achieved mainly using a technique called Linear Predictive Coding (LPC) [21], which became a standard for computer-generated speech. This technology became widely known through products such as *Speak & Spell* [22]; the devices, based on Texas Instruments LPC Speech Chips, were hand-held educational computers for children, which, by synthesizing words, helped them learn proper pronunciation. In the 90s, the development emphasized improving the synthesized voices regarding naturalness and speech intelligibility. At that time, the first female speech synthesizer was developed [23], marking a shift from the previously developed systems that exclusively featured masculine voices. The popularity of TTS systems rose as they were included in multiple computer programs — e.g., in 2000, Microsoft released *Microsoft Narrator* [24], a screen reader accessibility feature bundled with every version of their operating system. The rise in popularity of speech synthesis tools has also resulted in the development of products for languages other than English. One of them was *Ivona Software*, the Polish TTS system, which was later used to create the *Alexa* assistant [25] and is still used in public spaces like railway stations [26]. In recent years, progress in artificial intelligence has profoundly influenced the development of speech synthesis techniques. The neural network-based approaches have increased the quality and naturalness of the generated speech and made this technology far more accessible. Current approaches enable voice synthesis using several minutes of source material or synthesis of low-resource languages [27] — applications that were earlier not possible. Thanks to these results, deep learning became the current standard for generating artificial speech [28] — both in academic work but also in industry applications (see Sections 2.1.2 and 2.1.3).

In this dissertation, we cover various speech synthesis methods, as the means of creating new utterances will be considered possible attacks we detect.

2.1.2 Applications

Speech synthesis methods have many applications, making them essential in many aspects of contemporary human life. One of its most significant benefits is its ability to help people with speech impairments. Products like *Google Parrotron* [29], via the use of TTS systems, can improve the intelligibility of speech of deaf speakers. Such methods are also used in accessibility features. Synthesizing text can provide individuals with visual impairments access to content such as e-books or websites. Visual interfaces can become accessible by describing information on the screen using audio (e.g., by telling which button the user is currently hovering over) [24]. Speech synthesis is an increasingly popular method for creating audiobooks [30]. TTS has allowed products such as voice assistants to emerge. Siri [31], Alexa [32] or Google Assistant [33] use synthesized speech to respond and engage in natural language interactions with users. They can deliver information and control devices responding to the user's commands. Voice feedback is also used in navigation systems [34, 35]. Users are presented with visual information about routes or traffic situations and receive audio feedback, which helps improve safety due to not having to look at the navigation screen. In education, speech synthesis can be used in language learning applications. By listening to the speech created using the voices of native speakers, learners can improve their pronunciation and language skills [36, 37, 22]. Customer service also benefits from speech synthesis. The menu for selecting the conversation topic is often automated using synthesized speech before connecting callers to a human agent, resulting in lower operational costs. Moreover, some hotlines utilize chatbots communicating using synthesized speech, enabling performing certain operations without a human operator [†].

2.1.3 Products and toolkits

At the time of writing the dissertation, there exists a variety of companies offering products addressing particular demands of the market, e.g., Coqui.ai [38], Resemble.ai [39], ElevenLabs [40] or Respeecher [41]. In addition to these startups, major industry leaders such as Amazon [42], Baidu [43], Google [44] or Microsoft [45] provide their own cloud-based TTS services. The open-source voice synthesis community is also thriving — academia and open-source fields are supported by several free toolkits like Coqui-TTS [46], ESPNet [47], Hugging Face [48] or SpeechBrain [49] which contain open-source implementations of many of the state-of-the-art methods. The quality of the implemented methods is often comparable to the closed-source products [50]. The multitude of these solutions has significantly facilitated access to speech synthesis technology. Nowadays, these methods are no longer reserved only for research or industrial centers — more and more people can benefit from these technologies.

2.1.4 Taxonomy

We distinguish two main speech-generating approaches: Text-To-Speech (TTS) and Voice Conversion (VC). Text-To-Speech refers to the algorithms that generate artificial speech based on the provided text inputs (see Fig-

[†]Example: Max, the voice assistant of Orange https://www.orange.pl/view/max.

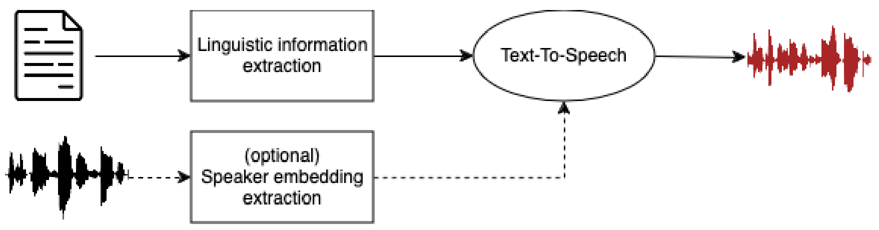

Linguistic information extraction
Text-To-Speech
(optional) Speaker embedding extraction

Figure 2.1: In the general pipeline of a Text-To-Speech (TTS) system, speech is synthesized based on provided input text. Modern TTS systems enable the synthesis of multiple voices, languages, emotions, and prosody based on additional information. Figure depicts system based on speaker embeddings — information extracted from the reference samples that allow generation of multiple voices using a single model.

ure 2.1). Such models are trained on the speech samples and their corresponding transcripts. They use the provided text to generate (mel-)spectrogram of the corresponding utterance [51, 52, 53, 54, 55], which is later converted into an audio signal using neural vocoders [56, 57, 58, 59, 60, 61, 62]. As a result, TTS models can generate arbitrary utterances based on the input transcript provided by the user. Recent TTS solutions perform this process end-to-end using a single model [63, 50]. Moreover, novel solutions support generating multiple voices [64, 65, 50, 63, 66, 67, 68, 69], languages [50, 67] or emotive speech [38].

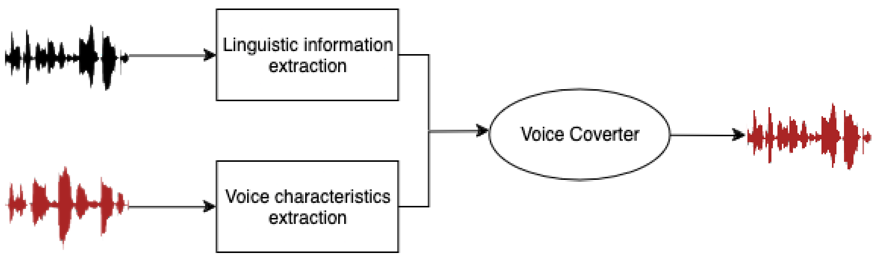

Linguistic information extraction
Voice characteristics extraction
Voice Coverter

Figure 2.2: The general pipeline of a Voice Conversion (VC) system — speech is synthesized based on two utterances used for extracting linguistic information and voice characteristics.

Voice Conversion covers the algorithms transforming one voice to another person's voice without changing the utterance's linguistic content (refer to Figure 2.2). By disentangling the content and the speaker information of a given utterance, these methods retrieve source content data from one utterance and speaker information from the other. As a result, they can reconstruct the original statement spoken by someone else. Similarly to the TTS methods, VC models can be either two-stage (synthesizer and vocoder) [70, 71], or end-to-end [72, 50, 63]. Due to their nature, classic VC methods are limited by the content of the original utterances. They cannot generate new statements (i.e., they do not contain any further text information). However, while a clear distinction exists between the TTS and VC algorithms, multiple methods incorporating both TTS and VC overcome these obstacles — they are referred to as Voice Cloning [50].

2.2 Audio Representations

The following subsection describes selected, popular, representations of audio signals used in speech processing. One uses different representations due to the features they emphasize (or dismiss) and due to the decreased dimensionality of the represented data. For more examples of audio representations, please refer to [73].

2.2.1 Waveform representation

Waveform representation is the most basic way to represent the audio signal. It uses a sequence of numbers x_t describing a relative air pressure (amplitude) with respect to a given time point $t \in \mathbb{N}$. This digital representation of the analog phenomena of sound waves is obtained using pulse-code modulation (PCM) technique [74].

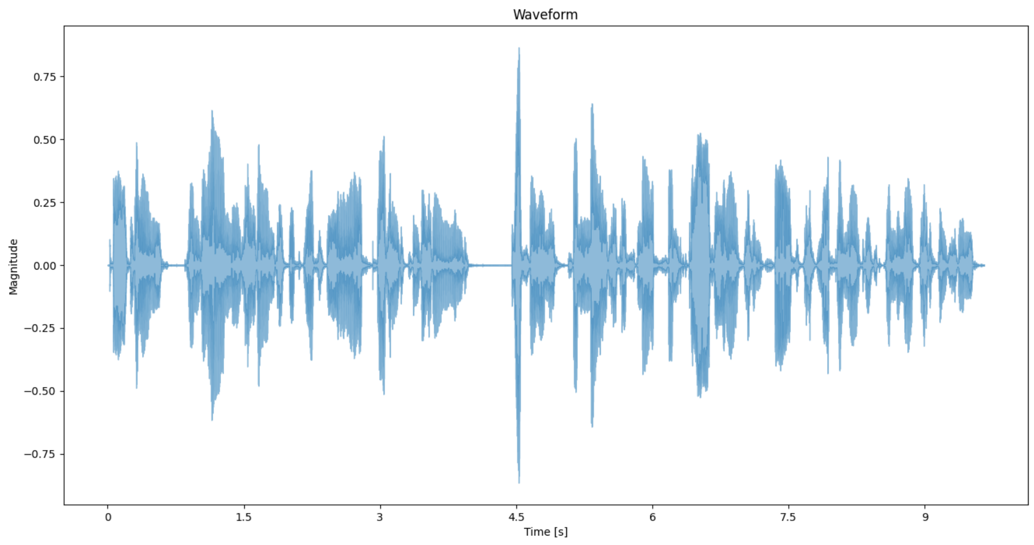


Figure 2.3: Waveform representation of sample LJ001-0001 of LJSpeech dataset [75].

The accuracy of a sound waveform's reconstruction is mainly affected by sampling rate and bit depth. The sampling rate determines the frequency at which a continuous signal (sound wave) is sampled to create its discrete representation (waveform) — higher values enable a more precise representation. Sampling rates of 16,000 Hz, 22,050 Hz, and 44,100 Hz are standard for speech processing (higher values are typically considered for non-speech audio). According to the Nyquist-Shannon sampling theorem, to properly represent a signal, its frequency should be below the Nyquist frequency (calculated as $f_n = \frac{f_s}{2}$, where f_s is a sampling rate of a signal) as such audio does not contain distortions known as aliasing [19].

The bit depth specifies the number of bits used to represent the amplitude of a signal, which can be interpreted as the resolution of the signal's loudness. A higher bit depth increases the dynamic range, allowing for a more precise representation of sound. Most speech processing applications typically employ a bit depth of 16 bits or higher [74].

2.2.2 Spectrograms

Spectrograms are another way to visually represent audio signals by displaying the frequency information over time. Creating this representation starts with dividing the waveform into small (e.g., 400 values) overlapping (e.g., 160 values) segments called frames. We then perform windowing — multiplication of the signal values by a window function (e.g., Hann, Hamming). This operation aims to reduce spectral leakage, resulting in inaccuracies in representing frequency components of the original signal. We then perform Short Time Fourier Transform (STFT) by applying Fast Fourier Transform (FFT) on each window separately to shift from time to frequency domain (input signal x_n and window w_n).

$$\mathrm{STFT}x_n(h,k) = X(h,k) = \sum_{n=0}^{N-1} x_{n+h} w_n e^{-i2\pi \frac{kn}{N}}$$

The results are aggregated into a matrix of complex numbers comprised of magnitude and phase information X_k. Final representation is obtained by computing magnitude ($|X_k|$, where $k \in \{0, ..., K-1\}$ are FFT coefficients), or square magnitude ($|X_k|^2$), however the most commonly we compute $20\log_{10}|X_k|$ — the spectrum that is described in decibels allows easy interpretation [74]. Visualizing the obtained real-valued matrix as a heatmap provides us with a spectrogram. Spectrograms and other 2D audio representations are often referred to as front-ends — features that are later processed by various speech-related solutions.

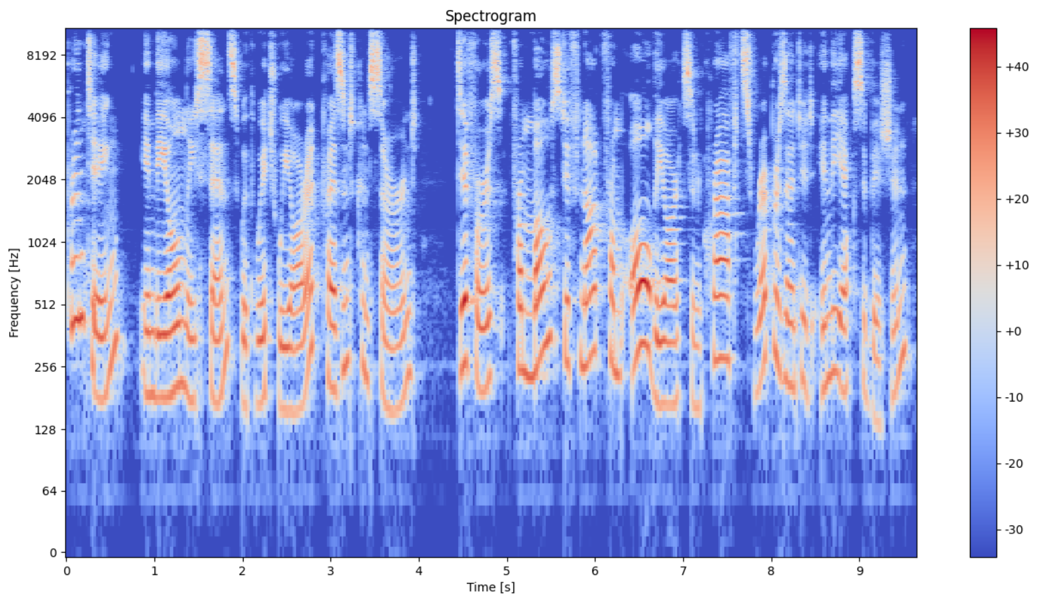


Figure 2.4: The x-axis of the spectrogram represents time, whereas the y-axis is the frequency (e.g. in Hz). The color denotes the frequency's amplitude (or power) and is measured in decibels. Sample LJ001-0001 of LJSpeech dataset [75].

Humans do not perceive frequencies on a linear scale — the differences in lower frequencies are much more noticeable than those from higher frequencies. Mel-scale is a non-linear transformation that reflects human response to different frequencies. To convert frequency f to a mel-scale, we use the following equation:

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

Mel transformation is used to create representations like mel-spectrograms, which are utilized in tasks that require better alignment with human perception of a sound, e.g., speech processing or music genre classification. Mel-spectrogram is created by applying mel filter banks to a standard spectrogram.

2.2.3 Linear and mel frequency cepstral coefficients

Mel frequency cepstral coefficients (MFCC) are a representation used in tasks like the speaker or speech recognition [76, 77]. MFCC are based on mel-spectrograms — we first compute its logarithm, apply a Discrete Cosine Transform (DCT) and compute amplitudes of the obtained spectrum. This representation is more abstract than (mel-)spectrograms, which makes it more difficult to interpret. However, as they can capture essential speech-related information [74, 76], they became a standard for speech processing.

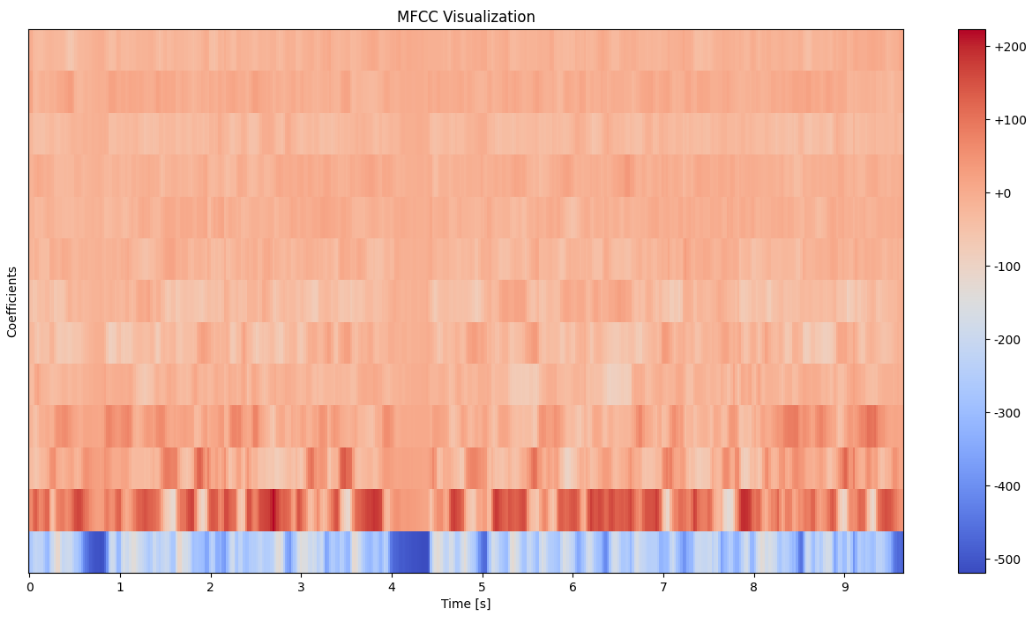


Figure 2.5: An example of a MFCC representation (13-coefficients) based on the LJ001-0001 sentence from LJSpeech dataset [75].

Linear frequency cepstral coefficients (LFCC) are features created similarly to MFCC, but instead of using mel-filter banks, they utilize a linear scale. Despite being less commonly used, some research shows that using LFCC in some speech-related tasks (e.g. speaker recognition or baby crying detection) provides similar or even better results over mel-scaled features [78, 79], which makes it an interesting alternative to MFCC. Apart from MFCC and LFCC, there exist other cepstral coefficient representations [80, 81].

2.3 DeepFakes

The field of speech synthesis is not the only branch of generative AI currently experiencing rapid and dynamic development. Deep learning techniques [82] primarily led to enhancements over the traditional algorithmic methods in fields like image, video and text processing. This new era has resulted in high-quality modified and synthetically generated content.

Synthetic content created using deep learning methods is called DeepFakes (DF). The term *deep* denotes using deep learning as a creation technique, whereas *fake* highlights the falsity of the generated content. DeepFakes initially referred to algorithms for manipulating facial images to create fake identities [83, 84]. The original algorithm [85] swapped the face of the person in the video with the face of the arbitrary individual. The resulting image kept the mimics and illumination of the source while depicting another person. The initial method is based on two autoencoder neural networks [86]; this technology has been significantly enhanced in the following years. The improvements were both in the context of expanding the original idea itself through modification of individual components (e.g. increasing output sizes [87]), as well as through the use of new, different technologies such as Generative Adversarial Networks [88, 89, 90] (with their extensions like CycleGANs [91, 92]) or Diffusion Models [93, 94, 95].

The advancements also introduced new DeepFake generation tasks such as facial reenactment. The idea covers manipulating the mimics of the original person in an image based on some external input (mimics of another individual) [96, 97, 90, 98, 99]. The output contains the original individual but with the changes in their facial and head movement. The results are often realistic and convincing (see Figure 2.6). Nowadays, the name DeepFake is often used to describe various methods related to the generation or modification of biometric features [100, 84] including topics like face editing (modification of the features such as hairstyle, age, facial expression [101]) or face synthesis (creating faces of non-existent individuals [102, 103]).

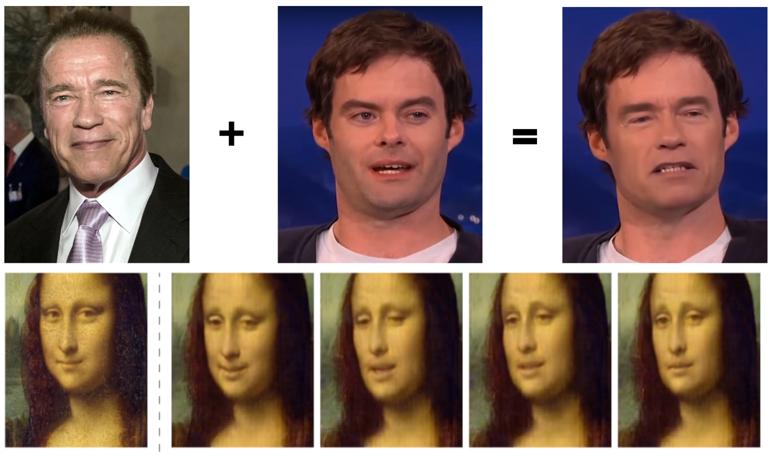


Figure 2.6: Originally, *DeepFakes* refers to the set facial manipulation algorithms that include both identity swap and expression (facial) reenactment. Source: [98].

Term *DeepFakes* also refers to a phenomenon existing in audio and speech processing. Generating utterances using Text-To-Speech and Voice Conversion (described in Section 2.1.4) is referred to as generating *audio DeepFakes*. The recent development of these methods significantly lowered the quantity of data required for generating realistic utterances. The end-to-end nature of the majority of the DeepFake generation techniques (both audio and video) significantly facilitates the creation of forged materials — preparing them no longer requires specialized knowledge and many hours of tedious work [28]. Nowadays, systems like Personal Voice in iOS smartphones [104] require about 15 minutes of speech to mimic the voice of the owner realistically. In addition, many services offering voice synthesis functionalities (see Section 2.1.3) make creating this type of content even more widely available.

Multi-modal DeepFakes is another branch of DeepFakes that is currently becoming increasingly popular. These solutions combine different modalities, such as vision and audio, creating spoofed materials consisting of consistent faces and voices. The most notable approach to combining these modalities is *talking heads*. They are videos featuring faces with expressions driven by audio input [105, 106]. This technology is currently employed in AI avatar solutions [107].

Audio DeepFake algorithms, despite having numerous beneficial applications (see Section 2.1.2), also pose significant security threats due to their ability to produce high-quality fake content that can be difficult to distinguish from genuine recordings. These malicious applications are a cause for concern.

One of the most concerning applications of audio DeepFakes is the creation of fake news. By impersonating authoritative figures such as politicians or celebrities, fake materials can damage their reputations and mislead the recipients of such content [108]. Moreover, their ability to influence political or military decisions by, for example, creating a DeepFake of the president ordering troops to surrender highlights the potential for widespread chaos and misinformation.

Another malicious scenario involves bypassing automatic speaker verification systems. Voice biometrics is increasingly popular and is currently used as an authentication measure in many systems like telecommunications or banking services. Positive verification of the spoofed utterance provided by an adversary grants them access to the system [109].

Extortion is another possible attack vector — audio DeepFake methods were successfully used to impersonate the transaction participants to order wire transfers to the adversary's account, resulting in millions of dollars in loss [2, 3]. Moreover, synthesized speech can also be used to fool family members by asking for immediate financial help.

Furthermore, DeepFakes can also be used for blackmailing and defamation. By creating ridiculing materials, someone can threaten to release the fake content publicly and tarnish someone's reputation [4].

2.3.1 Human perception of audio DeepFakes

Despite their high quality, modern speech synthesis methods are still unable to perfectly imitate every aspect of human speech. They face problems such as imperfect reproduction of the characteristics of someone's voice or inconsistencies in the prosody of speech. Despite that, the examples highlighted in the previous paragraphs show that DeepFakes, if used in bad faith, pose a great danger to various aspects of human life. They are not isolated cases: in recent years, studies have been done to determine an average citizen's capabilities to distinguish real and spoofed speech.

The authors of [110] created a game-like challenge in which participants competed with DeepFake detectors in distinguishing DeepFake and *bona fide* utterances. The reported results were based on 472 participants. The group was diverse in terms of spoken language (105 of them were English native speakers), age (20-100), and IT skills (self-reported on a scale from 1 to 5). Each participant listened to several recordings that they could replay multiple times — once they were ready, they were supposed to choose if the given sample was fake. After each decision, they were informed of the correct answer.

While the models detected 95% of fake utterances, humans were only able to detect 80% of them. This level of performance is weak and alarming, which the authors support with the following observations. Primarily — the participants were aware of the task and could replay the recordings multiple times. Such a controlled environment is much easier than a real-life scenario when a victim may not be aware of the malicious activity. Moreover, the recordings presented to the participants came from the ASVspoof 2019 Challenge [111] — in the last couple of years, the progress in speech synthesis has been profound, and utterances generated with current technology sound even more convincing. The scientists additionally observed that the performance of detecting DeepFakes declined with age and that there exists no significant correlation between IT skills and the capability to detect spoofed speech — DeepFakes are a threat to everyone.

The human ability to recognize DeepFakes was also covered in [112]. The authors conducted a survey where the participants were presented with several recordings as a part of the survey on the usability of voice messages in chat apps. This was a cover-up story, as the true nature of the experiment was the assessment of human DeepFake detection capabilities. Participants were supposed to select one out of three recordings containing false information. Moreover, they were instructed to report anything unusual about the audio files they listened to. The participants were, however, unaware that some of the presented recordings were artificially generated using state-of-the-art speech synthesis systems.

The research was performed on 31 respondents, with the majority of them being younger than 23 years old, about 40% working in IT and 85% aware of, or even interested in, the topic of DeepFakes. Out of all participants, only one suspected that some of the utterances were artificially generated; 13 respondents mentioned that some of the samples were of a slightly worse quality, but their answers did not indicate their suspicion about the genuineness of the speech. The authors point out that the reason for such weak results is that the participants were focused on analyzing the information in the recordings and were unaware of the presence of DeepFakes. As the participants did not suspect such manipulations, they ignored the interferences related to the existence of DeepFakes. Once they were informed about the true nature, 84% of them could correctly identify spoofed samples — performance comparable to the one presented in previous research [110].

Both experiments confirm that the malicious use of DeepFake techniques should be treated as a significant threat. The experiments show that people often cannot distinguish between real and fabricated samples. The performance is much worse when they do not expect that a given recording may be fabricated. Synthesized speech is often used to create controversial and shocking statements and can be presented (replayed) to the individuals under the influence of emotion (e.g., stress). The materials are often disseminated on social media or during phone calls during stressful situations — the compression of the uploaded recordings or environmental noises during the call are the factors that further reduce detection capabilities.

2.3.2 DeepFake detection

Due to the seriousness of the aforementioned threats posed by DeepFake technology, the scientific community began research on the topic of assessment if a given utterance was artificially modified. This task is referred to as *audio DeepFake detection*. The problem of detecting DeepFakes is a no-reference task, which means that one has to assess the validity of the sample based only on the analyzed material with no access to reference utterances (refer to Figure 2.7).

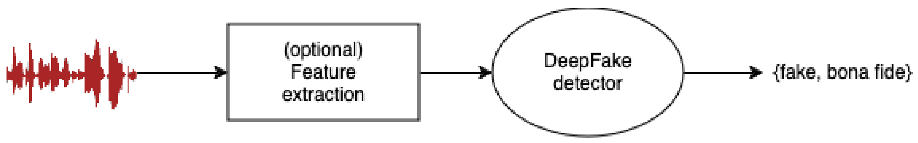


Figure 2.7: DeepFake detection task determines the genuineness of the audio sample. The process aims to find the artifacts introduced in the synthesis process.

A variety of factors influence the difficulty of the Deepfake detection task. The quality of the generated utterances is among the most significant. Samples of lower quality typically contain more artifacts that can be used to distinguish spoofed and bona fide speech. In systems like two-stage TTS or VC methods (see Section 2.1.4), the imperfections can be introduced either at the spectrogram synthesis stage or during the vocoding step. In the spectrogram synthesis, that aspect can be the quality of the phoneme mapping — how well the spectrogram created by the synthesizer module of TTS reflects the words included in the input transcript. The vocoding part of the process refers to transforming the synthesized spectrogram to the speech — the vocoding quality can suffer from the artifacts, including robotic (metallic) voice, muffled (or blurred) speech, or clicking (popping).

Other factors concern environmental interferences like distortions in telephony systems or background noises like traffic, white noise, or background music. The DeepFake manipulations are mostly spread through the Internet — the samples are often compressed when uploaded to social media platforms. Such manipulations strongly affect the final results of the prediction as the analyzed samples differ significantly from those seen by the model in the training pipeline. In addition, such degradation can effectively hide the artifacts used to distinguish bona fide and fake utterances [113].

2.3.3 Performance metrics

The problem of DeepFake detection is most often considered in the context of a binary classification task. In order to determine the performance of the detection methods, we typically use metrics known from the biometrics field. The following convention is used in the dissertation: bona fide samples are referred to as positive samples, while manipulated samples are referred to as negative. Therefore, we interpret True Positive (TP) and True Negative (TN) predictions as consecutively properly predicted pristine and DeepFake examples. Analogously, False Positive (FP) and False Negative (FN) samples are incorrectly assigned bona fide and DeepFake labels.

The most widely utilized metric is the Equal Error Rate (EER) [114]. It is commonly used in biometric tasks such as face recognition, speaker verification, or DeepFake detection. It comprises two quality factors: False Acceptance Rate (FAR) and False Rejection Rate (FRR) [114]. For the sake of completeness, let us recall them.

FAR (type I error) — refers to the percentage of the fake inputs incorrectly accepted as bona fide:

$$\text{FAR} = \frac{FP}{FP + TN}.$$

FRR (type II error) — measures the percentage of bona fide samples incorrectly classified as fake:

$$\text{FRR} = \frac{FN}{FN + TP}.$$

EER denotes the point where the values of FAR and FRR errors are equal (see Figure 2.8). EER values range from 0 to 1, where 0 denotes perfect performance with no false acceptances or false rejections, 0.5 denotes random choice, and 1 denotes completely wrong predictions.

In addition to EER, another used binary classification metric is the Area Under the ROC Curve. ROC Curve is a graphical representation of the performance of classification models considering all classification thresholds. ROC curve is created by plotting two metrics — True Positive Rate (TPR) and False Positive Rate (FPR).

$$\text{TPR} = \frac{TP}{TP + FN},$$

$$\text{FPR} = \frac{FP}{FP + TN}.$$

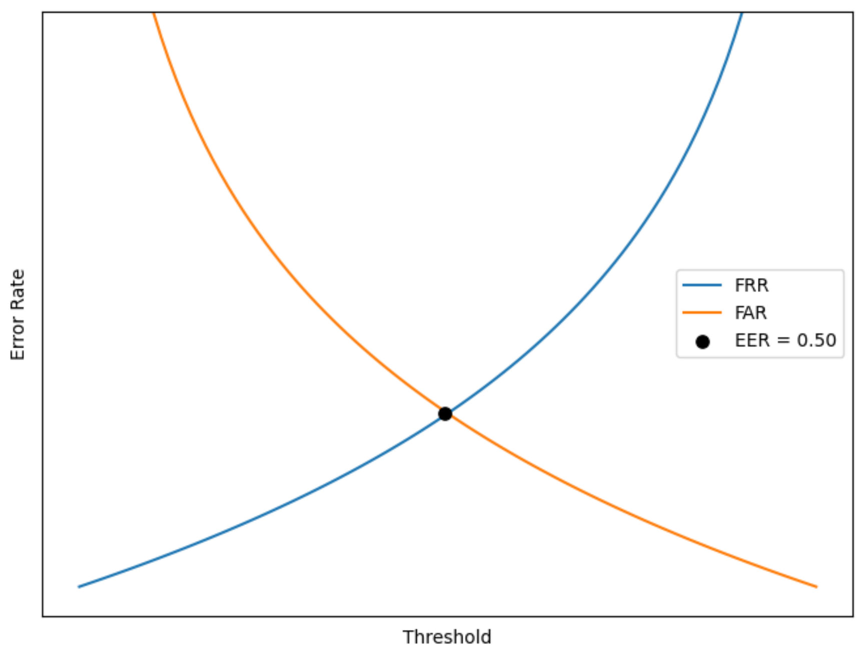


Figure 2.8: Visualization of Equal Error Rate. Changing the classification threshold influences False Acceptance and False Rejection Rates — choosing the right threshold value depends on the characteristics of the system.

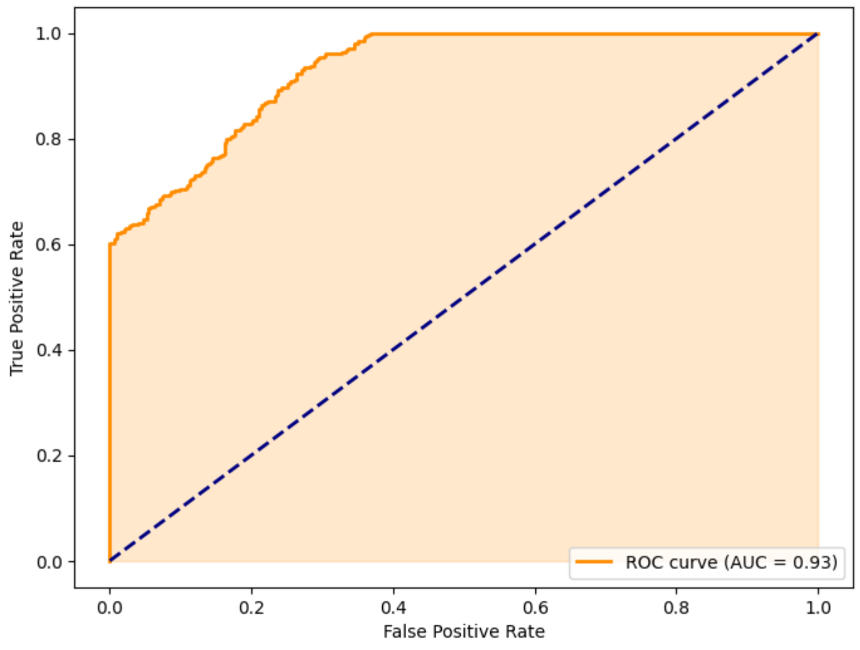


Figure 2.9: ROC curve and Area Under the ROC curve allow us to analyze the classifiers' performance concerning the used thresholds. Dashed diagonal line refers to the random predictions.

With the ROC curve, it is possible to determine how different decision threshold values affect the model's performance. Area Under the ROC curve (AUC) is the area under the ROC curve used to assess the classifier's quality (see Figure 2.9). AUCs take values ranging from 0 to 100%, where 100 means 100% TPR and 100% FPR; an AUC of 0 means no correct prediction at all, while 50% is random prediction. Alternatively, AUC can be expressed using $[0, 1]$ range.

All of the results presented in the dissertation follow the range of values $[0, 1]$ for EER and $[0, 100]$ for AUC.

2.3.4 Voice authenticity taxonomy

Within the field of voice authenticity attacks, we distinguish three primary categories (scenarios): physical access spoofing, logical access spoofing, and DeepFakes. Spoofing and DeepFakes differ in their intended targets. Deepfakes are artificial utterances deceiving human listeners and used, for instance, to spread disinformation, e.g. via social media, whereas spoofing seeks to compromise biometric identification (automatic speaker verification) systems. Distinct attack methods are characteristic of each field, highlighting the differentiation between them as separate areas of study.

We further divide spoofing into logical and physical access spoofing. Physical access spoofing primarily involves deceiving physical automatic speaker verification (ASV) systems through replay attacks. In these attacks, utterances are initially recorded and subsequently replayed to the system using various devices in different acoustic and location environments [115, 116, 117, 111, 118]. Logical access spoofing involves remote attacks targeting ASV systems. In these scenarios, spoofed audio, generated through Text-To-Speech and Voice Conversion methods, is directly injected into the communication channel, thereby circumventing the microphone [115, 119, 111, 118].

2.3.5 Detection methods

The task of audio DeepFake detection is a rapidly growing field of study. In the following section, we provide information on the most commonly used solutions.

Gausian Mixture Model (2018) The first methods for detecting fake utterances were based, among others, on the algorithmic (statistical) approaches, e.g. Gaussian Mixture Models [120]. Gaussian Mixture Model assumes that the signal may be represented as a sum of simple signals with Gaussian distribution, each with possibly different parameters (mean and standard deviation). Hence, the composed signal can be approximated and classified by the sum of such simple Gaussians. DeepFake detection systems are typically trained using two GMMs — one to model the distribution of bona fide samples, the other for the spoofed data. The data is presented to GMMs using front-end features like LFCC [121].

Recently, both the DeepFake generation and detection techniques are based on deep neural networks.

MesoNet (2018) [11] is a neural network architecture originally created for visual DeepFake detection; when applied to audio processing, this model processes front-end representations. The authors introduced two variants — Meso–4 and MesoInception–4. Meso–4 starts with four blocks composed of 2D convolution, ReLU activation, batch normalization and max pooling. It is followed by a fully–connected layer with a Leaky ReLU activation, a dropout, and a final output layer. MesoInception–4 is a variation that replaces two first blocks with inception modules [122]. Despite the minimal number of parameters — 27,977 and 28,615 for respectively Meso–4 and MesoInception–4, this family of networks achieves appropriate performance in many tasks. All work described in the following dissertation is based on the MesoInception–4 architecture.

LCNN (2021) is an architecture of neural network originally published as a face recognition method [123], which was later adapted to the problem of speech anti-spoofing [124] and is currently one of the most popular classifiers. Several versions of this architecture differ in number of blocks and use of recurrent layers. The general scheme of the recurrent-LCNN looks as follows: it operates on 2D features (front-ends), which are later processed by several blocks composed of convolutions, Max-Feature-Map layers, batch, normalization and max pooling. The signal is later passed to two bidirectional LSTM layers, and a fully connected layer then processes the final information. Instead of using typical activation functions like ReLU, the authors expand the concept of max out activation function [125] and introduce the Max-Feature-Map (MFM) operation. The operation aims to select the optimal feature map at each location learned by the convolutional filters. For that, MFM uses element-wise maximum providing a binary gradient to

”excite or suppress one neuron during back propagation” [123] as a part of the feature selection process.

RawNet2 (2021) is a neural network architecture originally used in speaker verification [126]. It was adapted to the task of the audio DeepFake detection in [127]. The model processes the raw audio signal. The architecture starts with a SincConv module [128] — signal is first filtered with sinc functions [129] and later processed by a convolutional layer. The model is later composed of two groups of residual blocks: signal is first processed by two groups of batch normalization and LeakyReLU activation and ends with downsampling the information with max pooling and feature map scaling (FMS) [126] (used as an attention mechanism). Both residual block groups differ only in the number of convolutions (128 in the first one and 512 in the second one). The extracted information is later processed using a Gated Recurrent Unit (GRU) module [130] and passed to a fully connected layer, and the final output layer is comprised of two neurons. SpecRNet, a neural network introduced in Chapter 4, is based on RawNet2 archicture.

The following list includes other notable audio DeepFake detection methods that are not directly utilized in the later chapters.

RawGAT-ST (2021) [131] is a raw audio spectro-temporal graph attention network (GAT). The architecture utilizes the research indicating that spoofing artifacts are present in specific sub-bands or temporal segments of audio files [132, 133, 134]. Thanks to the fusion of the spectral and temporal graphs at the model level, the network can learn relationships between spectral and temporal artifacts, leading to high performance. RawGAT-ST operates on raw audio waveform and contains about 440,000 parameters.

AASIST (2022) [135] is an extension to the RawGAT-ST architecture. Similarly to its predecessor, it focuses on the spectral and temporal nature of audio spoofs. The authors propose a series of enhancements to the processing and fusion of spectral and temporal information, including heterogeneous stacking graph attention layer and max graph operation. The default version of AASIST comprises around 297,000 parameters, whereas the lightweight version (AASIST-L) contains only 85,000 parameters.

wav2vec2.0 DF (2022) [136] is an extension of the AASIST architecture by replacing the original front-end of sinc-layer [128] with wav2vec2.0 [137] Self-

Supervised Learning model [138]. Authors additionally trained the models using the RawBoost data-augmentation technique [139], which, in summary, resulted in up to 90% performance improvement in relation to the original AASIST architecture.

2.4 DeepFake detection datasets

The problem of detecting audio DeepFakes started to be widely investigated significantly later than its visual counterpart. Since the introduction of visual DeepFakes in 2017, many new detection datasets have been introduced. Throughout the years and iterations of the datasets, many issues were addressed. They included problems like a small number of generation methods, gender, age, or ethnicity bias, or unclear legal matters of the samples in the dataset. Despite the dynamic growth of the audio DeepFake detection field, during our work on *Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection* [6], the only data commonly used by the scientific community was found in ASVspoof Challenges (2017 and 2019 [116, 111]). In recent years, several new datasets emerged (see Table 2.1), showing the community's significant interest in detecting artificial speech.

2.4.1 ASVspoof

ASVspoof [140, 119, 116, 111, 118] is a community-driven initiative that started in 2013; from 2015, it is a bi-yearly challenge focused on detecting audio spoofing. The competition is one of the most notable initiatives in the speech processing community, setting the standard in anti-spoofing and DeepFake detection — the datasets introduced in the challenges are a golden standard in the field. The initial installment, ASVspoof 2015 [119], concentrated on investigating countermeasures to detect TTS and VC attacks. In ASVspoof 2017 [116], the focus shifted to replay spoofing attacks and the corresponding countermeasures. Notably, the ASVspoof 2019 [111] was the first to include all three types of spoofing attacks in a single challenge. These attacks were generated from the same source database and adhered to the same underlying protocol, but they were examined within two distinct usage scenarios: samples from the logical access (LA) subset were created using cutting-edge TTS and VC techniques. Conversely, controlled simulations in real physical spaces were used to generate replay spoofing attacks of a physical access (PA) scenario. ASVspoof 2021 [118] introduced a new task of DeepFake speech detection (DF) — a collection of utterances processed using different lossy codecs to reflect the scenario of media access through the Internet. This installment was the first that did not include any new training and validation data. The reason for that was to reflect real-life scenarios of not being able to predict the nature of spoofed data with high confidence.

The following part details the subsets of various ASVspoof installments used throughout the dissertation.

ASVspoof 2019 LA

The samples of logical access scenario of ASVspoof 2019 [111] are derived from VCTK [141] base corpus. The dataset consists of 12,483 bona fide and 108,978 fake English samples — generated using 19 systems (labeled A01-A19). The attacks include text-to-speech (TTS), voice conversion (VC), and hybrid algorithms (VC with synthetic speech as input). The samples are split into training, development, and evaluation subsets comprising of respectively "20 (8 male, 12 female), 10 (4 male, 6 female), and 48 (21 male, 27 female) speakers"' [111].

ASVspoof 2021 LA Eval

The 2021 installment of the ASVspoof challenge did not contain any new training or development data — the participants were expected to utilize the 2019 dataset. The evaluation data was split into three types: progress (used for intermediate assessment), evaluation (used for final evaluation) and hidden subsets (for additional evaluation). The progress and evaluation subsets were composed in summary of 16,464 real and 148,148 fake samples containing 37 female and 30 male English speakers. The origin data was the VCTK database. The samples underwent processing through various telephony systems, including voice-over-IP (VoIP) and the public switched telephone network (PSTN). The authors specified that multiple codecs, including alaw and G.722, would be used. Fakes were generated using A07-A19 attacks from ASVspoof 2019.

ASVspoof 2021 DF

Similar to LA evaluation data, the DeepFake subset contained no new training data and was split into progress, evaluation, and hidden subsets. The dataset is comprised of 59,325 bona fide and 533,928 fake samples. The progress subset was composed of 37 females and 30 males, an evaluation of 50 females and 42 male English speakers. The origin samples were derived from the VCTK database and included 2018 [142] and 2020 Voice Conversion Challenges [143]. The fake samples were generated using over 100 different spoofing techniques [144]. The samples were created using one of nine various "codec conditions" with C1 being no codec, C2 and C3 created using mp3, C4 and C5 using m4a, C6 and C7 using ogg and C8, C9 being undisclosed.

2.4.2 WaveFake

WaveFake [145], at the time of its release (November 2021), was one of the largest audio DeepFake datasets. The audio clips featured utterances in English (LJSpeech containing 13,100 clips) and Japanese (5,000 clips of JSUT dataset). This corresponds to 18,100 pristine samples. The dataset contains 117,985 generated fake audio clips. The authors used eight methods: WaveGlow [56], Parallel-WaveGAN [61], HiFi-GAN [57], Multi-band MelGAN (with Full-band Mel-GAN) [62], MelGAN (with MelGAN Large) [60] and pipeline composed of a conformer [146] followed by the fine-tuned Parallel-WaveGAN (referred to as TTS). Please note that, except for WaveGlow, all the solutions are based on Generative Adversarial Networks (GANs). Furthermore, certain architectures are variations of others. For instance, MelGAN Large is an expanded version of MelGAN that incorporates a larger receptive field, while Full-band MelGAN is a variant of MB-MelGAN with differences in the computation of their auxiliary loss function.

2.4.3 FakeAVCeleb

FakeAVCeleb [147] is a multi-modal (audio-video) DeepFake detection dataset. It comprises samples composed of both generated speech and facial spoofing DeepFake techniques (see Figure 2.10). While facial spoofs are generated using three visual DF methods: FSGAN [90], FaceSwap [148] and Wav2Lip [149], the speech is generated using one voice cloning algorithm SV2TTS [150]. The authors state that each fake utterance is unique because they clone each of the real audio clips of their bona fide samples — first, the bona fide audio is transcribed using IBM Watson STT service [151], and then the voice is cloned with SV2TTS. FakeAVCeleb is also the first dataset that provides information on the ethnic background of the speakers — they distinguish five groups of "Black, South Asian, East Asian, Caucasian (American) and Caucasian (European)" [147]. The dataset we refer to in the dissertation as *FakeAVCeleb* is composed solely of this dataset's audio tracks, corresponding to 500 real and 11,357 fake audio clips in English.

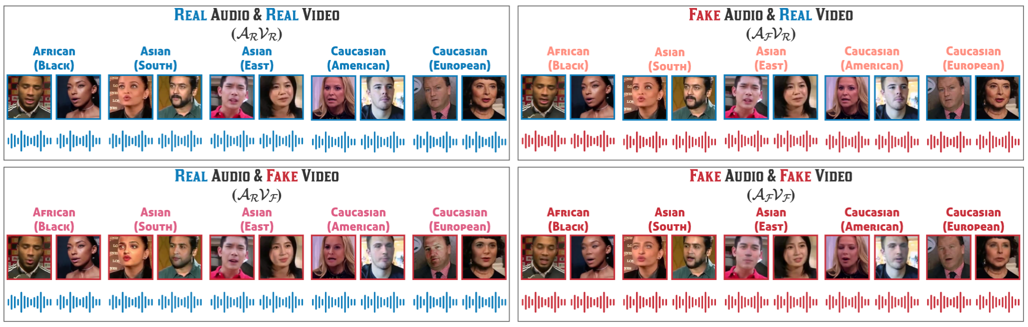

Real Audio & Real Video
($\mathcal{A}_\mathcal{R}\mathcal{V}_\mathcal{R}$)
Fake Audio & Real Video
($\mathcal{A}_\mathcal{F}\mathcal{V}_\mathcal{R}$)
Real Audio & Fake Video
($\mathcal{A}_\mathcal{R}\mathcal{V}_\mathcal{F}$)
Fake Audio & Fake Video
($\mathcal{A}_\mathcal{F}\mathcal{V}_\mathcal{F}$)
African (Black)
Asian (South)
Asian (East)
Caucasian (American)
Caucasian (European)

Figure 2.10: FakeAVCeleb dataset [147] is an example of multi-modal approach to the problem of DeepFakes containing both audio and visual manipulations. It is one of the first audio DF datasets providing information on the ethnicity of the participants. Source [147].

2.4.4 In-The-Wild

In-The-Wild dataset, introduced in [152], is a commonly used evaluation dataset. The main purpose of this dataset is to serve as benchmark data and to assess the generalization of DeepFake detection methods on real-life data. It comprises a total of 37.9 hours of English audio clips, with 20.7 hours consisting of bona fide clips and 17.2 hours of fake clips, which, in summary, corresponds to approximately 31,779 16 kHz audio samples of an average length of 4.3 s. The utterances come from openly accessible platforms, including social networks and multimedia video-sharing websites.

The samples feature recordings of English-speaking celebrities and politicians. The fake clips were generated by segmenting 219 publicly available video and audio files that state they are DeepFakes. The real samples were collected from publicly available sources such as podcasts, interviews, and speeches and featured the same speakers as in the spoofed part. The authors of the dataset took care to minimize the differences between the classes by selecting samples with a similar style, emotion and quality (e.g. background noise) to the fake ones. The DeepFake detector results on the data collected by the authors differ significantly from those reported in the literature. The best EER presented in the original paper is equal to 0.3394, making In-The-Wild one of the most difficult datasets for DeepFake detection today.

2.4.5 Other datasets

The following section describes audio DeepFake detection datasets that are not covered in the later chapters. Since the start of the author's Ph.D. in 2020, many new audio DeepFake detection datasets were introduced. However, many of the datasets described here were either available only for the challenges participants or released after the publication of the manuscripts upon which this dissertation is based [6, 7, 8, 9].

Audio Deep synthesis Detection challenge (ADD)

ADD [153, 154] is a series of challenges started in 2022. The authors aim to cover many real-life attack scenarios not included in ASVspoof 2021. They address three main issues: lack of background and real-life noises in the samples, small fake clips hidden in real utterances (partially spoofed audio) and use of cutting-edge synthesis algorithms. The 2022 installment [153] consists of three tracks: Low-quality fake audio detection (LF) containing samples with various background noises, Partially fake audio detection (PF) with partially spoofed utterances, Audio fake game (FG) where the participants (depending on the sub-track) either try to detect spoofed utterances or create samples to fool the detectors. The dataset comprises 493,123 utterances, with no information on the number of DF generators. The 2023 instalment [154] contains three tracks as well: Audio fake game (FG), Manipulation region location (RL) and Deepfake algorithm recognition. The FG track is similar to 2022; RL covers locating the exact spoofed regions in partially fake data, and AR addresses recognizing the algorithms that generated particular samples. The total number of samples is equal to 517,068.

Fake or Real (FoR)

FoR [155] is a dataset comprising 195,541 English utterances. The artificial speech was comprised of 33 voices generated using seven methods, including Amazon [42], Baidu [43], Google [44] and Microsoft TTS services [45]. The bona fide samples include the utterances gathered from the Internet, as well as well-known speech datasets like Arctic Dataset [156], LJSpeech [75], VoxForge [157].

FMFCC-A

FMFCC-A [158] is a dataset created to promote the development of anti-spoofing methods for the Mandarin language. It comprises 50,000 Mandarin utterances (10,000 bona fide and 40,000 fake) of 58 male and 73 female speakers. The synthetic utterances were created using thirtheen methods (11 TTS and 2 VC systems), including technologies like Baidu TTS [43], IBM Watson TTS [159], FastSpeech [54], Tacotron [52] or Medium VC [160].

Table 2.1: The comparison of available DeepFake and anti-spoofing datasets.

Dataset name	# methods	# utterances	# languages	Year
ASVspoof 2015 [119]	10	263,151	1	2015
ASVspoof 2017 [116]	19	18,030	1	2017
ASVspoof 2019 LA [111]	19	121,461	1	2019
FoR [155]	7	195,541	1	2019
ASVspoof 2021 LA [118]	13	164,612	1	2021
ASVspoof 2021 DF [118]	100+	593,253	1	2021
FakeAVCeleb [147]	1	11,857	1	2021
FMFCC-A [158]	13	50,000	1	2021
HAD [161]	2	160,836	1	2021
WaveFake [145]	9	136,085	2	2021
ADD 2022 [153]	N/A	493,123	1	2022
CFAD [162]	12	347,400	1	2022
In-The-Wild [152]	N/A	31,779	1	2022
ADD 2023 [154]	N/A	517,068	1	2023
PartialSpoof [163]	19	121,461	1	2023
Voc.v [164]	8	82,048	1	2023
MLAAD [12]	52	74,000	23	2024

Half-truth Audio Detection (HAD)

HAD [161] is the first anti-spoofing dataset that addressed the problem of partially spoofed utterances. This subproblem of DeepFake detection focuses on the utterances where only a part of the speech was altered; the task of the detection methods, instead of binary assessment of the authenticity, is to localize spoofed fragments. The dataset comprises two subsets of partially and fully spoofed audio and contains 160,836 Mandarin utterances. The artificial utterances are generated using Tacotron-GST synthesizer [165] and vocoded by LPCNet [166] vocoder (authors utilize two versions of this vocoder depending on the subset of the data).

Chinese Dataset for Fake Audio Detection (CFAD)

CFAD [162] is a dataset comprised of 347,400 Chinese utterances, including 1023 real and 279 fake speakers. Authors provide 12 types of spoofed audio — 11 of which are fully-spoofed utterances generated using systems like LPCNet [166], WaveNet [58], PWG [61], HiFiGAN [57], MelGAN [60]. They additionally provide partially spoofed audio. The authors provide three versions of the data: clean, under noisy conditions and after the transcoding process to achieve greater robustness of the models trained using the dataset.

PartialSpoof

PartialSpoof [163] is another dataset that focuses on the problem of partially spoofed data. The dataset is built using the ASVspoof 2019 LA challenge data and comprises 12,483 bona fide and 108,978 fake samples (generated using all 19 ASVspoof 2019 attacks). The authors consider six temporal resolutions of spoofed speech ranging from 20 ms to 640 ms.

Voc. v

The authors of [164] point out that creating voice DeepFake datasets is demanding and time-consuming. Instead of using full TTS or VC pipelines, they create a database of utterances vocoded using neural-network-based vocoders. The copy-synthesis of the bona fide samples enables the generation of a large volume of artificial data. The dataset comprises 82,048 English utterances created using eight commonly-used vocoders including [57, 62, 61, 167, 56]. The bona fide samples were extracted from ASVspoof 2019 [111]. Authors report that using only the vocoded data provides results similar to standard TTS and VC-based datasets, which makes it a promising technique in creating or enhancing new detection databases.

Multi-Language Audio Anti-Spoofing Dataset (MLAAD)

MLAAD [12] is the first multi-language DeepFake detection dataset and was co-created by the dissertation's author. It comprises 74,000 utterances with a total duration of 160 hours. MLAAD contains samples in 23 languages, being the first openly available multi-language DeepFake detection dataset.

The dataset covers popular languages like English, Arabic, Hindi, Russian, and Spanish, as well as local ones, including Polish, Czech, Estonian, Irish, Maltese, Swahili, and Ukrainian. The samples were generated using 52 state-of-the-art TTS models (22 different architectures) from Coqui.ai [38] and Hugging Face [48].

2.5 Adversarial Attacks

Adversarial attacks is a commonly used term describing the techniques for deceiving machine learning systems. They decrease their performance by adding superficial (difficult to spot by a human) changes to input data or inputting specially crafted samples. Typically, the manipulated samples are prepared using some information extracted from the neural network, e.g. gradient values or leveraging the knowledge obtained by querying the targeted model. Using such *adversarial examples* results in the incorrect predictions of models, often making the entire system ineffectual. Adversarial attacks threaten the systems of all domains, i.e. one can create these examples for problems such as computer vision, audio processing, natural language processing, or reinforcement learning.

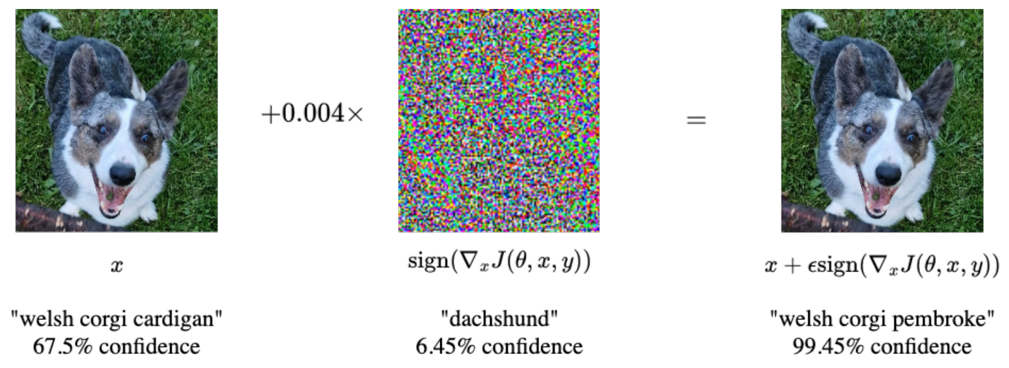


Figure 2.11: Adversarial attacks can result in faulty predictions without compromising much of the quality of the inputs.

Since introducing the concept [168], the field has been widely explored, resulting in various methods and attack models. One of the main distinctions of adversarial attacks is their objective: non-targeted or targeted. Non-targeted attacks achieve faulty prediction without specifying the desired output class, i.e., the primary objective is misclassification in any form. In targeted attacks, the adversary manipulates the input data so the model classifies it as a chosen, specific target class [169].

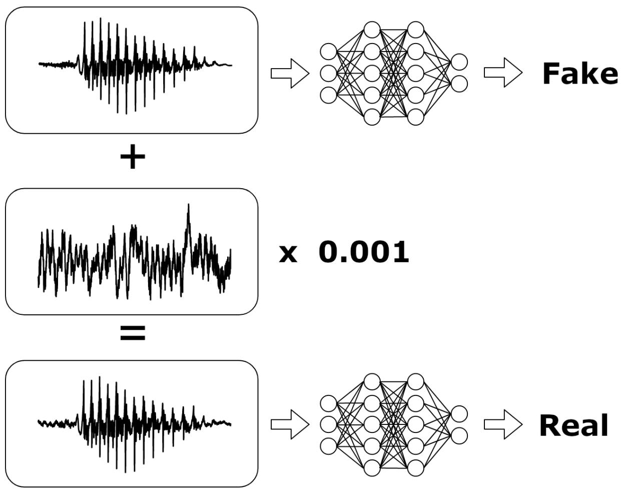


Figure 2.12: Many tasks like audio classification are threatened by adversarial attacks, which significantly decrease their performance. Source: [170]

Adversarial attacks are additionally divided according to the knowledge the adversary has: we distinguish between white-box and black-box attacks. In white-box attacks, the adversary can access the model architecture and its parameters. In such a scenario, the adversarial example can be, for instance, prepared using gradients of the corresponding sample. In a black-box attack, the adversary has limited (or no) information about the system under attack. To create adversarial samples, the adversary is forced to query the system and prepare adversarial samples based on input, output and confidence scores. Because of the high-level distinction between black- and white-box attacks, related work provides exact information on the data the adversary has access to, e.g., in a black-box setting, one may have information on the training set. The degree of the perturbations created using the adversarial attacks is bounded by L-norms. We follow the most commonly approaches of L_2 and L_∞ [171].

All adversarial attacks used in the dissertation are based on the gradient of differentiable functions defined over the model.

2.5.1 Fast Gradient Sign Method

Fast Gradient Sign Method (FGSM) [172] is one of the earliest algorithms for generating adversarial examples. The perturbation added to a sample is computed based on the gradients of the loss function with respect to the input. The adversary uses the sign of the gradients to create a sample that maximizes the loss. Formally, we describe the adversarial sample η as follows:

$$\eta = x + \epsilon \text{sign}\left(\nabla_x J(\theta, x, y)\right),$$

where θ denotes model's parameters, x — the input to the model ($x \in \mathbb{R}^n$), y — corresponding target, $J(\theta, x, y)$ is a loss function, and ϵ is a scaling factor describing the strength of the perturbation (it should be simultaneously large enough to cause faulty prediction and small enough that the perturbation would be imperceptible). The algorithm uses a L_∞ norm.

2.5.2 Projected Gradient Descent

Projected Gradient Descent (PGD) [173] adversarial attack expands on the concept of FGSM. This white-box attack, instead of preparing the perturbation as a one-step scheme, computes the noise using multiple iterations:

$$x^{t+1} = \Pi_{x+S}\left(x^t + \alpha \cdot \text{sign}(\nabla_x L(\theta, x, y))\right),$$

where θ denotes model's parameters, x — the input to the model, x^t — input after t perturbation steps, y — corresponding target and S is a perturbation. $L(\theta, x, y)$ is a loss function, and α is a scaling factor. PGD works for both L_2 and L_∞ norms.

2.5.3 Fast Adaptive Boundary

Fast Adaptive Boundary (FAB) [174] adversarial attack is another example of a white-box method. The algorithm is an iterative adaptive approach; the essential part is the box-constrained projections based on the linearization of the attacked classifier. It aims to produce minimally distorted adversarial examples and works for L_p norms of $p \in \{1, 2, \infty\}$. Unlike FGSM or PGD, FAB does not have a step size parameter.

2.6 Research goals

The following list contains research goals addressed in the dissertation:

1. *Assessment of the detectors' generalization capabilities based on the differences of DeepFake generators in particular training subsets*, addressed in Chapter 3 based on our paper [6]. Generalization is a common and still unsolved problem that many research groups address. It is often associated with overfitting, where models learn artifacts

of individual methods instead of the high-level features that characterize DeepFakes. This results in significantly lower performance on out-of-distribution data. Methods for assessing generalization during the training procedure may improve performance in real-life settings.

2. *Decreasing computational requirements of the methods without sacrificing the performance*, addressed in Chapter 4 based on our paper [7]. The global nature of the DeepFake phenomenon requires analysis of a large volume of audio-visual content. Therefore, solutions are needed to classify audio recordings with high performance while maintaining low processing time and hardware requirements. These models allow the democratization of anti-spoofing technology, thus increasing its accessibility to ordinary citizens.

3. *Assessment of the robustness of DF classifiers against the adversarial attacks and its influence on the generalization capabilities; enhancement of the defense methods against adversarial attacks*, addressed in Chapter 5 based on our paper [8]. Adversarial examples are among the most effective methods of attacks on anti-spoofing systems. Investigating their impact on DeepFake audio detectors and proposing methods of defense will help improve the security of these methods and start a discussion around this particular task.

4. *Introduction and evaluation of the detectors' generalization using front-ends based on large, non-SSL-based neural networks*, addressed in Chapter 6 based on our paper [9]. Recent studies show that the use of representations obtained from Self-Supervised Learning methods (trained on large-scale datasets) improves the generalization of DeepFake detection. Investigating the idea of using other architectures (e.g., dedicated to the topic of ASR) trained on large-scale datasets may improve generalization even further, increasing the number of the front-end methods used in the detection task.

Chapter 3

Evaluating generalization of detection models

Generalization of deep learning solutions concerns the performance the models achieve on the data outside of training distribution. Training data is often scarce and does not always capture all the nuances of the addressed problem. As a result, when the distributions of real-life and training data differ, deep learning models often perform weakly. While poor performance in the production environment is often an universal issue, it becomes critical in security-related tasks including DeepFake detection. Due to its young age, this field has a small, yet increasing, number of open-source datasets (see Chapter 2.4). The dynamic development of TTS and VC methods adds to the fact that these datasets do not contain the newest voice synthesis methods, which further limits the generalization capabilities of the detectors trained using these datasets.

3.1 Attack Agnostic Dataset

In this chapter, we propose a new approach of evaluating the generalization and stabilization of DeepFake detection models. We call it the *Attack Agnostic Dataset* (AAD) and use it to compare popular detection models and their front-end algorithms showing their influence on performance and generalization. Ultimately, we introduce a LCNN model trained on the concatenation of two popular front-ends: LFCC and a mel-spectrogram (see Section 6.10). Our architecture provides an enhancement in terms of generalization, stability, and performance. The following chapter is based on our manuscript [6].

The source code related to this research is available here: `https://github.com/piotrkawa/attack-agnostic-dataset`.

The anti-spoofing community has addressed the generalization issue in the past using various techniques. The subsets of ASVspoof 2019 challenge [111] utilized different attack settings to promote that the solutions generalize well. Authors of [175] proposed a generalization enhancement technique for detecting utterances spoofed by replay attacks — it was based on the specific divisions of training and testing data. The splits were conducted regarding the quality of the attacking device (replaying the utterance) and based on the distance to the microphone. In [145], the authors provided a limited analysis by preparing the training data by either including only one generation method or excluding one algorithm. This way, they evaluated the performance on the out-of-domain data.

The introduced Attack Agnostic Dataset benchmark addresses the analysis of both problems of generalization and stability. The classifier selected using this benchmark is characterized by the best possible and stable performance on the out-of-domain data — traits required from a reliable anti-spoofing system. We assess the generalization by considering multiple datasets and performing the pipelines using different folds of the data (samples are split across subsets differently, based on the method used for generating them). The idea of using splits is that the models with a small variance of the results on the different folds (i.e., trained and tested on another data split) are much more stable. Furthermore, they are less likely to fail in the case of the samples generated using new, unknown synthesis methods which means that they generalize better. Exposing the models to the varied generation methods (due to joining multiple datasets) allows us to capture the general DeepFakes properties to a much greater degree. We assess the stability of the methods by performing multiple runs with different randomness seeds and subsets of the data — models with a smaller deviation of the performance across the runs are considered more stable. The introduced framework is attack-agnostic, meaning it can be easily extended with other datasets and speech generation methods by incorporating other (more) data folds.

We based AAD on three distinct datasets — ASVspoof 2019 (LA subset) [111], which is a notable anti-spoofing dataset, as well as two new DeepFake detection datasets — WaveFake [145] and FakeAVCeleb [147]. To the best of our knowledge, this was, at the time, the most diverse and numerous audio DF detection dataset. It was comprised of 31,083 real and 222,035 fake utterances generated using 27 distinct methods.

Splitting utterances into different folds is essential to Attack Agnostic Dataset. It covers distributing the available generation methods across

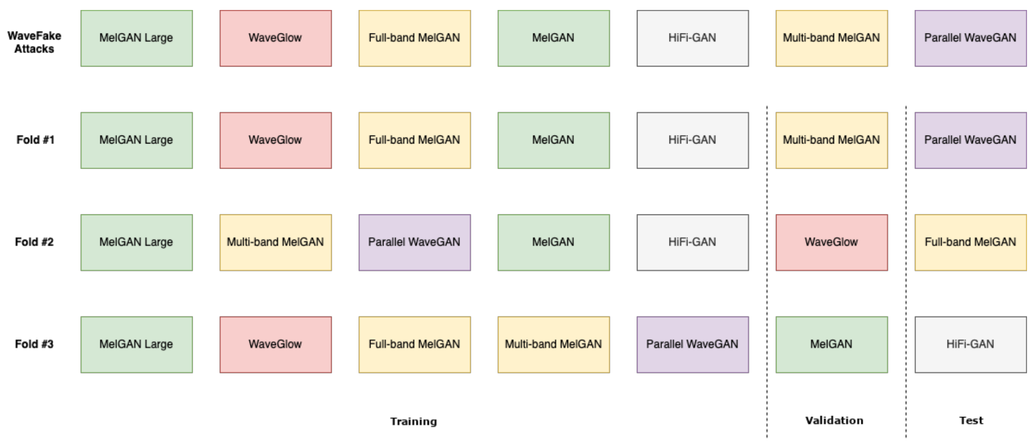


Figure 3.1: Distributing the attacks across different folds based on the example of WaveFake subset. Fold #1 depicts *information limitation* scenario — similar Full-band and Multi-band MelGAN models are split into train and test subsets. Fold #2 applies *limitation* scenario to train and test subsets. Fold #3 covers both *information limitation* (for MelGAN and MelGAN Large), as well as *information joining* for FB-GAN and MB-GAN — similar methods are in the same subset.

train, validation and test subsets. AAD can be compared to the cross-validation technique; however, the samples in AAD are distributed not in a random order but across generation methods concerning various criteria to expose models to different scenarios. By analyzing the differences across the folds' results, we can point out which models do not generalize well — such networks are less stable and, therefore, more likely to fail in the face of attacks outside of the training distribution.

In each fold, we allocated approximately 70% of the attack methods for the training subsets, i.e. five WaveFake attacks, twelve from ASVspoof, and 70% of the fake utterances for FakeAVCeleb. We distributed the remaining methods evenly between the validation and test subsets. We divided bona fide samples using the same proportion of 70:15:15 across all three subsets.

As our objective was to examine multiple scenarios, we chose the methods in each subset; some folds included several similar methods in the training subset, e.g. Multi-band MelGAN [62] in training and Full-band MelGAN [62] (its modification with another approach in computing auxiliary loss function) in the validation set (see Figure 3.1). Meanwhile, other folds were designed to restrict the presence of similar methods, e.g., only one of the five voice cloning methods from the ASVspoof dataset is included in the training subset.

We include the analysis of the training process' stability by looking at the accuracy after each epoch. The desired result is that the performance on the validation subsets grows monotonically. The variations in these values might indicate that the model does not accumulate knowledge about the artifacts used to distinguish between bona fide and fake samples. Instead, it learns different artifacts in each epoch.

3.2 Preprocessing procedure

The databases of which AAD was comprised differed not only in generation methods but also in the length of the samples, the files extensions and sampling rates. We applied the preprocessing used in the WaveFake dataset to address these differences. Their technique was based on the operations commonly used in spoofing and DeepFake works [145, 127]. Preprocessing started with the resampling to 16 kHz mono-channel, erasing the silence longer than 0.2s, and then standardizing the duration by trimming the file or padding (by repeating the sample) to about 4 seconds (64,600 frames).

3.3 Evaluated architectures

Our benchmark employed various solutions, covering most commonly used methods for detecting the validity of the utterances. The models differed in complexity, the representation of the analyzed data and even the initial purpose of these architectures. We tested LCNN [123], RawNet2 [127], Gausian Mixture Models (GMM) [176], MesoNet (MesoInception-4 variant) [11] and XceptionNet [177]. The first three models are commonly used in both tasks of spoofing and DF detection [118]. MesoNet and XceptionNet models are notable architectures used for assessing visual DeepFakes — we added them to the benchmark following FakeAVCeleb authors.

With the exception of RawNet2, which analyzes raw audio signals, all other models process 2-dimensional data (created using different *front-end* methods). For these features, we selected the most popular representations used in tasks related not only to DeepFake and spoofing detection but also speech recognition: mel-spectrogram, MFCC and LFCC (refer to Section 2.2). We aimed to provide models with features that are respectively perceivable to humans (mel-spectrogram and MFCC) or extend beyond the range of human hearing range (LFCC). The mel-spectrogram front-end was constructed by applying the Short-Time Fourier Transform (STFT) followed by the mel scale. Our calculations include the absolute value and the angle of the feature. The parameters of the used front-ends followed the ones used in FakeAVCeleb paper [147] — we used the Hann windowing function with the size of 400 frames (25 ms), window shift of 10 ms (160 frames), 512 FFT points and 80 cepstral coefficients.

3.4 Benchmark

The pipeline covered training and testing, each using a different fold out of three available. The procedure covered training for five epochs — each ended with validation on a test set. Audio DeepFake detection pipelines typically use a number of epochs of ten or greater. Our choice was motivated by the large size of the dataset compared to the related work. The checkpoint that scored the highest validation accuracy was later selected for the final testing procedure on the test set. Training and testing procedures, as well as data respresentation of Gaussian Mixture Models followed [145]. To ensure the results are reproducible, we ran each procedure thrice, each time using a different randomness seed. The seeds were as follows: {42, 1234, 4321}. Apart from using training and validation accuracy for selecting models for final tests, we reported an Equal Error Rate (EER) for the final model's performance (see Section 2.3.3).

Our training procedure used binary cross-entropy loss function, and we optimized models (using batches of 128 samples) with Adam [178] algorithm. To ensure reliable results, we used the original hyperparameters of the models whenever provided: a constant learning rate of 10^{-4} for RawNet2 [127] and LCNN [123], 10^{-3} for GMM [176] and 10^{-5} for MesoInception-4 and XceptionNet [177]. RawNet2 architecture additionally utilized regularization by decaying weights with a factor of 10^{-4}.

In Table 3.1, we report the test EERs for the evaluated models. We additionally provide a standard deviation of these results across all folds — the result of each fold is an average of repetitions on different seeds. In summary, for each model, the table contains the results of nine runs (three runs per each of the three folds).

There exists a significant difference in the results across different folds — all the models, apart from XceptionNet, provided the best EER on the third fold. This lets us conclude that opting out of some of the attacks significantly influences the performance of the models, which means one of two things: that some combination of the attacks in training and validation sets allowed the model to learn more general features and, therefore, detect DeepFakes to a better degree; or that some of the attacks are much more

Table 3.1: The results of evaluating the main architectures on the test subset. The front-end models are based on the LFCC front-end. The reported values are the mean and standard deviation of the EER for each fold. We average them across the result for three randomness seeds.

Fold	**LCNN**		**XceptionNet**	
	EER	STD	EER	STD
1	0.0953	0.0073	0.1621	0.0148
2	0.0952	0.0072	0.1277	0.0117
3	0.0237	0.0049	0.1306	0.0322
Fold	**MesoInception–4**		**RawNet2**	
	EER	STD	EER	STD
1	0.3858	0.0666	0.1944	0.0136
2	0.3638	0.0594	0.2379	0.0205
3	0.1765	0.0562	0.1414	0.0089
Fold	**GMM**			
	EER	STD		
1	0.2514	0.0072		
2	0.3438	0.0144		
3	0.2518	0.0067		

difficult to detect and therefore their presence in the test set negatively influenced the final score.

LCNN was the method that provided the best results across all folds and, therefore, showed the best generalization. It achieved EERs of respectively 0.0953, 0.0952 and 0.0237. In addition, its performance was the most stable, i.e., it was characterized by the lowest standard deviation — on average 0.0065. The second best model was XceptionNet which scored EERs of 0.1621, 0.1277, 0.1306. XceptionNet is a robust classifier capable of capturing features that may suggest fake speech; its results suggest that models previously used in detecting spoofed faces may be used to detect DeepFake utterances. Nevertheless, the results were, on average, almost 2 times worse than the results of LCNN. RawNet2 provided the third best results — they deviated from the ones typically reported in related work and were equal to EER of 0.1944, 0.2379 and 0.1414. The model, however, was characterized by a much smaller deviation than XceptionNet — on average 0.0143 vs. 0.0196 of the results, as the stability of the results is a desired feature of the detection system.

3.5 Front-ends comparison

Additionally, we provide an extensive comparison of the commonly used representations of the audio signal (front–ends) and their influence on both the generalization and stability of the models. We performed the study using LCNN, as it provided the best and most stable results in the first benchmark — the discussed benchmark was conducted similarly to the previous one.

Table 3.2: Test results of LCNN models based on the different front–ends (LFCC, MFCC, mel-spectrogram) and their concatenation. The results were obtained analogously to Table 3.1.

Fold	**LFCC**		**MFCC**		**Spec**	
	EER	STD	EER	STD	EER	STD
1	0.0953	0.0073	0.1530	0.0014	0.3870	0.0161
2	0.0952	0.0072	0.0873	0.0042	0.3044	0.0138
3	0.0237	0.0049	0.0507	0.0062	0.3009	0.0049
Fold	**LFCC+Spec**		**MFCC+LFCC**		**MFCC+Spec**	
	EER	STD	EER	STD	EER	STD
1	0.0914	0.0032	0.1344	0.0128	0.1502	0.0048
2	0.0916	0.0064	0.0826	0.0052	0.0755	0.0060
3	0.0313	0.0034	0.0318	0.0072	0.0472	0.0024

The results presented in Table 3.2 are much less conclusive than the ones shown in the original benchmark. Due to the differences across folds, one can not select a universally superior front-end. When treating LFCC as a baseline, we observe that all front-ends, apart from the standard mel-spectrogram, provide improvements in at least one fold. Front-ends of LFCC and LFCC+Spectrogram yielded similar results, with the latter showcasing improved Equal Error Rate (EER) in two out of three folds. Moreover, LFCC+Spectrogram provided enhanced stability by the standard deviations of the results equal to 0.0032, 0.0064, and 0.0034.

The least stable front-end was MFCC — the choice of the methods in training and testing subsets highly influenced its results. The situation was similar in the case of the concatenation of MFCC and LFCC, which, despite providing better results in relation to MFCC, was still worse than the LFCC front-end. Using only the mel-spectrogram front-end resulted in the worst results across all front-ends. However, concatenating it with other features enhanced all but one fold (fold three of LFCC+Spectrogram).

The provided results show that there often exist significant differences in the results depending on the methods used as the training and testing data. In most cases (except for XceptionNet) third fold provided the best EER scores. The ASVspoof subset comprised A02, A09 and A10 attacks, all neural network-based systems. A02 and A09 attacks based on statistical parametric speech synthesis (SPSS) framework [179] vocoded by respectively WORLD [180] (A02) and Voicaine [181] (A09) vocoders. A10 system was based on Tacotron2 synthesizer [53] vocoded by WaveRNN network [59]. The WaveFake subset was composed of samples vocoded by HiFi-GAN network [57], whereas FakeAVCeleb data consisted of 15% of the samples.

We conclude the presence of SPSS framework based models might be the one of the reasons for the best results on the third fold making these attacks easiest to detect. As SPSS is a method introduced in 2013 and the quality of these audio samples is worse in relation to other solutions, we hypothesize that these utterances contained the artifacts that are easier to spot.

3.6 Ablation study

In order to further investigate the influence of the front-ends in the task of DeepFake detection, we conducted an ablation study. In this study, we looked closely at the results obtained by mel- and linear-scaled frequency cepstral coefficient front-ends with respect to the particular subsets of the Attack Agnostic Dataset. Table 3.3 contains the results of the LCNN model trained using the LFCC and MFCC model for each subset of AAD. We report the results for each fold separately.

Table 3.3: Ablation study of the impact of front-end algorithms on the performance of individual subsets. The results are based on the test subset (and one seed). All the results were obtained by LCNN model with LFCC or MFCC frontend.

Dataset	**ASVspoof**		**WaveFake**		**FakeAVCeleb**	
Fold no.	LFCC	MFCC	LFCC	MFCC	LFCC	MFCC
1	0.1877	0.1364	0.0211	0.2440	0.0595	0.0747
2	0.1654	0.1643	0.0030	0.0040	0.0293	0.0685
3	0.0325	0.0673	0.0038	0.0088	0.0606	0.0732

Based on the results presented in Table 3.3, it is evident that the input of the LFCC front-end contributed to better results for DeepFake subsets (WaveFake and FakeAVCeleb). Most DeepFake generators prioritize creating realistic utterances that can deceive human perception, often neglecting other artifacts imperceptible to humans. For this reason, standard DeepFake methods can be relatively easily detected using appropriate signal features, such as LFCC.

The results of the ASVspoof subset are similar in the case of both front-ends. As the mel-scale is used to reduce high-range frequencies and bring out the features of human hearing range, we conclude that the fake samples of the ASVspoof 2019 LA subset did not have that many high-range frequency artifacts.

The lowest EER characterizes the results on the WaveFake subset. We conclude that the reason for that is that they are generated using similar methods: out of the eight available solutions, seven are based on GAN networks. Moreover, some of them are interdependent (e.g. MelGAN Large being a variant of MelGAN), resulting in only four unique generation architectures.

The results of FakeAVCeleb subset, apart from an evident outlier in Fold 1 of the MFCC model, are worse than WaveFake's. These results are also much more stable across all folds. FakeAVCeleb is composed of only one generation method — SV2TTS [150]; the results indicate that the choice of the methods in training and validation subsets did not significantly impact the detection of this method.

Based on Figure 3.2, it is evident that the training stability of DeepFake detection was generally poor. While the accuracy consistently improved for the training subsets, there were considerable fluctuations in accuracy for the test subsets. This suggests that the training of these models was generally unstable — the reason for that is the utilization of various methods and attacks. However, the stability greatly improved when using LFCC front-ends.

3.7 Conclusion

The research described in this chapter covers the problem of generalization and stabilization of the models used in audio DeepFake detection addressing the Research Goal 1 (refer to Section 2.6). We proposed a framework for assessing the aforementioned traits — *Attack Agnostic Dataset*. It consisted of three DeepFake and spoofing datasets — ASVspoof 2019 LA, WaveFake

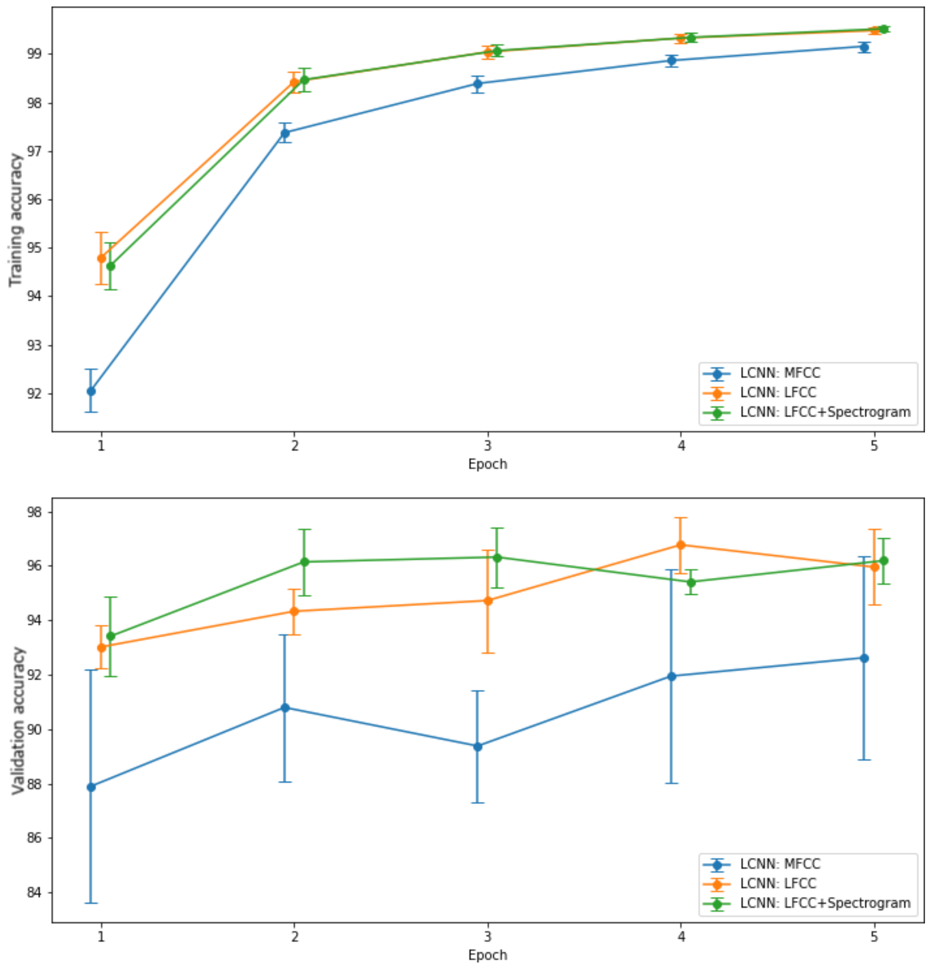


Figure 3.2: Training and validation accuracy (along with standard deviation) after each of the training epochs. The presented results cover LCNN model with three selected front-ends. Source: [6].

and FakeAVCeleb. Using the training based on the disjoint division of the attacks, allows for the assessment of the generalization and stabilization capabilities and later selection of the best performing model. Due to its framework-like nature, the solution is easily extendable and can be adapted to other new databases. We used AAD to thoroughly benchmark the most popular DeepFake detection architectures from both audio and visual domains. The concatenation of the datasets we used, at the time, was one of the largest audio DeepFake detection databases. Our contribution additionally covered research on the influence of different front-ends. By introducing a novel front-end composed of LFCC and mel-spectrogram and using it with the LCNN model, we achieved an average EER of 0.0714, which is an improvement of 5% in relation to the sole LFCC front-end.

Chapter 4

Fast and reliable detection using SpecRNet

The following chapter focuses on fast and reliable methods for detecting audio DeepFakes. We describe the method with severely limited parameters that achieves results comparable to those proposed in Chapter 3. The main motivation is that multimedia platforms and VOD services are increasingly popular, e.g. 720,000 hours of content is uploaded to YouTube on a daily basis [182]. These websites, as well as other social media platforms, often struggle with the problem of fake news [183] — misleading content in text, audio or video domains. The threat of misinformation constantly grows as the generation methods in all these domains are constantly enhanced and are becoming more popular [184, 185]. New state-of-the-art speech generation methods not only improve on the quality of speech, but also decrease the quantity of the material required for its creation. Nowadays, several minutes of someone's speech is enough to generate convincing fake utterances [40].

As a countermeasure against such threats, regulations like the European Union's Strengthened Code of Practice on Disinformation [186] are being introduced. It obliges content providers (such as online multimedia platforms) to implement self-regulations to counter disinformation, including DeepFakes. The scientific community must remedy this problem by creating fast and reliable DeepFake detection methods. Such solutions can later be used at a large scale to constantly process the uploaded materials and filter out misleading content. Such lightweight algorithms can also be used to equip average citizens with means to defend themselves against misinformation by checking the content on their own. As neural networks typically require high-end Graphic Processing Units (GPUs), their high costs financially exclude many from independently verifying the materials. Thanks to methods with low computational complexity, such systems can be run on local devices like laptops, personal computers or even smartphones.

In this chapter, we introduce a novel neural network architecture called SpecRNet. We evaluate it by comparing it to reliable, contemporary DF detection architectures: RawNet2 [126] and LCNN [123], which were the most popular at the time of the research. We train the models using the WaveFake dataset. Apart from the efficacy, we compare the processing speeds of the models with respect to the used device (CPU or GPU) and the batch size. We additionally propose and conduct three new benchmarks. Their main aim is to evaluate models in different scenarios: the scarcity of the training samples and in the case of short utterances. The last evaluation prodives more in-depth information on the difficulty of the attacks available in the dataset used.

4.1 SpecRNet

We propose SpecRNet — a neural network for fast and reliable DeepFake detection. Its architecture was inspired by RawNet2, a model known for its good performance in speech processing tasks like speaker recognition or spoofing and DeepFake detection. RawNet2 operates on the raw audio signal, i.e., processes one-dimensional data, whereas SpecRNet operates on two-dimensional speech signal representations. This enables use of various front-ends like spectrograms, linear-frequency or mel-frequency cepstral coefficients (LFCC or MFCC described in Section 2.2). In this work, we preprocessed audio using LFCC. The reason behind such a choice is its superior performance observed in Chapter 3.

The architecture of SpecRNet is presented in Table 4.1. The exact (hyper-)parameters were established in the course of the conducted experiments. The network's input first undergoes the initial normalization with two-dimensional batch normalization [187] followed by SeLU activation function introducted in [188]. The information is later processed by three residual blocks — their task is to perform feature extraction, emphasizing the properties important for the given task. Each block comprises two 2-dimensional convolution layers preceded by the batch-normalization and LeakyReLU activation function. Note that the initial normalization and activation layers are skipped only in the first residual block, as the preceding normalization and activation layers are already processing the in-

The source code related to this research is available here: `https://github.com/piotrkawa/specrnet`.

put. The input signal to the residual block (called residual identity path) is then added to the result of the blocks mentioned above to enhance the processed information. The identity path's task is to expose the model to the original information and facilitate the flow of gradients during backpropagation (originally introduced in ResNet networks [189]). To ensure the synchronization of channel numbers between the identity and main paths, we process the original information using an extra convolutional layer with a kernel size of one. Akin to RawNet2, SpecRNet contains two types of residual blocks differing in the number of convolution channels. Due to the constant number of 64 channels in the third residual block, there is no synchronization between the identity and main paths. Consequently, the additional convolution layer is not employed in this scenario. The output of each residual block is processed by the 2D max-pooling layer, followed by Feature Map Scaling (FMS) attention [126] and another 2D max-pooling. Figure 4.2 depicts the residual block and FMS attention structures.

The signal processed by the residual blocks is later normalized using batch normalization followed by the SeLU activation function and passed to two bidirectional GRU layers to provide temporal analysis of the given embedding. The architecture ends with two fully connected layers. The second outputs a single value from the range [0, 1], which after thresholding, denotes the final prediction.

4.2 Pipeline description

The pipeline in the conducted experiments was based on the WaveFake dataset. At this time, this was one of the first DeepFake-oriented detection datasets. More information on the dataset can be found in Section 2.4. The utterances used the preprocessing procedure introduced by the dataset's authors. It covered first resampling the samples to 16 kHz one-channel audio and then deleting the silence intervals longer than 0.2 s. All samples underwent duration normalization to about 4 s (64,600 frames). It was done by either trimming too long or repeating too short utterances.

Unless specified otherwise, all benchmarks included three architectures: LCNN [123], RawNet2 [127] and SpecRNet. The reason for this choice of solutions was as follows: RawNet2 was the basis for developing the SpecRNet architecture and represented models based on the raw audio. LCNN is a widely used architecture in the literature which is based on front-end features. In addition, both models score highly on many benchmarks.

Table 4.1: The architecture of SpecRNet model. The convolution layers use the following convention: Conv2D(kernel size, input channels, output channels). Convolutions listed below the dotted lines refer to operations performed on identity paths before summation. The table design was inspired by the original RawNet2 work [126]. The table adapted from the source manuscript [7].

Layer	**Input**	**Output shape**
LFCC	64,600	$1 \times 80 \times N$
preliminary normalization	BN2D SELU	$1 \times 80 \times N$

Residual Block	Conv2D(3, 1, 20)
	BN2D
	LeakyReLU(0.3)
	Conv2D(3, 20, 20)
	. . . .
	Conv2D(1, 1, 20)
		$20 \times 80 \times N$

FMS Attention	Maxpool(2) FMS Maxpool(2)	$20 \times 20 \times \frac{N}{4}$

Residual Block	BN2D
	LeakyReLU(0.3)
	Conv2D(3, 20, 64)
	BN2D
	LeakyReLU(0.3)
	Conv2D(3, 64, 64)
	. . . .
	Conv2D(1, 20, 64)
		$64 \times 20 \times \frac{N}{4}$

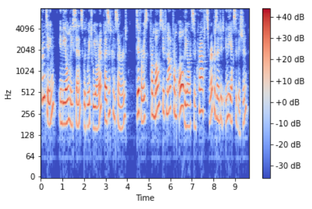
FMS Attention	Maxpool(2) FMS Maxpool(2)	$64 \times 5 \times \frac{N}{16}$

Residual Block	BN2D
	LeakyReLU(0.3)
	Conv2D(3, 64, 64)
	BN2D
	LeakyReLU(0.3)
	Conv2D(3, 64, 64)
	. . . .
	-
		$64 \times 5 \times \frac{N}{16}$

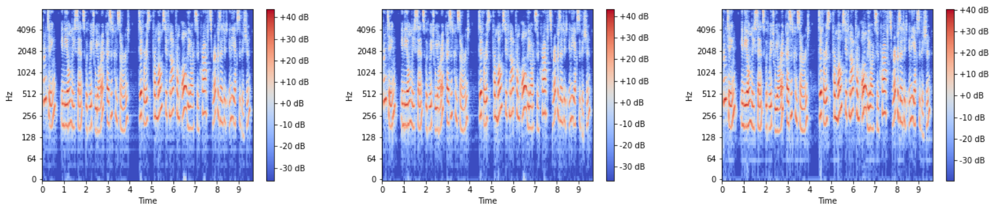
FMS Attention	Maxpool(2) FMS Maxpool(2)	$64 \times 1 \times \frac{N}{64}$
pre-recurrent normalization	BN2D SELU	$64 \times 1 \times \frac{N}{64}$
GRU	GRU(64, bi)	128
GRU	GRU(128, bi)	128
FC	128	128
FC	128	1

4.3 Baseline comparison on a full dataset

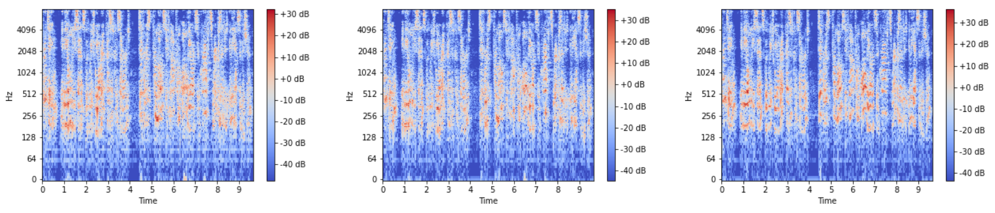
The baseline training procedure, denoted in this chapter as *full dataset training*, was the basis for all other benchmarks. The pipeline covered training the models for ten epochs on the full dataset. Each epoch ended with the validation using the entire validation subset. The checkpoints with the



(a) Bona fide spectrogram.



(b) Spectrograms of MelGAN-Large, MelGAN and Parallel WaveGAN.



(c) Differences between bona fide spectrogram and the vocoders' outputs.

Figure 4.1: Spectrograms of the bona fide sample (LJSpeech dataset) along with corresponding WaveFake manipulations and the absolute differences between spoofed and bona fide spectrograms. The figure adapted from the source manuscript [7].

highest validation scores were later evaluated using the test set.

We performed the optimization using Adam optimizer [178] on the batches of 128 samples. LCNN and RawNet2 used originally reported learning rates of 10^{-4} with no scheduling procedure. The same value was used for the SpecRNet training. SpecRNet and LCNN underwent additional regularization using the weight decay of 10^{-4}. Both front-end-based models were processing LFCC created using the parameters of 10 ms window shift with 5 ms Hann windowing, 512 FFT points and 80 coefficients.

The samples of the WaveFake dataset were split into three disjoint subsets of training, validation and testing using the ratio of 70:15:15. The performances were reported using EER and AUC metrics along with their standard deviations (refer to Section 2.3.3). We ensured that the reported results were reproducible and generalizable — each processes was repeated

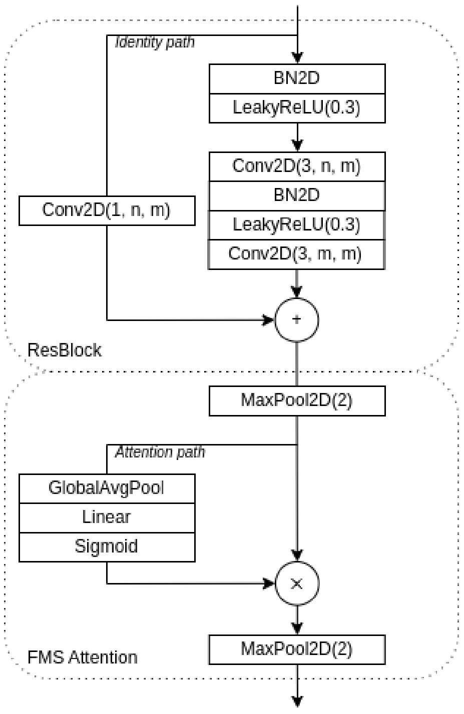


Figure 4.2: Schemes of the residual block (denoted as ResBlock) along with the FMS Attention block. The figure adapted from the source manuscript [7].

three times, each time using a different randomness seed {42, 1234, 4321}.

Table 4.2: The test results of the baseline benchmark using the full dataset. We report the average Equal Error Rate (EER) and Area Under Curve (AUC), along with their standard deviations (Std) as a mean of runs with three different randomness seeds.

Model	**EER (±Std)**	**AUC (±Std)**
LCNN	0.001399 (± 0.000325)	99.9952 (± 0.0031)
RawNet2	0.045973 (± 0.002742)	99.1254 (± 0.0670)
SpecRNet	0.001549 (± 0.000283)	99.9941 (± 0.0015)

Table 4.2 presents the baseline benchmark results. LCNN demonstrated the best performance in both metrics — 0.001399 (EER) and 99.9952 (AUC). SpecRNet closely followed by achieving EER of 0.001549 and AUC of 99.9941. Please note that the results obtained by SpecRNet exhibited

greater consistency — as evidenced by a smaller standard deviation. This feature is crucial in security systems, where we expect the consistent performance of the classifiers. RawNet2 exhibited the weakest performance, with an EER of 0.045973 and an AUC of 99.1254. The difference between RawNet2 and other models indicates that models based on 2-dimensional data (in particular LFCC) can perform significantly better than the raw-signal-processing architectures.

4.4 Time benchmark

The number of parameters is one of the most important factors when it comes to the speed of the neural networks. As our main task was to provide a lightweight architecture for the settings of big-data processing and in-house computations, the number of parameters should be significantly lower in relation to other architectures. In this section, we present the number of trainable parameters for each of the evaluated models and provide an in-depth analysis of inference times concerning the used device and the batch size.

Table 4.3: Number of the trainable parameters in evaluated architectures. SpecRNet is almost twofold smaller than LCNN.

Model name	**Trainable parameters**
SpecRNet	277,963
LCNN	467,425
RawNet2	17,620,385

Table 4.3 contains a number of trainable parameters of each model. SpecRNet contains significantly fewer parameters — up to two times fewer than LCNN and 60 times fewer than RawNet2. This also contributes to smaller VRAM usage: processing a batch of 32 samples lasting 4 s requires 1025.34 MB and 1135.36 MB for respectively LCNN and RawNet2. Meanwhile, SpecRNet requires only 826.72 MB. This means that it is much more accessible due to fitting on a much more significant number of GPUs.

We additionally compare the inference speed conducted on randomly drawn samples and present an average of 1,000 trials. To cover a wide range of scenarios (i.e., both big-data and personal PC settings), we ran the experiments using GPU (NVIDIA Tesla P40) and CPU (2 GHz Quad-Core

Intel Core i5). DeepFake samples (unlike spoofing) can often be long — for this reason, we provide timings for various batch sizes. By breaking down the utterances into smaller audio chunks, it is possible to analyze the entire material in a single forward pass using batched input.

Table 4.4: Inference times (milliseconds) in relation to the batch size (BS) and the used device (CPU and GPU). Larger batch sizes result in more significant differences between SpecRNet and other models.

Model Name		**CPU**	
	BS 1	***BS 16***	***BS 32***
LCNN	41.632	451.745	944.672
RawNet2	151.913	1386.375	2637.200
SpecRNet	27.358	370.843	706.300
Model Name		**GPU**	
	BS 1	***BS 16***	***BS 32***
LCNN	3.713	11.764	20.088
RawNet2	11.934	43.317	56.787
SpecRNet	3.669	7.1492	13.244

Table 4.4 presents the results, for CPU and GPU, showing that a larger batch size naturally results in a much longer inference time. The inference time per sample is, however, decreasing. With the increasing batch size, the results on GPU show that differences between models become more significant — while on a single element batch, they are negligible, they become more prominent when the batch size is increased to 16 or 32. The results also confirm the intuition related to the number of the parameters — RawNet2, which contains the largest number of parameters, has the most extended inference times, taking up to six times longer than SpecRNet.

The times in Table 4.4 concern only the forward pass through the networks. This duration covers the whole procedure only in the case of RawNet2, as the model uses raw audio and, therefore, does not require any additional computations. One should consider the preparation of front-end features to fully measure the time required for the inference for LCNN and SpecRNet. The additional computations take a similar time on a GPU for both networks and are equal to 0.9779 ms, 7.2666 ms, and 13.6549 ms for batch sizes 1, 16, and 32, respectively. We observe similar tendencies in the results on the CPU.

4.5 Limited attacks benchmark

In this section, we present the *limited attacks* benchmark, which explores the relations between particular generation methods (attacks) and the performance of detecting them. There exists multiple methods of generating audio DeepFakes, which differ in the quality of their outputs. This quality is influenced, among others, by the types and quantity of the artifacts in the generated waveforms. Intuitively, the greater number of the artifacts, the easier it is to detect such fake utterances.

The idea for the benchmark was the following: we performed a full training and testing pipeline N times, where N is the number of the generation methods in the training dataset (here: $N = 8$). Each time, we omited, in all steps, one of the DF generation methods. The results obtained through such a scenario can be interpreted in the following manner: if the scored EER was higher than the baseline (see Section 4.3), the omitted attack contributed positively to the baseline results, i.e. it was detected correctly, and due to its absence, the results decreased. One can think of such an attack as *easy to detect*. Enhancement of the results (i.e. decrease in EER) suggests that a particular attack was not predicted correctly in the baseline, its presence had a negative influence on the results and omitting it resulted in the enhancement of the results — it was a *difficult attack*. *Limited attacks* setting can be compared to out-of-distribution approach used in WaveFake paper [145]. However, the primary objective of their scenario was to test the models' generalization abilities, whereas our benchmark aims to evaluate the difficulty level posed by specific attacks. Despite similarities in the training procedures, i.e. omitting one of the attacks, their evaluation parts differ — WaveFake runs the evaluation on all the attacks, whereas we did it only on $N - 1$.

For this pipeline we chose LCNN and SpecRNet due to their better performance in relation to RawNet2. All the training details, apart from not using all attacks simultaneously, were the same as in the baseline approach (refer to Section 4.3).

The results in Tables 4.5 and 4.6 show that there exist differences in the performance in relation to the used attacks. While some attacks improved the performance of the classifiers, others reduced it. LCNN's results show that MelGAN was the most challenging attack for this classifier, with its absence improving the EER from 0.00140 (full dataset) to 0.00087. The only

Table 4.5: EER results and standard deviations of *limited attacks* benchmark. Particular columns correspond to the test results when omitting a generation method in the whole training and testing procedure. We include the results on the full dataset (denoted as *Full DS*) for reference. The best performance for each of the models is displayed in bold.

Model	**Full DS**	**HifiGAN**	**FB–MelGAN**
LCNN	0.00140 (± 0.00033)	0.00097 (± 0.00008)	0.00101 (± 0.00026)
SpecRNet	0.00155 (± 0.00028)	0.00137 (± 0.00022)	0.00144 (± 0.00066)
Model	**MelGAN–Large**	**PWG**	**MB–MelGAN**
LCNN	0.00148 (± 0.00016)	0.00122 (± 0.00023)	0.00123 (± 0.00023)
SpecRNet	0.00157 (± 0.00083)	0.00181 (± 0.00038)	0.00146 (± 0.00062)
Model	**WaveGlow**	**MelGAN**	**TTS**
LCNN	0.00119 (± 0.00031)	**0.00087** (± 0.00012)	0.00094 (± 0.00030)
SpecRNet	0.00168 (± 0.00075)	**0.00136** (± 0.00030)	0.00140 (± 0.00083)

attack contributing to the result's decline was MelGAN-Large (increase in EER to 0.00148). Similarly to LCNN, we observed the highest EER in the results of SpecRNet for MelGAN — the decrease in EER from 0.00155 to 0.00136. The *easiest* to detect was Parallel WaveGAN [61], which scored the EER of 0.00180 — we conclude that the utterances generated using this method were detected best in the baseline.

SpecRNet demonstrated stable and comparable performance to LCNN. Notably, both classifiers could differentiate between all types of attacks rather than learning to identify only one and ignoring others. Additionally, the lower standard deviation of the EER achieved by SpecRNet, compared to LCNN, suggests that our architecture has superior generalization properties and is likely to perform reliably when faced with novel attack methods.

4.6 Short utterances scenario

Synthesis quality and the sample degradation e.g. through noises or background audio, are one of the most important factors that influence the difficulty of the detection task. The duration of the generated sample is yet another significant factor. When generating extended audio segments comprising entire sentences, many TTS and VC models are prone to exhibit greater imperfections such as inconsistencies, artificiality in pronunciation and prosody, or abrupt changes in tone or frequency.

Table 4.6: AUC results with standard deviation of *limited attacks* benchmark. The best performance for each of the models is displayed in bold.

Model	Full DS	HifiGAN	FB–MelGAN
LCNN	99.9952 (± 0.0031)	99.9994 (± 0.0003)	99.9989 (± 0.0008)
SpecRNet	99.9941 (± 0.0015)	99.9949 (± 0.0039)	99.9960 (± 0.0016)
Model	**MelGAN–Large**	**PWG**	**MB–MelGAN**
LCNN	99.9976 (± 0.0022)	**99.9991** (± 0.0003)	99.9972 (± 0.0023)
SpecRNet	99.9961 (± 0.0044)	99.9960 (± 0.0018)	99.9965 (± 0.0015)
Model	**WaveGlow**	**MelGAN**	**TTS**
LCNN	99.9970 (± 0.0034)	99.9986 (± 0.0008)	99.9973 (± 0.0033)
SpecRNet	99.9955 (± 0.0020)	**99.9987** (± 0.0006)	99.9971 (± 0.0033)

The objective of this benchmark was to assess the effectiveness of the models on short sequences. This setting emulates a scenario when an attacker replaces or adds specific keywords to a sentence using generated content instead of spoofing an entire utterance. Swapping the entity, action, or negating a sentence, can result for instance in politicians denouncing their ally rather than criticizing their opponent. This way, the utterance can preserve, to a great degree, the naturalness of the original while altering the intended meaning.

The discussed benchmark differed from the baseline only in the length of the processed samples. Instead of using the utterances of around 4 s, a standard for the task of DF detection, all utterances were truncated to 1 s. Such a period is long enough to contain a keyword that will be swapped in altered utterance while short enough to show as little inconsistencies and imperfections as possible.

Table 4.7: The results of EER and AUC scored in *Short utterances* benchmark, along with their standard deviations. Instead of using standard duration of 4 s, models were trained and tested using the samples of 1 s.

Model name	EER (Std)	AUC (Std)
LCNN	0.00996 (± 0.00374)	99.9486 (± 0.0324)
RawNet2	0.14450 (± 0.00842)	93.4013 (± 0.7183)
SpecRNet	0.01178 (± 0.00095)	99.9322 (± 0.0324)

Table 4.7 contains the benchmark results using four times shorter samples. Even though we performed trimming of the silences, assuring that all the samples contained speech, the performances degraded significantly.

LCNN provided the results most consistent in regards to the baseline benchmark and scored EER and AUC of respectively 0.00996 and 99.9486. SpecRNet was characterized by lower robustness against less data by providing an EER of 0.01178 and AUC of 99.9322. RawNet2 achieved the least stable performance — EER of 0.14450 and AUC of 93.4013.

The over fourfold reduction of the data volume was the primary factor influencing the decline in the performance. Such reduction in the available information resulted in fewer potential features (imperfections) used to distinguish bona fide and fake utterances. Despite the handicap of a much more difficult task resulting in a performance's decline, smaller, front-end-based architectures demonstrated robustness and yielded satisfactory results — scoring the EER of around 0.01. These architectures can still successfully spot targeted substitution of keywords. The larger architecture of RawNet2 could not capture the nuances of the data — we conclude that the reason for that was a much smaller volume of the dataset: bigger architectures requires a much greater quantity of samples to converge.

4.7 Data scarcity scenario

The next benchmark we introduce addressed the problem of the continous introduction of new DeepFake manipulation methods. Many DeepFake generation techniques are open-sourced or available as a web service. By using a considerably smaller volume of the data in relation to the baseline benchmark, we mimicked a situation when some new audio DeepFakes emerged. However, there is no information on the creation procedure or the algorithm.

To prepare against such a new attack, a detection classifier should be trained on the samples created using this new method. However, due to a lack of an access to this synthesis model, one can perform the training using only the available samples — e.g. gathered from the Internet. This however results in a much smaller quantity than typically used for the training. The presented problem can be considered an assessment of the generalization capabilities of the detection architectures. Typically, the term *generalization* concerns the robustness against unseen methods [6, 145], but in this particular sense, one can consider it as the evaluation of how much of the data a model needs to learn the essential features required to distinguish fake and bona fide samples.

The benchmark, once again, was derived from the procedure introduced in Chapter 3. To mimic the scarcity of the data, we reduced training and validation subsets to 10% of their original size. Moreover, we reduced the number of epochs from ten to four to avoid overfitting this smaller dataset. We performed a testing procedure on the original number of samples.

Table 4.8: The results of *data scarcity* benchmark — using 10% of the training data resulted in the performance decline.

Model	**EER (Std)**	**AUC (Std)**
LCNN	0.00631 (± 0.00126)	99.9599 (± 0.0148)
RawNet2	0.24082 (± 0.01912)	84.3158 (± 2.0555)
SpecRNet	0.00800 (± 0.00192)	99.9390 (± 0.0290)

Table 4.8 shows that the smaller volume of the training data negatively impacted the classifier's performance. LCNN and SpecRNet showed the performance of an EER of 0.00631 and 0.00800 and an AUC of 99.9599 and 99.9390, respectively. Despite using ten times less training data resulting in the decline in relation to the baseline benchmark (0.00140→0.00631 for LCNN and 0.00155→0.00800 for SpecRNet), we still consider them as acceptable. On the other hand, RawNet2 showed a much greater decline by scoring the EER of 0.240821 and AUC of 84.3158. Such results of a DeepFake detection model indicate that a classifier is ineffectual — while such AUC score can be considered promising, there exists a significant difference between FAR and FRR results leading to the high EER value. We conclude that the reason for such high EER score is the data shortage and the architecture size — RawNet2 contains roughly 17 million parameters and therefore requires more data and more steps to converge than LCNN and SpecRNet, which both contain less than 0.5 million parameters.

The performance reported in this benchmark suggests that smaller architectures of LCNN and SpecRNet are much more robust against the data shortage. They showed good generalization capabilities, as only 10% of the original data was sufficient to capture the meaningful features. We conclude that newly introduced DF detection datasets should, instead of focusing on the number of utterances in the dataset, emphasize the variety of generation techniques, as this factor mainly influences the performance of the detectors.

The results obtained in both *data scarcity* and *short utterances* benchmarks prove the robustness of SpecRNet architecture. Despite being exposed to a much smaller quantity of data by a smaller number of samples and shorter utterances, the architecture maintained satisfying results, which can lead to the conclusion that the model can successfully be used in real-life DeepFake detection systems.

4.8 Conclusions

In this chapter, we addressed Research Goal 2 (refer to Section 2.6) by introducing a fast and efficient solution for detecting audio DeepFakes — a novel neural network architecture called SpecRNet. It achieves a 40% boost in inference speed compared to the LCNN model, which is considered one of the fastest DeepFake detection methods. Despite the increased speed, SpecRNet maintained comparable detection results, showcasing its efficiency without compromising much of the performance.

SpecRNet, thanks to its speed and efficiency, has broad applications that cover using it in a big-data environment like online multimedia platforms or as a classifier running on local consumer-grade devices. We supported this claim by evaluating our architecture with four benchmarks: *baseline*, *limited attacks*, *short utterances* and *data scarcity*. The last three benchmarks are novel — introduced in this chapter. We compared our model with two state-of-the-art DeepFake detection architectures of RawNet2 and LCNN using one of the largest, available at the time, DeepFake datasets: WaveFake [145]. In the basic benchmark, the proposed architecture exhibited only a marginal 0.001% decrease in AUC performance compared to the LCNN model. Furthermore, when assessing the reduced datasets of trimmed samples and reduced sample numbers, the performance difference did not exceed 0.02%.

Chapter 5

Adversarial attacks and defense against them

Current audio DeepFake detectors provide satisfying performance on many benchmark datasets [111, 118, 145]. While these results may indicate that with the current detection methods, DeepFakes do not pose much of a threat, this may not reflect real-life scenarios. As highlighted in Chapters 2, 3, 4, one of the main issues of the detectors is their generalization — the ability to perform well on the utterances generated using methods outside of training distribution.

The generated samples can be further modified using various digital signal processing algorithms to further deviate their characteristics from the features observed in the training sets. Such manipulations may include pitch or speed alteration, encoding (telephony, lossy, etc.) or adding various background noises (see Section 2.3.2). Most currently used DeepFake detection methods are based on deep neural networks and, therefore, are vulnerable to *adversarial attacks* (AA). These small, imperceptible changes in the audio tracks result in incorrect predictions of the neural networks, effectively decreasing their performance (see Section 2.5).

In this chapter, we cover the topic of adversarial attacks on audio DeepFake detectors. Previously, these attacks were addressed in the fields of audio spoofing [190, 191], speaker and speech recognition [192, 193] and adversarial training in ASR environment [194, 195]. To our knowledge, we were the first to address adversarial attacks on DeepFake detection. Our contribution covers the analysis of three classifiers (LCNN [124], SpecRNet [7] and RawNet3 [196]), which we successfully compromised with the use of three commonly used methods for generating adversarial examples: FGSM, PGDL2 and FAB (cf. Section 2.5). We consider two attack scenarios: *white-box* and using a *transferability* mechanism. As a result, we demonstrate that even with limited knowledge of the attacked system, the adversary can successfully target the model, leading to the decreased performance of the predictions and, therefore, compromising the system. Then, we introduce a novel method of adaptive adversarial training to increase the robustness against such attacks. We analyze this training algorithm in-depth in an ablation study.

5.1 Included scenarios

In our experiments, we addressed two scenarios of adversarial attacks: *white-box* and *transferability* [169]. These methods differ in the amount of knowledge about the system which is known to the adversary. The white-box scenario assumes that an adversary has access to the targeted model — the adversarial samples are prepared using the very same network that will be later targeted.

On the other hand, the transferability scenario involves adversarial attacks using models that differ from the target model, i.e. it does not require access to the targeted model or its weights. The scenario is highly similar to the black-box attack; however, these two have fundamental differences. Audio DeepFakes is a relatively new domain with limited available datasets and detection architectures. Moreover, due to their quality and popularity, architectures like LCNN [124] or datasets like ASVspoof [111] are highly probable to be used in deployed systems. Additionally, the preprocessing procedures applied to the analyzed samples are standardized — they involve resampling to 16 kHz and standardizing the duration. As a result, an adversary, despite no exact knowledge about the system, could effectively launch an attack using similar methods. However, since the exact parameters are not disclosed, the adversary is only aware of the preprocessing operations and datasets; one can compare this approach to a black-box setting. Typically, related work refers to this exact setting as a black box. However, we decided to comply with computer security nomenclature, naming it *transferability* scenario.

5.2 Adversarial attacks

Due to the novelty of our research on AA in the domain of audio DeepFakes, we selected the most popular attacks, which were extensively studied in

The source code related to this research is available here: `https://github.com/piotrkawa/audio-deepfake-adversarial-attacks`.

other fields like audio processing or computer vision: Fast Gradient Signed Method (FGSM) [172], Projected Gradient Descent (PGD) [173], and Fast Adaptive Boundary (FAB) [174] attacks. FGSM and FAB attacks were based on a L_∞ norm, whereas PGDL2 was on L_2. For more details on these attacks, please refer to Section 2.5.

We present each of the attacks above in three variants based on the levels of their parameters. They correspond to the degree of the degradation of the processed sample (visibility of the manipulation) and are often correlated with the performance of the attack. DeepFakes aim to trick people into believing that a given utterance is genuine, so their quality should be as close to human speech as possible. Meanwhile, the adversary should degrade the audio file significantly to increase the possibility of a successful attack. This means that the proper adversarial attack on the DF detection system should comply with two mutually exclusive properties — the attacked utterance should be as least distorted as possible to still sound convincing to humans and contain enough distortions to trick the classifier successfully.

In order to comply with these requirements, we selected the parameters of the attacks based on the evaluation of ten volunteers. The participants consisted of five women and five men aged 20-50. Six of them had technical backgrounds, but none of the individuals were professionally involved in speech or audio processing. By asking about their opinion on the quality of the adversarially attacked samples, we selected the parameters where the introduced distortions were imperceptible. We then considered two following levels of distortions where the artifacts were hearable; however, the utterances were still illegible, and the participants could successfully identify content and speakers. The selected parameters for the attacks were as follows: for FGSM, we used $\varepsilon \in \{0.0005, 0.00075, 0.001\}$, we performed PGDL2 for ten steps using $\varepsilon \in \{0.1, 0.15, 0.20\}$, whereas for FAB we applied parameters $\eta \in \{10, 20, 30\}$.

We additionally provide a quantitative analysis of the influence of the attacks as mentioned above. To do so, we reported mel-cepstral distortion (MCD) [197] (also referred to as mel-cepstral distance) which is an algorithm often used in speech-related tasks to assess the similarity (quality) of the generated samples compared to the source utterance. The MCD score indicates the similarity between MFCC obtained from the source and target utterances. The lower the value of MCD, the more significant the similarity between the two, which can be interpreted as a closer resemblance of these signals in terms of spectral characteristics. The MCD value computed between two identical signals is equal to 0.

Table 5.1: The mel-cepstral distortion (MCD) values between original and attacked samples. We report values on the samples that successfully fooled the classifiers.

	FGSM	
$\varepsilon = 0.0005$	$\varepsilon = 0.00075$	$\varepsilon = 0.001$
2.602	3.460	4.182
	PGDL2	
$\varepsilon = 0.1$	$\varepsilon = 0.15$	$\varepsilon = 0.20$
2.445	3.637	4.450
	FAB	
$\eta = 10$	$\eta = 20$	$\eta = 30$
2.311	3.508	4.469

The data in Table 5.1 indicates that more powerful attacks contributed to more significant differences between altered and original samples. The most remarkable differences can be observed in the case of the FAB ($\eta = 30$). Figure 5.1 depicts a spectrogram of a successfully spoofed sample along with its original.

5.3 Models

In our experiments, we considered three neural networks: LCNN [123], SpecRNet [7] and RawNet3 [196]. The reasoning behind this choice was the following: LCNN is among the most popular and effective methods, SpecRNet is an example of a small architecture providing reliable results, whereas RawNet3 is the next version of RawNet2 [127], another state-of-the-art DF detection architecture, initially used in speaker recognition [126]. The newer version significantly improves over its predecessor. Therefore, we concluded that this would allow us to build on generalization aspects which were problematic in some of our research (see Section 3 and Section 4). Please note that, to the best of our knowledge, this work was the first to adapt RawNet3 to DF detection.

One can further divide the presented methods based on their approaches to representing the audio data: RawNet3 processes raw waveforms, whereas LCNN and SpecRNet are based on front-end representations. For this research, we selected linear frequency cepstral coefficients (LFCC) front-end due to the satisfying results we obtained using it in our previous works (refer to Section 3 and 4). The exact parameters were: frequency of 16 kHz,

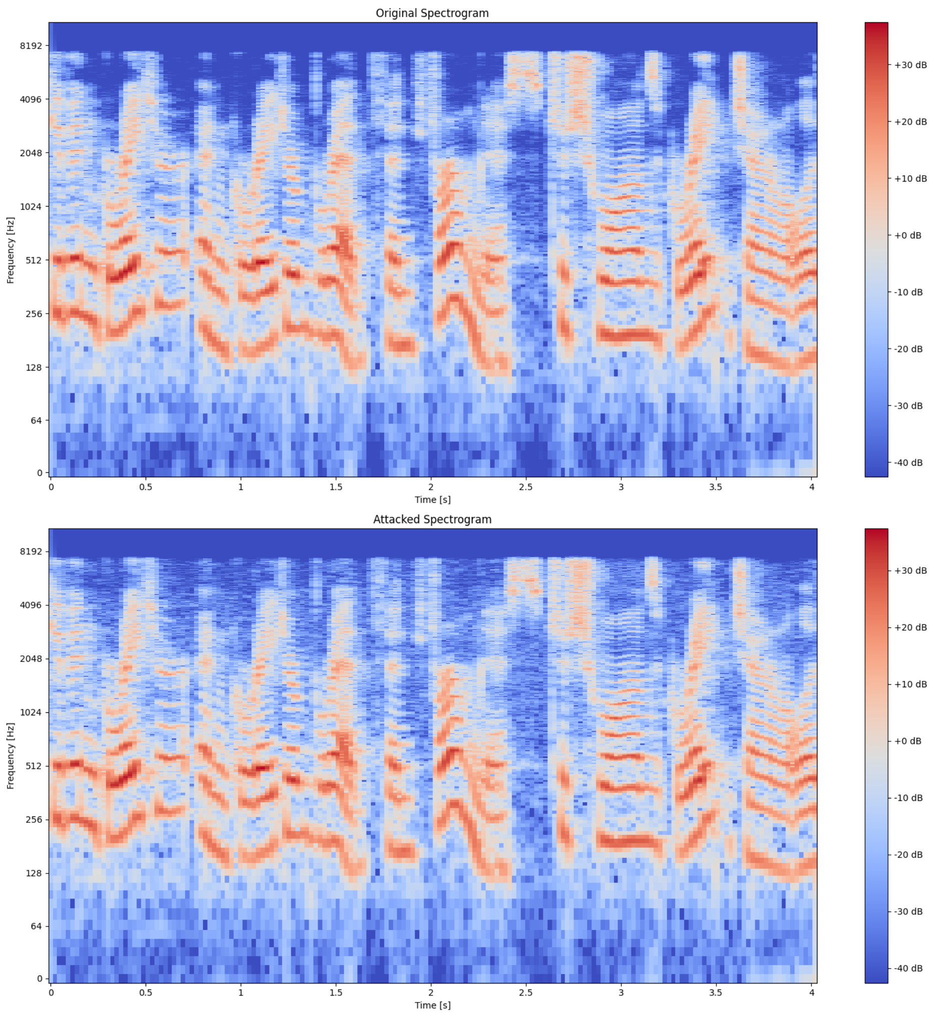


Figure 5.1: The comparison of the spectrograms of an utterance before (above) and after (below) applying adversarial noise (FGSM $\varepsilon = 0.001$). Presented utterance is a synthesized sample that successfully fooled one of the evaluated classifiers.

512 FFT points, Hann windowing algorithm (25 ms) with 10 ms window shift and 80 cepstral coefficients. The used architectures also differ in their complexity, containing 467,425 (LCNN), 15,496,197 (RawNet3), and 277,963 (SpecRNet) trainable parameters. To process 32 samples in a batch, RawNet3 requires 11,766 MB of VRAM, LCNN —— 2,300 MB and SpecRNet —— 1,299 MB. This also corresponded to the inference duration — with SpecRNet as a baseline, LCNN takes 52%, and RawNet3 — 608%, more processing time.

5.4 Datasets and preprocessing

The data used in our experiments was composed of three DeepFake datasets — ASVspoof 2021 (DF) [118], WaveFake [145] and FakeAVCeleb [147]. The reason for combining these datasets was the generalization issue raised in Section 2.5 and [152]. The combined datasets comprised 41,217 bona fide and 702,269 fake utterances. Spoofed clips were generated using over a hundred generation methods: one from FakeAVCeleb, eight from WaveFake and "over hundred" [118] from ASVspoof. The discussed experiments were based on the subset of this dataset — 100,000 training and 10,000 validation and test samples. The bona fide and fake classes were distributed in 50:50 proportion. This resulted in about 500 training and 50 validation and test samples per each DeepFake method. Bona fide samples were first divided across subsets using the same proportion (10:1:1). Then, we oversampled the training utterances to balance the classes. Please note that despite using only a subset of the combined datasets, our quantity surpassed recent works [194].

The preprocessing used in the training and testing complied with the community standard [145, 127, 6, 7] and was composed of resampling to 16 kHz mono, trimming silences longer than 0.2 s, followed by trimming or padding to about 64,600 frames (which corresponded to the duration of about 4 s). We did not use any data augmentation techniques (e.g. additive noises or compression), as they might have influenced the adversarial attacks.

5.5 White–box benchmark

The general pipeline looked as follows: detection models were first trained, then tested on pristine (not altered) data and finally evaluated using the adversarially altered samples. The discussed attacks were performed directly on the raw audio.

We trained the detection models for ten epochs, with a validation step performed after each. We selected the checkpoint for the subsequent testing stage based on the highest validation accuracy. The learning rates were set to 10^{-4} for LCNN and SpecRNet, and to 10^{-3} for RawNet3. SpecRNet and RawNet3 were additionally regularized using weight decay: respectively 10^{-4} and $5{\cdot}10^{-5}$. Moreover, to comply with the original training pipeline, we employed Stochastic Gradient Descent with Warm Restarts (SGDR) [198] procedure for the RawNet3. SGDR is a learning rate scheduling technique

Table 5.2: The EER results on the original test set of the data (before performing the adversarial attacks).

Model	**EER**
LCNN	0.0227
SpecRNet	0.0258
RawNet3	0.0180

extending the standard Stochastic Gradient Descent (SGD) optimization algorithm to train neural networks. While SGD involves updating model parameters based on a mini-batch of training data, SGDR extends this concept by using *warm restarts*. This involves first applying learning rate decaying — a popular technique whose value decreases with successive steps [199]. In SGDR, after several steps (or epochs), the algorithm returns to the original learning rate by starting the annealing procedure again. As a result, convergence during training can be noticeably accelerated. Please note that our restarting frequency differed from the originally used in RawNet3 training, where the authors had eighty training epochs and restarted every sixteen epochs. By having ten epochs, we restarted after each of them.

We trained the models using binary cross-entropy loss function and batch sizes of 128 samples. To ensure the reproducibility of our results, we applied the same randomness seed of 42 in each process. We report the results using Equal Error Rate (EER).

When tested in the default (non-attack) scenario (see Table 5.2), all models achieved a good performance comparable to the results reported in the related work — EERs of 0.0227 (LCNN), 0.0258 (SpecRNet), and 0.0180 (RawNet3). Meanwhile, as presented in Table 5.3, using adversarial attacks abruptly decreased the classifiers' performance, making them ineffectual and, in most cases, worse than random guessing. The most devastating attack was performed with PGDL2 ($\varepsilon = 0.20$) on SpecRNet, which resulted in an EER of 0.9905. Table 5.3 also shows the disparity of the attacks' effectiveness on different architectures: PGDL2 ($\varepsilon = 0.20$) was the most destructive for LCNN and SpecRNet, whereas FAB($\eta = 30$) was for RawNet3. This shows the absence of a universally superior model. Moreover, FAB($\eta = 30$) exhibited the lowest level of effectiveness among the remaining models. Another discrepancy related to the efficiency with the increasing parameter values. While this was the case for FAB and PGD, it was not observed for FGSM, where the efficiency either decreased with the increase of the parameter or fluctuated.

Table 5.3: The EER values of the detection on the samples attacked using the white-box scenario. The bolded values represent the highest EERs for each of the attacks.

FGSM			
Model	$\varepsilon = 0.0005$	$\varepsilon = 0.00075$	$\varepsilon = 0.001$
LCNN	0.8126	0.7686	0.7072
SpecRNet	**0.9676**	**0.9386**	**0.8716**
RawNet3	0.2118	0.2095	0.2112
PGDL2			
Model	$\varepsilon = 0.1$	$\varepsilon = 0.15$	$\varepsilon = 0.20$
LCNN	0.9501	0.9716	0.9791
SpecRNet	**0.9803**	**0.9865**	**0.9905**
RawNet3	0.3102	0.4567	0.5499
FAB			
Model	$\eta = 10$	$\eta = 20$	$\eta = 30$
LCNN	0.6305	0.6315	0.6321
SpecRNet	0.6833	0.7394	0.7614
RawNet3	**0.7430**	**0.8265**	**0.8549**

The observations based on these results indicate that the successful execution of adversarial attacks requires the adversary's careful choice of the method and its parameters to perform maximum damage. Moreover, a simple heuristic of increasing the value of the parameters does not always correspond to the best results.

5.6 Transferability benchmark

Next, we evaluated the transferability scenario. Its pipeline differed from the white-box approach only in terms of the models used for preparing the attack — they were different from the targeted models. The presented results are based on the models trained in the previous steps, i.e. no new models were trained for this stage.

The results presented in Table 5.4 show a decrease in the effectiveness of transferability attacks when compared to the white-box approach. This is the expected outcome due to the different difficulties of these two scenarios. The lowest EERs can be observed in the experiments concerning RawNet3 — both as the target and as an attack model. This suggests that the model was robust to the attacks conducted with other architectures, but, at the same time, the artifacts created with its use did not harm other models.

Table 5.4: The results of the transferability scenario (spectrogram models based on LFCC front-end). We report parameter values in increasing order. The bold results denote EERs between smaller models.

FGSM				
Target Model	Attack Model	ε_1	ε_2	ε_3
LCNN	SpecRNet	**0.2500**	**0.2560**	**0.2530**
LCNN	RawNet3	0.0680	0.0984	0.1352
SpecRNet	RawNet3	0.1039	0.1683	0.2061
SpecRNet	LCNN	**0.2542**	**0.2835**	**0.2993**
RawNet3	LCNN	0.0795	0.0948	0.1082
RawNet3	SpecRNet	0.0750	0.0901	0.1003
PGDL2				
Target Model	Attack Model	ε_1	ε_2	ε_3
LCNN	SpecRNet	**0.1625**	**0.2375**	**0.2908**
LCNN	RawNet3	0.0322	0.0444	0.0630
SpecRNet	RawNet3	0.0746	0.1267	0.1916
SpecRNet	LCNN	**0.3526**	**0.4429**	**0.4867**
RawNet3	LCNN	0.0368	0.0813	0.1124
RawNet3	SpecRNet	0.0294	0.0549	0.0839
FAB				
Target Model	Attack Model	η_1	η_2	η_3
LCNN	SpecRNet	**0.0591**	**0.1207**	**0.1710**
LCNN	RawNet3	0.0228	0.0228	0.0228
SpecRNet	RawNet3	0.0249	0.0249	0.0249
SpecRNet	LCNN	**0.2129**	**0.2778**	**0.3025**
RawNet3	LCNN	0.1010	0.1149	0.1270
RawNet3	SpecRNet	0.0257	0.0408	0.0525

Although no model achieved the EER worse than random guessing, the highest EER was 0.4867 for PGDL2 ($\varepsilon = 0.20$), and many of the combinations were successfully compromised. The attacks on LCNN prepared with SpecRNet and vice versa scored, on average, EERs of 0.2001 and 0.3236, respectively, resulting in over eight and twelve times worse results than in the baseline scenario. This indicates that transferability attacks can effectively damage the classifiers.

To further investigate the transferability between LCNN and SpecRNet, the experiments should be performed using another front-end instead of LFCC — e.g. MFCC. We assumed that the results were related to the same front-end of the models — they both used LFCC and due to the

Table 5.5: Transferability scenario results with spectrogram models using MFCC front-end. The bold results denote the most devastating attacks per architecture.

FGSM				
Target Model	Attack Model	ε_1	ε_2	ε_3
LCNN	(MFCC) LCNN	0.1132	0.1562	**0.1967**
SpecRNet	(MFCC) LCNN	0.1511	0.2074	**0.2482**
RawNet3	(MFCC) LCNN	0.0824	0.1012	**0.1107**
PGDL2				
Target Model	Attack Model	ε_1	ε_2	ε_3
LCNN	(MFCC) LCNN	0.1613	0.2904	**0.3820**
SpecRNet	(MFCC) LCNN	0.2305	0.3300	**0.4220**
RawNet3	(MFCC) LCNN	0.0309	0.0569	**0.0847**
FAB				
Target Model	Attack Model	η_1	η_2	η_3
LCNN	(MFCC) LCNN	0.1606	0.2226	**0.2484**
SpecRNet	(MFCC) LCNN	**0.1659**	0.1502	0.1324
RawNet3	(MFCC) LCNN	**0.1155**	0.0984	0.0757

vast differences in the architectures, that was their only common module. The preliminary results presented in Table 5.5 show the performance of the attacks prepared using LCNN with the MFCC front-end. The results are comparable to the ones reported in Table 5.4, and therefore, we conclude that the similarity of the networks' sizes is the main reason behind the good transferability between these models. Architectures of different sizes tend to capture different patterns. Therefore, an attack prepared with a model of vastly different size and construction may produce artifacts that are not crucial for the attacked model and, therefore, effectively decrease the performance of the prepared attacks.

5.7 Adversarial Training

The following section contains our research on making the DeepFake detection models more robust against adversarial attacks. We chose LCNN as a model for our experiments. The main reason for that was the low robustness exposed in Section 5.5 and Section 5.6. Moreover, LCNN is one of the most popular detection architectures, making it very likely to be used in deployed detection systems.

Currently used methods for circumventing adversarial attacks cover preprocessing input by, e.g. removing perturbations using JPEG compression [200], denoising or random input transformations [201]. However, the most popular and effective-proven approach is *adversarial training* [168, 202]. This term covers a set of training strategies involving various use cases of adversarial samples in the training process. The original idea concerns training the models using the original dataset and later fine-tuning them using only adversarial data [168]. This approach, however, addresses only one type of attack. In other procedures, batches of training data are constructed using different attacks (and even the original, non-altered samples). These adversarial samples can also be prepared using an ensemble of different parameters and models [203]. This approach, however, does not consider the complexities of particular attacks — artifacts like background noise can make the attacks simpler to detect and, therefore, easier to learn.

Another branch of the adversarial training is *adaptive adversarial training*. These solutions do not apply the same strategies to every sample; instead, they adapt according to the characteristics of the particular samples. In [204], authors got rid of uniform perturbation margins (distances from inputs to a decision boundary) for every training sample for the sake of sample-specific perturbation margins. They hypothesized that applying one strategy to all samples results in weaker generalization capabilities.

This section presents our method of *adaptive adversarial training*. The motivation for this was the limited robustness we achieved during standard adversarial training approaches. We propose a novel adaptive approach to address the AAs robustness issue and circumvent phenomena like catastrophic overfitting [205] or generalization issues. We analyzed standard adversarial training strategies — starting with fine-tuning using adversarially modified samples, then using a batch of original and modified data (in a 50:50 ratio) and ending with populating a batch using random sampling. From our observations, the models tended to focus on only a subset of the attacks and, as a result, were resistant only to them.

We addressed these issues, as our *adaptive* approach involves continuous analysis of the *difficult* types of samples, where the difficulty is assessed by the loss value for a particular (non-)attack — the greater the loss, the more *difficult* the type. The decision on the construction of the training batch (whether a particular attack is included or skipped) is based on a sampling vector $w = (w_0, \ldots, w_N) \in {\mathbb{R}_+}^{N+1}$, that $\sum_{i=0}^{N} w_i = 1$, where N is the number of attacks, and w_0 is a non-attack scenario. We assess the model's performance based on a loss of a previous iteration. The pseudocode for our approach is shown in Algorithm 1.

Algorithm 1: Adaptive update of the sampling weights vector.

```
Input  : $w$ — $N+1$-dimensional sampling vector,
         $loss_i$ — loss result,
         $i \in \{0, \dots, N\}$ — (non-)attack index,
         $c$ — clip value (default: $1$),
         $m$ — momentum ($\frac{1}{5}$),
         $p$ — non-attack proportion ($\frac{1}{3}$).
Output : updated sampling vector $w$
$w_i \leftarrow m \cdot \min(loss_i, c) + (1-m) \cdot w_i$
$s_w \leftarrow \sum_{k=0}^{N} w_k$
for $k \leftarrow 0$ to $N$ do
    if $k = 0$ then
        $r \leftarrow p$
    else
        $r \leftarrow \frac{1-p}{N}$
    end
    $w_k \leftarrow \frac{1}{2} \cdot \frac{w_k}{s_w} + \frac{1}{2} \cdot r$
end
```

Choosing the ”easy” attacks can be formalized as follows: the sampling vector’s elements (w_i) are $w_i \gg w_j$ for $j \in \{0, \dots, N\} \setminus \{i\}$ resulting in choosing only them w.h.p. To counteract it, we employ two control mechanisms. Firstly, instead of updating w_i with loss value, we constrain it using clipping constant c. Furthermore, we average the whole sampling vector w with a constant ratio r. The reason for that is to prevent outliers. Secondly, we utilize a constant parameter p, which denotes the proportion of sampling original (non-altered) samples. Using it to compute constant ratio r allows us to keep the model robust in the non-AA scenario.

The presented experiments covered fine-tuning the baseline model for ten epochs with the same training hyper-parameters (cf. Section 5.5). The reason behind this approach was a problem with convergence when training from the start (a similar approach was used in [194]). To prevent catastrophic overfitting, after each epoch, we validate the models’ performance using the formula:

$$e^* = \underset{e \in \{1, \dots, 10\}}{\operatorname{argmax}} \frac{(N+1) \prod_{i=0}^{N} a_i^e}{\sum_{i=0}^{N} a_i^e}, \tag{5.1}$$

where a_i^e is an accuracy of the i-th (non-)attack index after e-th epoch. This allows us to select for further evaluations a checkpoint from such an epoch e^*, where, on average, a performance across all (non-)attacks is the best.

Table 5.6: The EER results on white-box attacks of the classifiers trained using *adaptive* adversarial training. All the results are enhanced in relation to the baseline (cf. Table 5.3.)

FGSM			
Model	$\varepsilon = 0.0005$	$\varepsilon = 0.00075$	$\varepsilon = 0.001$
LCNN	**0.0985**	**0.1079**	**0.1144**
PGDL2			
Model	$\varepsilon = 0.1$	$\varepsilon = 0.15$	$\varepsilon = 0.20$
LCNN	**0.1245**	**0.1511**	**0.1870**
FAB			
Model	$\eta = 10$	$\eta = 20$	$\eta = 30$
LCNN	**0.0982**	**0.1116**	**0.1246**

The results presented in Table 5.6 show that fine-tuning using our adaptive adversarial training can significantly enhance the robustness of the model in a white-box scenario — all of the combinations provided better detection performance in relation to the baseline scenario. At the same time, the enhancement in the performance in the transferability scenario was observed in 50% of all cases (refer to Table 5.7).

Table 5.7: The results of transferability attacks on models trained using adversarial training. Bold cells refer to the enhanced results (cf. Table 5.4).

FGSM				
TM	AM	ε_1	ε_2	ε_3
LCNN	SpecRNet	**0.1472**	**0.1515**	**0.1560**
LCNN	RawNet3	0.0994	0.1064	**0.1091**
PGDL2				
TM	AM	ε_1	ε_2	ε_3
LCNN	SpecRNet	**0.1162**	**0.1247**	**0.1434**
LCNN	RawNet3	0.0876	0.0920	0.0945
FAB				
TM	AM	η_1	η_2	η_3
LCNN	SpecRNet	0.1018	**0.1158**	**0.1229**
LCNN	RawNet3	0.0901	0.0902	0.0904

In addition to the tests on the attacked data, we evaluated the adversarially trained model on the initial, non-altered dataset: the EER of the fine-tuned LCNN was equal to 0.0882, in relation to 0.0227 scored before adversarial training. We consider this a satisfying result, which proves that the DeepFake detection models can become more robust against AAs without sacrificing much performance on the standard (non-altered) samples. The performance decline, however, exists — the system owners should assess if adversarial attacks are a significant threat in their scenario and how much performance they are willing to sacrifice for this resilience. The degree of resilience (inversely proportional to performance on normal data) can be controlled by the parameters of adversarial training, such as the number of epochs or learning rate.

5.8 Ablation study

The following section contains an ablation study of our adaptive adversarial training method. For demonstration and deeper analysis of this strategy, we compared four training approaches. Starting from the proposed *adaptive method* (A), we removed further components in the *adaptive without non-attack proportion p* ($A-p$), and *adaptive without non-attack proportion and momentum* ($A-p-m$) methods, ending up with the random sampling method (*random*). It is worth mentioning that this ablation study followed our steps in creating the final algorithm — due to the poor performance of the default adversarial training methods, we added multiple control mechanisms, resulting in the final solution.

The A method is an adaptive method presented in Section 5.7. The $A-p$ method differs from the Algorithm 1 in lines 6-10, where the sampling vector is updated via $w_k \frac{w_k}{s_w}$. Moreover, the $A-p-m$ method does not contain a momentum introduced in line 1 (its value is $m=1$). In the random method, we selected the next attack based on the random selection with uniform distribution out of the set of supported adversarial attacks and non-altered data. We apply value clipping for all the algorithms. Our experiments showed that its absence caused the convergence of the sampling vector to the one dominant element that was then chosen most frequently, resulting in the poor overall performance of the models.

Figure 5.2 shows the results of the ablation study. Our main goal was to achieve general robustness against adversarial attacks while providing acceptable performance for non-attacked data. We proposed an evaluation metric similar to the Equation 5.1 — multi-class F1 score calculated using (non-)attack accuracies to achieve that. It promoted a scenario where a small variance characterizes the performance results. The results of the $A-p-m$ method were worse than those of the random method. The presence

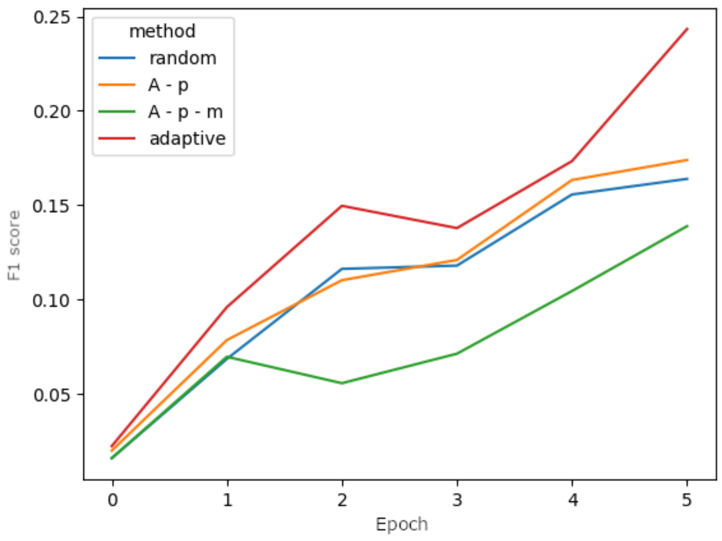


Figure 5.2: Our adaptive method provides the best F1 score among all of the four training methods covered in the ablation study. Source: [8].

of both momentum m and non-attack proportion p is crucial, especially in the early phase of the training, as the starting loss values for scenarios were relatively high and with no control mechanisms, they quickly became unbalanced (far from uniform). As a result, sampling used only a subset of the cases, resulting in overall weak generalization — training focused only on a subset of the attacks. To prevent this, we added the mechanisms of clipping and momentum. While this contributed to the performance enhancement, it was poor, only slightly overperforming the random strategy. Ultimately, by introducing non-attack proportion, we achieved the adaptive algorithm as presented in our work. It significantly enhanced the F1 scores compared to the other methods, resulting in superior performance. This ensured that the model retained its ability to differentiate between fake and bona fide samples.

5.9 Conclusions

In this chapter, we covered the topic of adversarial attacks on audio DeepFake detection methods. By introducing subtle, typically difficult-to-notice perturbations into input data, these samples can lead to erroneous predictions by the neural network. Using white-box and transferability attack scenarios, we showed that adversarial samples severely threaten the detection systems. As a result, even with limited knowledge about the system, the adversary can significantly decrease the systems' performance, often making them ineffectual. The average EER performance for the baseline models was equal to 0.0221, which we later degraded to up to 0.9905 (for white-box attack) and 0.4867 (for transferability attack).

In order to increase the robustness of these models, we introduced a novel method of adaptive adversarial training. Selecting training examples using our adaptive sampling strategy allowed the architectures to generalize well across multiple attacks and resist overfitting, decreasing EER down to 0.0982 for the white-box. In the transferability scenario, we managed to reduce EER to 0.1091. In this case, The best model's performance was equal to EER of 0.0876, as not all of the results were improved by the adversarial training.

Our ablation study proves that the introduced components are crucial for successful training. As a result of our adversarial training, the performance of the attacks significantly decreased, leading to acceptable results on the adversarially altered samples without compromising much of the performance on the non-altered data, which fulfils the Research Goal 3 (refer to Section 2.6) and proves that while adversarial attacks successfully degrade the classifiers' performance, this can be mitigated by the use of adversarial training.

Chapter 6

Using Whisper in detecting DeepFakes

Self-supervised learning (SSL) is an increasingly popular paradigm of deep learning [206]. This concept relies on training the neural networks using large amounts of unstructured and unlabeled data. The goal is to train the model meaningful structures, patterns, and relationships between data without costly human annotation. It is accomplished using general tasks such as autoencoding, image rotation prediction and contrastive learning [206]. Models trained on a large volume of data using a general task, later fine-tuned for a specific problem, often achieve much better results than the solutions based on standard supervised learning paradigms [206, 138, 207].

In the field of speech processing, the most prominent examples of SSL architectures include WavLM [208], wav2vec [209], wav2vec2.0 [137] and HuBERT [210]. Models fine-tuned for specific tasks achieve high performance in speech recognition or speaker identification [138, 207]. Features obtained using SSL architectures can also benefit tasks like audio DeepFake detection. Works such as [211] and [212] have shown that using the information provided by SSL-based front-ends leads to performance improvement and better generalization of the detectors in relation to the commonly used DSP-based front-ends (see Section 2.2). This suggests that exploring front-ends based on the information provided by the architectures trained on the large datasets may be an essential part of the research focusing on the generalization of DeepFake detection.

This chapter presents a novel front-end based on the Whisper automatic speech recognition model [5]. We show that by using the representations obtained from the encoder of the Whisper model, we enhance DeepFake detection results and show the improvement on a novel In-The-Wild dataset [152] by decreasing the baseline EER by 21%. We achieve that by using only

125,000 samples of the ASVspoof 2021 DF dataset without employing data augmentation techniques. Moreover, to the best of our knowledge, we were among the first to use the Whisper model successfully in tasks other than automatic speech recognition (ASR).

6.1 Whisper ASR model

Whisper [5] is a state-of-the-art automatic speech recognition (ASR) model developed by OpenAI. It was trained on 680,000 hours of multilingual and multitask speech in a weakly-supervised fashion. The paradigm refers to training models using noisy, inaccurate labels (e.g. imperfect transcripts), which, thanks to the volume of the data, can outperform fully-supervised approaches. The main similarity between Whisper and other SSL architectures is the large dataset used for their training procedures. The dataset used in the training of Whisper comprises 98 languages, with most content (54%) being English. The diversity and magnitude of the dataset surpassed each ASR dataset to date and allowed for robustness against environmental interferences like background noises, providing sufficient performance for many languages and accents.

The name *Whisper* refers to the whole family of architectures differing in their sizes (e.g. by width and sizes of layers) — from *tiny* (39 million trainable parameters) to *large* (1550 million parameters). The authors additionally provided separate models based only on the English language — denoted **.en* e.g. *tiny.en*. Besides the general speech recognition task (for which multiple languages are supported), Whisper enables language identification and translation from the supported languages to English. The high performance achieved across multiple tasks makes this model one of the most commonly used speech-processing models nowadays.

Whisper architecture is based on the default encoder-decoder Transformer [214] (cf. Figure 6.1). In our experiments, we focused on the capabilities of its encoder. The module first comprises two blocks of a GeLU activation function [215] and a convolutional layer. The signal is later modified by adding position embeddings [214]. The architecture ends with blocks of the pre-activation residual attention blocks [216], followed by the normalization layer.

In this chapter, we use the encoder of the pre-trained Whisper not to extract features relevant to speech recognition but to capture features rel-

The source code related to this research is available here: `https://github.com/piotrkawa/deepfake-whisper-features`.

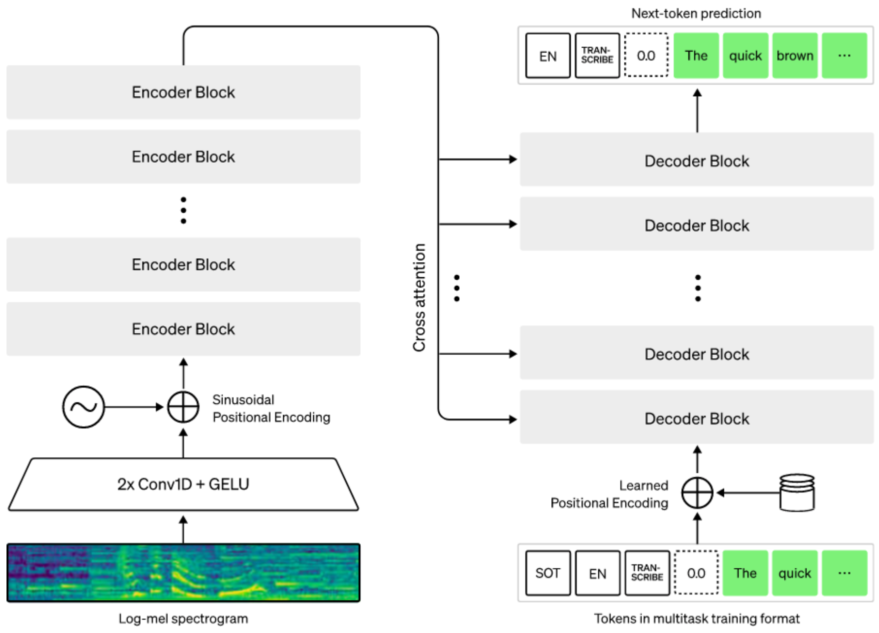


Figure 6.1: The architecture of Whisper is based on an off-shelf Transformer model. Source: [213].

evant to the problem of audio DeepFake detection. We treat the encoder output as an input to three front-end-based detectors and show that these features can be meaningful in DeepFake detection, thus enhancing performance. We selected Whisper for this task for multiple reasons: first and foremost, it was trained using a vast amount of data — the training set is more than 16x larger than the one used to train for wav2vec2.0 [209]. The data was diverse, containing 98 languages and including many different accents. In addition, Whisper is robust to natural noise and interferences occurring within the analyzed utterances [5]; hence, we conclude that the features will be more meaningful in identifying artificially generated speech. This, in turn, can help with the generalization problem — the poor quality of results on samples on out-of-distribution data, which is one of the main problems we address in the dissertation.

We conducted experiments using the smallest version of Whisper — *tiny.en*. Detecting DeepFakes is a global problem; therefore, by adding a minimal possible overhead, we want to make the presented solution as accessible as possible. Please note that we use the **.en* version of Whisper, which was trained using only the English subset of the training dataset. The reason is that English is currently the most widely used language in DF detection datasets (however, recent datasets introduce other languages [12]).

Please note that larger versions of Whisper have demonstrated even

Table 6.1: Configuration parameters of Whisper's *tiny.en* encoder used in the experiments.

Name	Value
Mel dimensionality	80
Temporal dimension	1,500
Hidden state dimension	384
Attention heads number	6
Residual attention blocks number	4

more impressive performance across various tasks. For example, the *large* model outperforms the *tiny.en* model by up to threefold in tasks such as speech recognition and translation [5]. This indicates that the presented results can be enhanced further.

6.2 Models

In our study, we evaluated four architectures. Three of the models processed front-end features: LCNN [124], MesoNet [11] (MesoInception-4), and SpecRNet [7]. We additionally included RawNet3 [196], which, on the contrary, analyzes raw audio — this model took the place of RawNet2 due to the high performance achieved in Chapter 5. The parameter counts for these models are 467,425 (LCNN), 28,486 (MesoNet), 277,963 (SpecRNet), and 15,496,197 (RawNet3), respectively. We based on the encoder of *tiny.en* version of Whisper, which contains 7,632,384 trainable parameters. One can easily note that even with the Whisper front-end, the architectures are comprised of nearly 50% fewer parameters than RawNet3. For more details on the architectures, please refer to Section 2.3.5.

6.3 Front-ends

For the front-end-based models, we used the following representations: MFCC, LFCC and the output of Whisper's encoder. We compared Whisper with mel- and linear-frequency features due to their popularity and well-established position in many audio tasks, including DeepFake detection [176, 124]. Please note that our research additionally included constant Q-cepstral coefficients (CQCC) [81], a popular anti-spoofing front-end [217, 212]. We initially included the LCNN + CQCC model in our comparison. However, the training procedure based on parameters as for other feature extractors provided unsatisfactory results, resulting in around 60% accuracy on the ASVspoof 2021 DF dataset. Low performance was the reason for the ceasing the further evaluation.

We based LFCC and MFCC (referred to as *non-trainable* front-ends) on the following parameters: sampling rate of 16 kHz, a window size of 400 frames, hop length of 160 frames and 128 coefficients. We enriched the *non-trainable* front-ends by providing their delta and double delta features [74]. These representations are widely used in speech-processing tasks to express the dynamics of the audio signal properties over time. Delta features are the differences between signal features. These transformations can be intuitively compared to discrete image derivatives, e.g. Sobel filters, which highlight particular features. For given feature f (e.g. LFCC or MFCC) at time-instant k, the delta feature is defined as:

$$\Delta_k = f_k - f_{k-1}$$

Analogously, double delta features are the differences between delta feature vectors:

$$\Delta\Delta_k = \Delta_k - \Delta_{k-1}$$

Whisper analyzes audio chunks of 30 s. To equalize the sizes of all front-ends used throughout the experiments, we also decided to base on this duration in the *non-trainable* front-ends. As a result, their size was equal to 376×3000, where 376 was the number of front-ends' components concatenated with its delta and double delta features, whereas 3000 was the number of frames. As mentioned, we used a *tiny.en* version of Whisper. Its encoder returns data in the shape of 376×1500. To conform to the shape of MFCC and LFCC features, we replicated the second dimension, resulting in a tensor of size 376×3000. We state that such an operation should not have a negative influence, as we expose a model to the same features twice. Please note that the encoders in different Whisper architectures return tensors of other (larger) shapes, and to use them with our models, one should modify network parameters accordingly.

The shapes of the presented front-ends were larger than the ones used in our previous research — the standard architectures of the models could not process these inputs. We adapted them to make processing this higher-dimensional data possible. In the SpecRNet model, we included a layer of adaptive 2D average pooling right after the ultimate SeLU layer. In MesoNet, we introduced an adaptive 1D average pooling before the penultimate fully connected layer. Adaptation of LCNN included increasing the number of input features and hidden features of two bi-LSTM layers, as well as the input features of the last linear layer, from 160 to 768.

6.4 Setup

The baseline setup for all the experiments was the following: each model was first trained on the same randomly selected subset of 100,000 training and 25,000 validation samples from the ASVspoof 2021 DF dataset [118]. Checkpoint, which scored the highest validation accuracy, was chosen for the testing procedure on all utterances of the In-The-Wild dataset (refer to Section 2.4.4).

Since we aimed for the most accessible solution (in terms of training and inference), we have chosen the smallest architecture of Whisper and performed training using only a subset of the data available in the ASVspoof dataset. 125,000 training samples still exceeded some of the related work, moreover, such quantity allowed a model's training using a single GPU (NVIDIA TITAN RTX GPU, 24 GB VRAM) in about a day. Please note that using bigger and more diverse (e.g. through augmentation techniques [139]) datasets can lead to even better performance.

The random selection of samples for the training and validation sets resulted in a disproportionate number of samples per class, which we addressed using an oversampling procedure.

We conducted each training procedure for ten epochs and validated models using the entire validation subset at the end of each. We used a batch size of eight, which differed from the previous chapters due to the increased input size. Models were optimized using the binary cross-entropy loss function. We used a learning rate of 10^{-4} and a weight decay of 10^{-4} for all front-end-based models. RawNet3 was trained using a learning rate of 10^{-3} and a weight decay of $5 \cdot 10^{-5}$, additionally involving SGDR scheduling with a restart after each epoch (see Section 5.5). We ran each process with a fixed randomness seed of 42 to ensure our results were reproducible.

6.5 Preprocessing

Each audio sample used throughout our experiments underwent a preprocessing procedure based on the standard approaches commonly used in the relevant literature (cf. Section 3, Section 4 and [145, 127]). It consisted of resampling to 16 kHz mono, removing silences longer than 0.2 s and padding (repeating) or trimming content to the predefined duration. As stated above, instead of using a default duration of about 4 s, we used 30 s due to the requirements of the Whisper model. The official implementation [218] pads shorter utterances with zeros, whereas we repeated the data. In our opinion, exposing models several times to the same information would allow more reliable comparison, as no new information is provided, and the whole audio file, similarly to 4 s samples, is filled with speech. Moreover, works like [152] show that analyzing longer sequences benefits performance. For this reason, we standardized the durations for all front-ends — a longer input to the Whisper front-end might contain a larger number of artifacts, making the comparison of front-ends unreliable.

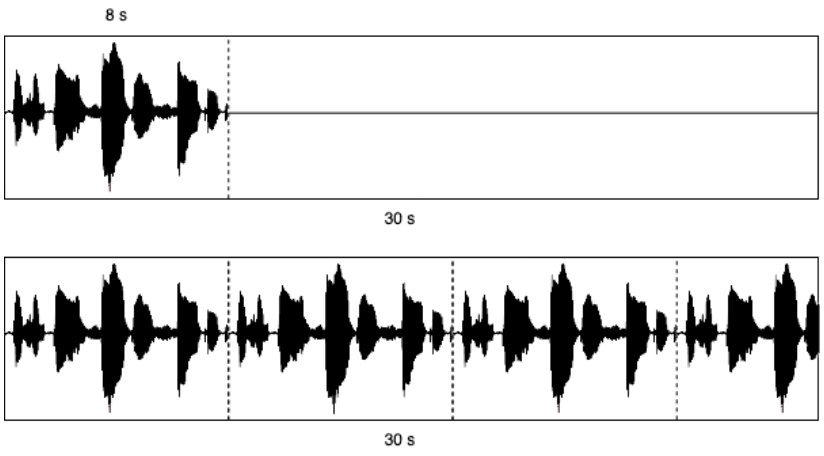


Figure 6.2: Standard Whisper preprocessing pads the input with zeros to chunks of 30 s (top). Instead, we repeat (or trim) the processed samples (bottom).

6.6 Baseline comparison

As our baseline comparison, we considered LCNN, SpecRNet, MesoNet, and RawNet3 models and tested LFCC, MFCC, and Whisper front-ends. Please note that during the training procedure, the weights of Whisper's encoder were not optimized (i.e., were frozen). We treated this model as a feature extractor using the information obtained from its original training procedure on a large-scale dataset.

The test results obtained on the In-The-Wild dataset, presented in Table 6.2, significantly diverge from the low EERs reported in the earlier chapters of the dissertation. The reason for that is the controlled manner in which related work is mostly based during the creation of the datasets — even though it is a result of the community-based effort of dozens of

Table 6.2: The EER results of the models tested on the In-The-Wild dataset. The front-end, composed of Whisper's encoder, significantly enhances the results for SpecRNet and LCNN networks. The lowest reported EER is presented in bold.

Model	**Front-end**	**EER**
SpecRNet	LFCC	0.5184
SpecRNet	MFCC	0.6897
SpecRNet	Whisper	0.3644
LCNN	LFCC	0.7756
LCNN	MFCC	0.6762
LCNN	Whisper	0.3567
MesoNet	LFCC	0.5451
MesoNet	MFCC	**0.3132**
MesoNet	Whisper	0.3856
RawNet3	-	0.5199

distinct teams like ASVspoof and contains additional postprocessing procedures. In-The-Wild is composed of real-world samples of not only the unknown generation method but also unknown processing algorithms, including artifacts from the compression during upload to social media. This results in the distribution of features vastly different from the ones seen (and learned) in the training procedure. As the models cannot generalize sufficiently, the detection capabilities deteriorate significantly when exposed to different data.

Of the ten presented models, six performed worse than random guessing (EER=0.5). The models that scored unsatisfactory results on the In-The-Wild test dataset at the same time provided satisfying efficacy on the validation part (taken from ASVspoof), e.g. LCNN with LFCC and SpecRNet with MFCC (two worst architectures on In-The-Wild), achieved EERs of 0.0149 and 0.0218, respectively.

The best EER of 0.3132 was obtained using the MesoNet based on the MFCC front-end. The architecture achieved similar results in the original work (EER=0.3741) [152] and was among the best front-end-based models. Please remember that in the original In-The-Wild work, this model was trained on ASVspoof 2019 and based on a log spectrogram considering the entire duration of analyzed utterances. We conclude that the reason behind these results is a small number of model parameters. MesoNet, which contains roughly 30k parameters, learns the artifacts from the ASVspoof to a smaller degree than the other architectures and, therefore, is characterized by better generalization on other datasets. SpecRNet was the model that scored, on average across front-ends, the second-best results. The results in Table 6.2 also show the positive impact of Whisper-based features on the generalization of the models. Using LFCC as a baseline, SpecRNet achieved an enhancement of 29.71%, LCNN 54% and MesoNet 29.26%. The enhancements were even greater for MFCC in the case of the two first models.

6.7 Concatenated features

The following section covers our evaluation of the models based on concatenated front-end features. We motivate this with the results we presented in Chapter 3, where we showed that using front-ends composed of several algorithms can increase performance. Our second aim of these experiments was to investigate the relations between the front-ends. In this section, we consider models based on the concatenation of Whisper-features and *non-trainable* front-ends.

Table 6.3: The test results of the models based on concatenations of front-end features. Including Whisper improves the results of *non-trainable* front-ends and meanwhile degrades them in relation to using Whisper only (for all but two architectures – presented in bold).

Model	Front-end	EER
SpecRNet	Whisper + LFCC	**0.3485**
SpecRNet	Whisper + MFCC	0.4116
LCNN	Whisper + LFCC	0.6270
LCNN	Whisper + MFCC	0.6117
MesoNet	Whisper + LFCC	0.8029
MesoNet	Whisper + MFCC	**0.3822**

Table 6.3 presents the test results of the concatenation of front-end features. No model achieved EER better than the best reported in the first benchmark (0.3132). However, using Whisper with other front-ends enhanced the results for SpecRNet and LCNN by up to 40.32% and 19.15%

If we look at the results from the perspective of *non-trainable* front-ends, joining them with Whisper indicated a positive synergy between them, leading to the gain of additional knowledge and enhancing the results. Despite alleged gains, only two of the architectures outperformed the results of Whisper-only models. The reason for that could be the phenomena of

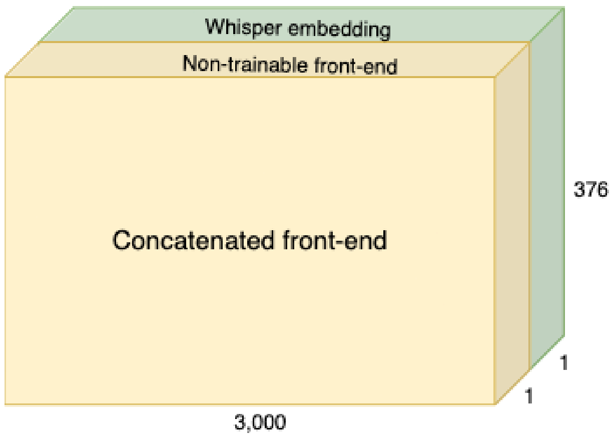


Figure 6.3: Visualization of concatenated features — we join *non-trainable* front-end (MFCC or LFCC with its delta and double delta features) and Whisper embedding in a channel dimension. The dimensionality of a single sample is equal to $2 \times 376 \times 3000$.

covering some vital Whisper features by the features of spectrogram front-ends.

Conversely, looking at the results from the perspective of the Whisper front-end, using other features degraded the results for all but two architectures — the decrease was not substantial for MFCC but significant for LFCC. We conclude that this negative synergy came from architecture's details and *compression* of the provided information, which resulted in the classification based on the *noised* information rather than an enhanced set of features. We investigate this phenomenon further in Section 6.10.

6.8 Fine-tuning

In the benchmarks described above, we treated Whisper's encoder as a front-end algorithm — a predefined function returning specific audio representations. In the following section, we describe the process of adapting Whisper's encoder to the problem of DeepFake detection by fine-tuning. We based on the following intuition: Whisper was trained on a dataset where the vast majority of speech was non-generated, which should be considered biased in terms of the DeepFake detection problem; moreover, it was trained for different purposes: speech recognition and language identification. The fact that the provided features significantly increased performance over other algorithms (cf. Table 6.2) is a rationale for the assumption that fine-tuning Whisper may result in even further improvements.

In the following benchmark, we used the models trained during the experiments described in Section 6.6 and Section 6.7. We considered only architectures using Whisper or concatenated with another front-end. We fine-tuned these architectures for an additional five epochs with the unfrozen Whisper weights using the learning rate of 10^{-6}.

Table 6.4: Results of the Whisper-based architectures. *EER (frozen)* refers to the Table 6.2 and Table 6.3 results and *EER (tuned)* to the results of fine-tuning Whisper-based models. The lowest EER for each of the approaches is presented in bold.

Model	**Front-end**	**EER (frozen)**	**EER (tuned)**
SpecRNet	Whisper + LFCC	**0.3485**	0.3795
SpecRNet	Whisper + MFCC	0.4116	0.3769
SpecRNet	Whisper	0.3644	0.3338
LCNN	Whisper + LFCC	0.6270	0.6270
LCNN	Whisper + MFCC	0.6117	0.5899
LCNN	Whisper	0.3567	0.3290
MesoNet	Whisper + LFCC	0.8029	0.5526
MesoNet	Whisper + MFCC	0.3822	**0.2672**
MesoNet	Whisper	0.3856	0.3362

Table 6.4 compares the results achieved in the first two benchmarks and the ones obtained after fine-tuning. Fine-tuning enhanced the results for all models except for SpecRNet with Whisper+LFCC front-end (the results degraded by less than 9%). We improved the previously best result of MFCC-based MesoNet by 14.69% — the best architecture was based on MFCC and Whisper with MesoNet network and achieved an EER of 0.2672. This also improves the original work [152], where RawNet2 [127] achieved an EER of 0.3394. Please note that the model was trained on 4 s samples from ASVspoof 2019 [111].

The results indicated that utilizing the features obtained with the Whisper model, trained on a vast amount of data, and later fine-tuned to the downstream task may significantly improve performance. Note that the large training set used for Whisper is particularly important due to the scarcity of DeepFake-specific training data, as it is magnitudes larger than all DeepFake datasets combined.

6.9 ASVspoof 2021 DF tests

Apart from the tests on the In-The-Wild dataset (refer to Sections 6.6, 6.7 and 6.8), we additionally tested every presented model on a subset of 100,000 randomly selected samples from the ASVspoof2021 DF dataset.

Table 6.5: The comparison of test EER results obtained on In-The-Wild (ITW) dataset and on the additional 100,000 random samples of ASVspoof 2021 DF (ASV21 DF) dataset. The used samples were outside of training and validation subsets. The lowest reported EERs for each of the datasets are presented in bold.

Model	Front-end	EER (ASV21 DF)	EER (ITW)
		Baseline	
SpecRNet	LFCC	0.0208	0.5184
SpecRNet	MFCC	0.0299	0.6897
SpecRNet	Whisper	0.0407	0.3644
LCNN	LFCC	0.0140	0.7756
LCNN	MFCC	0.0099	0.6762
LCNN	Whisper	0.0212	0.3567
MesoNet	LFCC	0.1242	0.5451
MesoNet	MFCC	0.1941	0.3132
MesoNet	Whisper	0.1234	0.3856
RawNet3	-	0.0136	0.5199
		Concatenated features	
SpecRNet	Whisper + LFCC	0.0187	0.3485
SpecRNet	Whisper + MFCC	0.0264	0.4116
LCNN	Whisper + LFCC	0.0091	0.6270
LCNN	Whisper + MFCC	0.0101	0.6117
MesoNet	Whisper + LFCC	0.1552	0.8029
MesoNet	Whisper + MFCC	0.1580	0.3822
		Finetuning	
SpecRNet	Whisper + LFCC	0.0192	0.3795
SpecRNet	Whisper + MFCC	0.0105	0.3769
SpecRNet	Whisper	0.0105	0.3338
LCNN	Whisper + LFCC	0.0106	0.6270
LCNN	Whisper + MFCC	0.0091	0.5899
LCNN	Whisper	**0.0019**	0.3290
MesoNet	Whisper + LFCC	0.0455	0.5526
MesoNet	Whisper + MFCC	0.0072	**0.2672**
MesoNet	Whisper	0.0036	0.3362

Table 6.5 shows a significant disparity between the performances on

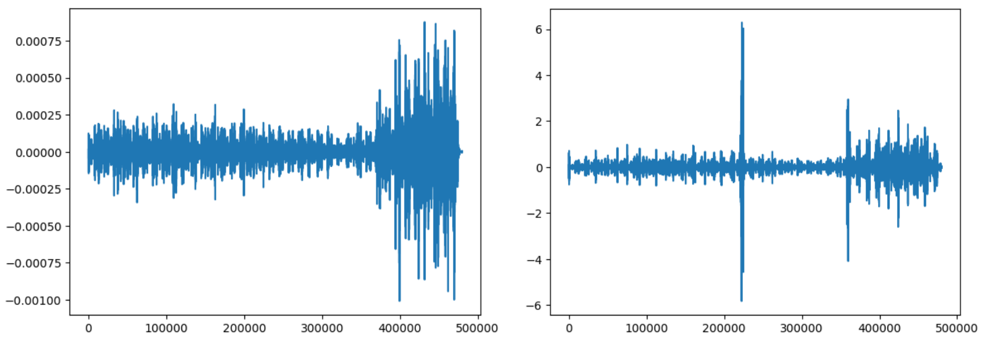
In-The-Wild and ASVspoof 2021 DF datasets. Most EERs obtained on ASVspoof are below 0.03, whereas those achieved on In-The-Wild are in the $[0.3, 0.5]$ range. This again shows that the artifacts present in the ASVspoof dataset are similar across different subsets and vastly different from those present in the In-The-Wild samples. The Table also contains other interesting information about the MesoNet architecture. The model provides the weakest performance in the case of *baseline* and *concatenated features* benchmarks, i.e. when the front-ends were not fine-tuned. However, the performance significantly increases once the whole model is fine-tuned. This phenomenon is investigated further in Section 6.10.

6.10 Features comparison

Due to the inconsistent results provided by the concatenation of the front-ends, we decided to investigate the influence of particular parts of the input data on the final decision of neural networks. For the analysis, we chose the neural networks differing in structure: SpecRNet, based on a recurrent layer (in particular Gated Recurrent Unit [130]) and MesoNet, mainly composed of max-pooling and convolution layers. We investigated by calculating and analyzing the gradient on the input data — the technique commonly used in adversarial attacks [173].

Figure 6.4: Gradient calculated on a spoofed signal for the SpecRNet models (LFCC and Whisper without fine-tuning). Note that the y-axis ranges differ.

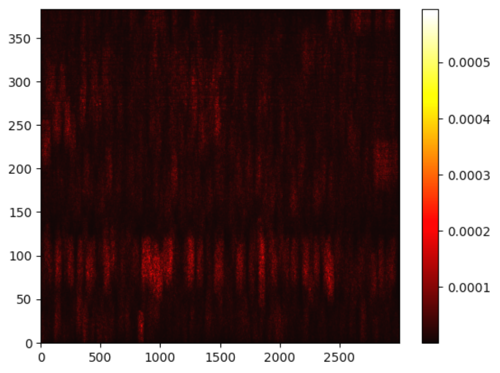
(a) LFCC (b) Whisper (frozen)



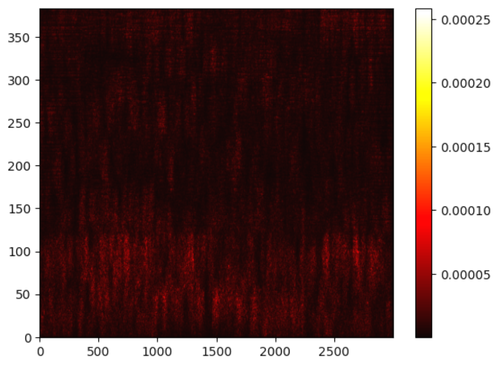
We observed that the model's architecture significantly influences the processing of front-end inputs. Figure 6.4 shows the gradients of the spoofed signal with respect to the audio frames analyzed by the two SpecRNet architectures (Whisper and LFCC). Even though the whole utterance is spoofed, the decision was made based only on some particular parts of this signal — as SpecRNet processes information using a GRU layer, the final decision is largely influenced by the latter part of the signal (refer to Figure 6.4a). Additionally, models based on Whisper features often rely on certain attributes extracted from specific narrow signal segments — as evidenced by the two peaks around frames 220,000 and 360,000 in Figure 6.4b. These peaks have much greater values than the gradients of the LFCC signal (please refer to the Y-axis of the plots). We conjecture that Whisper works effectively in conjunction with recurrent layers because it captures *prominent* features that tend not to be concealed traversing through the recurrent sequence.

Figure 6.5: Saliency map of bona fide utterance processed using MesoNet with different front-ends. Concatenating front-ends and fine-tuning a model discards some of the previously important information. Source: [9].

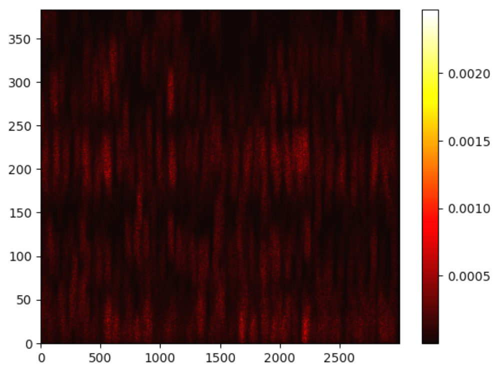
(a) LFCC



(b) MFCC



(c) Whisper (frozen)



(d) MFCC + Whisper (tuned) [MFCC channel]

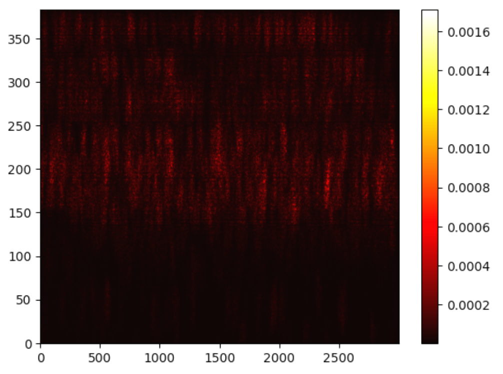
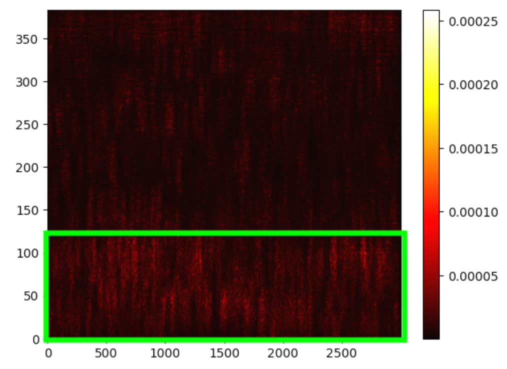


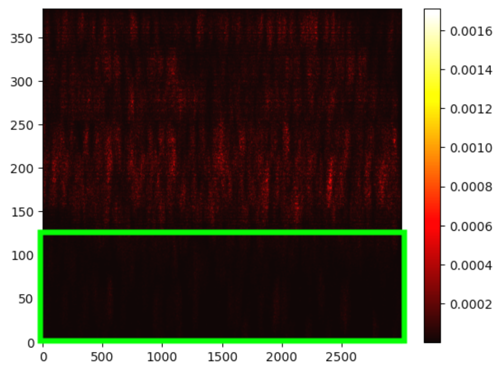
Figure 6.5 contains the saliency maps [219] of the bona fide samples processed by MesoNet architecture. Saliency maps are visual representations that highlight the most important regions within a given representation, e.g., an image or a spectrogram. They can be used to interpret the decisions of deep learning models in classification tasks — typically, the higher (the brighter) the values of a given region, the more important it is for the decision.

Figure 6.6: The comparison of saliency maps displayed in Figure 6.5. The area marked in green shows the part of the MFCC front-end that is considered important (due to its high values). As a result of the fine-tuning process, Whisper forces focus on the data that previously contributed less to the outcome, effectively discarding previously important information. We hypothesize that the reason for these positional features is due to the convolutions used in the model.

(a) MFCC



(b) MFCC + Whisper (tuned) [MFCC channel]



MesoNet comprises four max-pooling layers — the ultimate linear layers receive information that has been *max-pooled* from spatially distributed blocks measuring 32×32. In Figure 6.5, we show four different front-ends: LFCC, MFCC, Whisper (frozen weights) and Whisper (fine-tuned) concatenated with MFCC. The saliency maps have the same dimensions as the front-ends they are derived from: 376×3000, where 376 is the dimensionality of the parameters, and 3000 refers to the temporal dimension (the number of frames). Due to the concatenation of the Whisper features and MFCC in the channel dimension, we display only the MFCC part of this front-end. The areas with higher values, shown using brighter colours, are interpreted as the ones contributing to a greater degree to the final decision of the models due to the greater values of the gradient there.

As mentioned, we use the *non-trainable* front-ends along with their delta and double delta features — they are displayed in the following order: the lower band of the first 128 rows are either LFCC (cf. Figure 6.5a) or MFCC (cf. Figure 6.5b), whereas the next 256 are their first, and second derivatives. Saliency maps show that the areas with the highest values, contributing to the final results most significantly, were the initial front-end representations (LFCC and MFCC) — delta and double-delta features seem less important. We hypothesize the reason for that is the redundancy within the representation — model relied on the original information in the decision-making and disregarded delta and double-delta features. At the same time, standard, non-optimized Whisper features (see Figure 6.5c) did not manifest similar properties — the decision was made based on a full band, focusing on the middle.

This changed in the case of fine-tuning the front-end composed of Whisper and MFCC (refer to Figure 6.5d). The fine-tuning process resulted in performance improvement, and surprisingly, the importance of previously relevant MFCC decreased for the sake of the remaining part of the signal (delta and double-deltas). We assume that the Whisper front-end efficiently captures speech characteristics, and incorporating deltas could improve outcomes by adding extra coefficients to depict spectral dynamics. Additionally, please note the differences in the magnitudes of the gradients — the Whisper (frozen) features had a significantly greater gradient than MFCC or LFCC. We conclude that as we do not include any spatially independent processing mechanism to process a combination of the front-ends, the negative impact of Whisper features on other front-ends comes from its enforced locality.

The results in Section 6.8 show that using combined features can benefit the model's performance. On the other hand, we also show that combining such features can result in hiding previously considered important features. One could employ spatially independent processing mechanisms like attention [214] to enable greater synergy between the features.

6.11 Conclusions

In this chapter, we addressed Research Goal 4 (cf. Section 2.6) by introducing a novel front-end based on the Whisper [5] ASR model. Using it as a feature extractor and applying it to the task of DeepFake detection improved the performance of not only ASVspoof 2021 DF (used as a validation set) but, more importantly, on the In-The-Wild dataset, characterized by the significantly different distribution of the artifacts and known for its difficulty. By using a front-end composed of a fine-tuned Whisper, we achieved an EER of 0.33 $\pm$ 0.01 for all three front-end-based architectures (LCNN, SpecRNet, MesoNet). Moreover, using a fine-tuned model comprised of Whisper and MFCC features with a MesoNet back-end enabled us to achieve EER as low as 0.2672. We showed that even the smallest version of Whisper (*tiny.en*) can positively contribute to the generalization of the detection architectures and allow for achieving state-of-the-art results on the In-The-Wild dataset.

It is important to note that concurrently, a method with higher efficacy has been developed [164]. It achieved an EER of 0.0755 on the In-The-Wild dataset. However, these results were obtained using three different components, each of them separately increasing the results significantly: by using a larger front-end architecture (wav2vec2.0 [137], version XLSR-53 with 300 million parameters), applying RawBoost data augmentation [139] and using the different dataset (vocoded bona fide utterances) [164]. The Whisper results can be further enhanced by the number of components: using larger Whisper architectures, using multilingual variants of the model, employing additional front-end processing mechanisms (e.g. attention), using a larger training dataset and applying data augmentation techniques.

Chapter 7

Summary

The dissertation presents solutions to improve methods for detecting computer-generated speech, a problem known as *audio DeepFake detection*. The development of artificial intelligence methods has contributed to significant improvements in the field of speech synthesis in recent years. Despite positive applications such as voice assistants [31, 32, 33] or helping people with speech impariments [29], the methods can pose a threat due to many malicious applications. These include creating fake news [4], circumventing ASV systems [109], impersonation [108], or extortion [2, 3]. The content of this dissertation is based on four scientific articles published at peer-reviewed conferences. The content was further expanded to include additional experiments and analyses. Research conducted in the scope of each of the manuscripts comprises separate chapters.

We first focused on improving the generalization of DeepFake detection models. The corresponding content is described in Chapter 3, which is based on the work *Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection*. The chapter addresses Research Goal 1 — we introduced a test framework based on specific TTS and VC methods splits. Using different strategies to divide the DeepFake generators among subsets allows in-depth verification of the generalization and stability of models. In addition, we showed that the concatenation of different front-ends (LFCC and mel-spectrogram) improved the results by 5% over standard front-ends composed of a single algorithm (refer to Section 3.5).

The speed and complexity of detection models are other problems in this field since they prevent the general public from verifying suspicious audio samples independently. Therefore, we focused on fast and reliable DeepFake detection methods. The related research is presented in Chapter 4 and is based on the manuscript *SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection*. There, we address Research Goal 2 — the proposed SpecRNet architecture, despite approximately 40% lower computational requirements, can achieve results comparable to similarly sized architectures. SpecRNet shows a marginal 0.001% decrease in AUC performance compared to the LCNN model. The performance of our model is further evaluated by three novel benchmarks focusing on the influence of the particular DF generators (cf. Section 4.5), short utterances (cf. Section 4.6) and data scarcity (cf. Section 4.7).

Specific manipulations, including adversarial attacks, can further hinder the DeepFake detection. We investigate the impact of adversarial attacks on DeepFake detectors and the means of defense against them. This research is described in Chapter 5 and is based on the paper *Defense Against Adversarial Attacks on Audio DeepFake Detection*. The chapter addresses Research Goal 3 and shows that adversarial attacks effectively degrade detectors' performance. The experiments show that the attacks decrease the performance by up to EER of 0.9905 for white-box attacks and up to EER of 0.4867 for transferability attacks — the results which make the models ineffectual (see Sections 5.5 and 5.6). To address this, we proposed a novel adaptive adversarial training (refer to Section 5.7), effectively improving robustness by decreasing an EER to 0.0982 for white-box and 0.1091 for transferability attacks.

Recently, there has been a significant improvement in many speech-processing tasks thanks to the use of architectures trained on large datasets. We decided to leverage tools provided by automatic speech recognition, a different speech processing task, to utilize Whisper for its significant dataset and use it as a feature extractor for DeepFake detection. We describe it in Chapter 6, based on the manuscript *Improved DeepFake Detection Using Whisper Features*. It covers enhancing the performance of DeepFake detection models by using front-ends based on other neural networks. The chapter addresses Research Goal 4: the use of information obtained from the encoder of the Whisper neural network (designed for the ASR task) can significantly increase the performance of models, leading to their increased generalization. The best of the proposed architectures improved the baseline results on the In-The-Wild dataset by 26% (see Section 6.8).

The research conducted in the dissertation not only enhanced the current state of audio DeepFake detection but also introduced ideas that can be successfully used in other fields of machine learning. The proposed *attack* *agnostic dataset* is a general framework (strategy) that can be used for the assessment and choice of the architecture that has the best generalization capabilities in anti-spoofing tasks. By preparing the dataset splits to cover many scenarios, e.g., grouping similar generation methods in testing subsets, one can assess the classifiers' performance in out-of-domain scenarios.

SpecRNet is a lightweight architecture that can quickly process large amounts of data without compromising much of the performance. The proposed solution shows an important trend of sharing technological innovations with the community: low model requirements increase the accessibility of DeepFake detection tools.

Our research on adversarial attacks was among the first works focused on attacking the DeepFake detectors. The reason for this research is that adversaries can intentionally degrade the quality of the samples under the guise of standard artifacts resulting from compression, etc. Contrary to the related anti-spoofing work, we focused on the fact that these samples primarily target human listeners. Moreover, our strategy of *adaptive adversarial training* can be successfully implemented in other classification tasks endangered by adversarial attacks.

Using features obtained from the Whisper ASR model in the tasks DeepFake detection, we were among the first to show that this seemingly unrelated architecture provided meaningful features beneficial in a vastly different problem. This approach is an alternative to the currently investigated state-of-the-art front-ends based on Self-Supervised Learning models. The improvements provided by using the Whisper model indicate that the other speech-processing architectures, preferably trained using large datasets, should be investigated in terms of feature extraction for the task of DeepFake detection.

Future work in DeepFake detection should focus primarily on addressing the most important challenges of the audio DeepFake detection field. The generalization of the audio DeepFake detectors, i.e. their performance on previously unseen types of recordings, is one of such tasks. Although current DeepFake detection methods achieve satisfying results on many datasets, this performance significantly degrades for *in-the-wild* samples [152]. This is out-of-distribution data that differs from one found in training datasets — it is created using newer synthesis models, altered in the course of attacks (using background noises or adversarial noises, compression), or modified unintentionally (standard compression resulting from upload to social media). The results provided using embedding- [9] or SSL-based [**?**] indicate that future work in this matter should focus on obtaining relevant features using such architectures.

Interpretability of detector results is another important aspect of future DeepFake detection research. Determining the exact information used by the detector for the classification allows for better interpretability of the results. Increasing the reliability of such technologies will enable them to be used in a court of law, where the DeepFake phenomenon may soon become an important issue. Careful analysis of classifier decisions will also help in improving the methods, making them more resistant to so-called shortcuts [220] — artifacts not directly related to the problem task at hand, which allow very good results on training and test sets leading to a poor generalization of models on out-of-distribution data [221].

To this date, the DeepFake datasets and related research have focused on English and Chinese languages. However, current voice synthesis methods support many other languages. Therefore, the scientific community's attention should also be directed to creating datasets and analyzing the impact of different languages on the issues of audio DeepFake detection. The first steps in this direction were made in our research [12], where we proposed the first multi-lingual DeepFake database. However, this issue requires not only support for additional languages and generation methods but also a thorough analysis of the relations between the synthesis methods in different languages.

In addition to developing DeepFake detection methods, other approaches should also be considered to counteract these technologies' malicious use effectively. Many of these utterances are created using easily accessible web services (cf. Section 2.1.3). In order to facilitate the detection of their content, the providers could employ audio watermarking techniques [222]. These methods embed additional information in the recordings at the expense of a slight, often unnoticeable, drop in the quality. During subsequent analysis, the information in the sample can provide intel about the system or users that created the particular DeepFake sample [223]. However, this approach does not solve the issue of DeepFakes — the attacks can target the watermarked recordings to remove or significantly degrade the embedded information [222]. Such attacks include various types of postprocessing, e.g. additive noises, compression, time shifting. In addition, watermarks can be applied only to samples created by the services that have implemented such functionality. When using open-source implementations containing watermarking, the malicious user can modify the source code to skip such a step.

Another crucial aspect of effectively combating DeepFakes is educating the public about their existence and its dangers. The primary motivation behind it is that the main goal of DeepFakes is to create samples recognized by people as spoken by a real person. Research presented in Section 2.3.1 shows that the ability to spot fake samples is significantly worse when people do not expect that the utterance may be generated — even when they are aware of the DeepFake phenomenon. Increasing awareness can instil scepticism in people about the source of the presented content — expectant listeners can notice the manipulation much more effectively. However, this solution also does not solve the issue for several reasons. First, not everyone may be informed about such manipulation methods despite the potential social actions. Moreover, emotional or controversial situations can negatively influence awareness. The problem of educating the public also does not solve the other vector of the DeepFake attack — bypassing ASV systems (see Section 2.3). Educating people about the issue of DeepFakes is also essential, as even with omnipresent detection algorithms, it is not helpful without personnel who can perform meaningful interpretations of the results.

As none of the presented ways to combat DeepFake, such as detection, watermarking, and educating society, can currently completely solve the presented problem, they should be considered complementary. Together, the methods represent a serious impediment to the misuse of speech synthesis and minimize the scale of the problem as much as possible.

Bibliography

[1] DHS. Inreasing Threat of DeepFake Identities. https://www.dhs.gov/sites/default/files/publications/increasing_threats_of_deepfake_identities_0.pdf, 2023. Accessed: 17.11.2023. [citation on page 1]

[2] Forbes. Fraudsters Cloned Company Director's Voice In $35 Million Bank Heist, Police Find. https://www.forbes.com/sites/thomasbrewster/2021/10/14/huge-bank-fraud-uses-deep-fake-voice-tech-to-steal-millions, 2021. Accessed: 17.11.2023. [citation on page 1, 17, 95]

[3] Avast. Voice fraud scams company out of 243,000 dollars. https://blog.avast.com/deepfake-voice-fraud-causes-243k-scam, 2019. Accessed: 17.11.2023. [citation on page 1, 17, 95]

[4] Vice. AI-Generated Voice Firm Clamps Down After 4chan Makes Celebrity Voices for Abuse. https://www.vice.com/en/article/dy7mww/ai-voice-firm-4chan-celebrity-voices-emma-watson-joe-rogan-elevenlabs, 2023. Accessed: 17.11.2023. [citation on page 1, 17, 95]

[5] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023. [citation on page 2, 77, 78, 79, 80, 92]

[6] Piotr Kawa, Marcin Plata, and Piotr Syga. Attack Agnostic Dataset: Towards Generalization and Stabilization of Audio DeepFake Detection. In *Proc. Interspeech 2022*, pages 4023–4027, 2022. [citation on page 2, 25, 29, 34, 37, 46, 58, 66]

[7] Piotr Kawa, Marcin Plata, and Piotr Syga. SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communi-*

cations (TrustCom), pages 792–799, 2022. [citation on page 2, 29, 35, 50, 51, 52, 61, 64, 66, 80]

[8] Piotr Kawa, Marcin Plata, and Piotr Syga. Defense Against Adversarial Attacks on Audio DeepFake Detection. In *Proc. INTERSPEECH 2023*, pages 5276–5280, 2023. [citation on page 3, 29, 35, 75]

[9] Piotr Kawa, Marcin Plata, Michał Czuba, Piotr Szymański, and Piotr Syga. Improved DeepFake Detection Using Whisper Features. In *Proc. INTERSPEECH 2023*, pages 4009–4013, 2023. [citation on page 3, 29, 35, 90, 97]

[10] Piotr Kawa and Piotr Syga. Verify It Yourself: A Note on Activation Functions' Influence on Fast DeepFake Detection. In *International Conference on Security and Cryptography*, 2021. [citation on page 3]

[11] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. MesoNet: a Compact Facial Video Forgery Detection Network. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7, 2018. [citation on page 3, 23, 40, 80]

[12] Nicolas M. Müller, Piotr Kawa, Wei Herng Choong, Edresson Casanova, Eren Gölge, Thorsten Müller, Piotr Syga, Philip Sperl, and Konstantin Böttinger. MLAAD: The Multi-Language Audio Anti-Spoofing Dataset. *arXiv preprint arXiv:2401.09512*, 2024. [citation on page 3, 30, 31, 79, 98]

[13] Nicolas M. Müller, Piotr Kawa, Shen Hu, Matthias Neu, Jennifer Williams, Philip Sperl, and Konstantin Böttinger. A New Approach to Voice Authenticity. *arXiv preprint arXiv:2402.06304*, 2024. [citation on page 4]

[14] Fact check: "Drunk" Nancy Pelosi video is manipulated | Reuters. `https://www.reuters.com/article/uk-factcheck-nancypelosi-manipulated-idUSKCN24Z2BI`. (Accessed: 23.08.2023). [citation on page 4]

[15] „Przesterowana" wypowiedź Rafała Trzaskowskiego w TVP. Celowy zabieg czy wypadek przy pracy? - YouTube. `https://www.youtube.com/watch?v=8pan0Ejrbls`. (Accessed on 02/05/2024). [citation on page 4]

[16] Fox News edits video of Biden to make it seem he was being racially insensitive | Fox News | The Guardian. `https://www.theguardian.com/media/2021/nov/13/fox-news-edits-biden-video-negro-leagues-satchel-paige`. (Accessed on 02/06/2024). [citation on page 4]

[17] The M-AILABS Speech Dataset. The M-AILABS Speech Dataset. `https://www.caito.de/2019/01/03/the-m-ailabs-speech-dataset/`, 2023. Accessed on 01/02/2024. [citation on page 4]

[18] James L. Flanagan. *Speech Analysis, Synthesis, and Perception.* Springer-Verlag, Berlin, Heidelberg, 1972. [citation on page 7]

[19] Jacob Benesty, M. Mohan Sondhi, and Yiteng (Arden) Huang. *Springer Handbook of Speech Processing.* Springer-Verlag, Berlin, Heidelberg, 2007. [citation on page 7, 12]

[20] Stanley Kubrick. 2001: A Space Odyssey, 1968. [citation on page 7]

[21] Deng, Li and O'Shaughnessy, Douglas. *Speech processing: a dynamic and optimization-oriented approach.* CRC Press, 2003. [citation on page 7]

[22] IEEE Spectrum. The Consumer Electronics Hall of Fame: Texas Instruments' Speak And Spell, 2023. `https://spectrum.ieee.org/the-consumer-electronics-hall-of-fame-texas-instruments-speak-spell`, Accessed: 30.06.2023. [citation on page 7, 9]

[23] The New York Times. Ann Syrdal, Who Helped Give Computers a Female Voice, Dies at 74, 2023. `https://www.nytimes.com/2020/08/20/technology/ann-syrdal-who-helped-give-computers-a-female-voice-dies-at-74.html`, Accessed: 30.06.2023. [citation on page 8]

[24] Microsoft. Complete guide to Narrator, 2023. `https://support.microsoft.com/en-us/windows/complete-guide-to-narrator-e4397a0d-ef4f-b386-d8ae-c172f109bdb1`, Accessed: 30.06.2023. [citation on page 8]

[25] Amazon. Amazon.com Announces Acquisition of IVONA Software, 2023. `https://press.aboutamazon.com/2013/1/amazon-com-announces-acquisition-of-ivona-software`, Accessed: 30.06.2023. [citation on page 8]

[26] www.sztucznainteligencja.org.pl. Ivona, Alexa, Vika or intelligent girls from Gdansk, 2023. `https://www.sztucznainteligencja.org.pl/en/ivona-alexa-vika-or-intelligent-girls-from-gdansk/`, Accessed: 30.06.2023. [citation on page 8]

[27] Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*, 2021. [citation on page 8]

[28] Xu Tan. *Neural Text-to-Speech Synthesis.* Springer Nature, 2023. [citation on page 8, 16]

[29] Fadi Biadsy, Ron J. Weiss, Pedro J. Moreno, Dimitri Kanvesky, and Ye Jia. Parrotron: An End-to-End Speech-to-Speech Conversion Model and its Applications to Hearing-Impaired Speech and Speech Separation. In *Proc. Interspeech 2019*, pages 4115–4119, 2019. [citation on page 8, 95]

[30] Wired. Audiobooks synthetic voices. `https://www.wired.com/story/audiobooks-synthetic-voices/`. Accessed: 21.01.2024. [citation on page 8]

[31] Apple. Siri Product Website. `https://www.apple.com/siri/`, 2023. Accessed: 17.11.2023. [citation on page 8, 95]

[32] Amazon. Alexa Developer Website. `https://developer.amazon.com/en-US/alexa`, 2023. Accessed: 17.11.2023. [citation on page 9, 95]

[33] Google. Google Assistant Website. `https://assistant.google.com/`, 2023. Accessed: 17.11.2023. [citation on page 9, 95]

[34] Google. Google Maps. `https://www.google.pl/maps`. (Accessed on 02/05/2024). [citation on page 9]

[35] Apple. Apple Maps. `https://www.apple.com/maps/`. (Accessed on 02/05/2024). [citation on page 9]

[36] Speechify. Speechify Education Offer, 2023. `https://speechify.com/edu/`, Accessed: 15.10.2023. [citation on page 9]

[37] Shu-Chuan Tseng, Yi-Fen Liu, and Xiang-Li Lu. Model-assisted lexical tone evaluation of three-year-old chinese-speaking children by also considering segment production. In *Proc. INTERSPEECH 2023*, pages 3909–3913, 2023. [citation on page 9]

[38] Coqui.ai. Coqui.ai, 2023. `https://coqui.ai/`, Accessed: 30.06.2023. [citation on page 9, 10, 32]

[39] Resemble.ai. Generative Voice AI built for Enterprise, 2023. `https://www.resemble.ai/`, Accessed: 30.06.2023. [citation on page 9]

[40] ElevenLabs. Generative voice ai, 2023. `https://elevenlabs.io/`, Accessed: 30.06.2023. [citation on page 9, 47]

[41] Respeecher. Voice Cloning for Content Creators, 2023. `https://www.respeecher.com/`, Accessed: 30.06.2023. [citation on page 9]

[42] Amazon. Amazon Polly. `https://aws.amazon.com/polly/`. Accessed: 21.01.2024. [citation on page 9, 29]

[43] Baidu AI Cloud. Speech technology. `https://intl.cloud.baidu.com/product/speech.html`. Accessed: 25.06.2023. [citation on page 9, 29, 30]

[44] Google. Text-to-Speech AI, 2023. `https://cloud.google.com/text-to-speech`, Accessed: 30.06.2023. [citation on page 9, 29]

[45] Azure Microsoft. Text to speech. `https://azure.microsoft.com/en-en/products/ai-services/text-to-speech`. Accessed: 21.01.2024. [citation on page 9, 29]

[46] Coqui.ai. Coqui-TTS, 2023. `https://github.com/coqui-ai/TTS`, Accessed: 30.06.2023. [citation on page 9]

[47] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211, 2018. [citation on page 9]

[48] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-Art Natural Language Processing. pages 38–45. Association for Computational Linguistics, October 2020. [citation on page 9, 32]

[49] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio. SpeechBrain. `https://github.com/speechbrain/speechbrain/`. [citation on page 9]

[50] Coqui.ai. Xtts, 2023. `https://coqui.ai/blog/tts/xtts_taking_tts_to_the_next_level/`, Accessed: 30.06.2023. [citation on page 9, 10, 11]

[51] Sercan Ö Arık, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, et al. Deep voice: Real-time neural text-to-speech. In *International conference on machine learning*, pages 195–204. PMLR, 2017. [citation on page 10]

[52] Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Z. Chen, Samy Bengio, Quoc V. Le, Yannis Agiomyrgiannakis, Robert A. J. Clark, and Rif A. Saurous. Tacotron: Towards End-to-End Speech Synthesis. In *Interspeech*, 2017. [citation on page 10, 30]

[53] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4779–4783. IEEE, 2018. [citation on page 10, 44]

[54] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. *FastSpeech: Fast, Robust and Controllable Text to Speech.* 2019. [citation on page 10, 30]

[55] Jan Vainer and Ondřej Dušek. SpeedySpeech: Efficient Neural Speech Synthesis. In *Proc. Interspeech 2020*, pages 3575–3579, 2020. [citation on page 10]

[56] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621, 2019. [citation on page 10, 27, 31]

[57] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33:17022–17033, 2020. [citation on page 10, 27, 31, 44]

[58] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. In *Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*, page 125, 2016. [citation on page 10, 31]

[59] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *International Conference on Machine Learning*, pages 2410–2419. PMLR, 2018. [citation on page 10, 44]

[60] Kundan Kumar, Rithesh Kumar, Thibault De Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and

Aaron C Courville. Melgan: Generative adversarial networks for conditional waveform synthesis. *Advances in neural information processing systems*, 32, 2019. [citation on page 10, 27, 31]

[61] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6199–6203. IEEE, 2020. [citation on page 10, 27, 31, 56]

[62] Geng Yang, Shan Yang, Kai Liu, Peng Fang, Wei Chen, and Lei Xie. Multi-band melgan: Faster waveform generation for high-quality text-to-speech. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 492–498. IEEE, 2021. [citation on page 10, 27, 31, 39]

[63] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021. [citation on page 10, 11]

[64] Adrian Łańcucki. Fastpitch: Parallel text-to-speech with pitch prediction. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6588–6592. IEEE, 2021. [citation on page 10]

[65] James Betker. Better speech synthesis through scaling. *arXiv preprint arXiv:2305.07243*, 2023. [citation on page 10]

[66] Eric Battenberg, Soroosh Mariooryad, Daisy Stanton, RJ Skerry-Ryan, Matt Shannon, David Kao, and Tom Bagby. Effective Use of Variational Embedding Capacity in Expressive End-to-End Speech Synthesis, 2019. [citation on page 10]

[67] Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*, pages 2709–2720. PMLR, 2022. [citation on page 10]

[68] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Advances in Neural Information Processing Systems*, 33:8067–8077, 2020. [citation on page 10]

[69] Shivam Mehta, Ambika Kirkland, Harm Lameris, Jonas Beskow, Éva Székely, and Gustav Eje Henter. OverFlow: Putting flows on top of neural transducers for better TTS. In *Proc. INTERSPEECH 2023*, pages 4279–4283, 2023. [citation on page 10]

[70] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo. Stargan-vc: Non-parallel many-to-many voice conversion using star generative adversarial networks. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 266–273. IEEE, 2018. [citation on page 11]

[71] Takuhiro Kaneko and Hirokazu Kameoka. Parallel-data-free voice conversion using cycle-consistent adversarial networks. *arXiv preprint arXiv:1711.11293*, 2017. [citation on page 11]

[72] Jingyi Li, Weiping Tu, and Li Xiao. Freevc: Towards High-Quality Text-Free One-Shot Voice Conversion. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. [citation on page 11]

[73] Anastasia Natsiou and Seán O'Leary. Audio representations for deep learning in sound synthesis: A review. *2021 IEEE/ACS 18th International Conference on Computer Systems and Applications (AICCSA)*, pages 1–8, 2021. [citation on page 11]

[74] Tom Bäckström, Okko Räsänen, Abraham Zewoudie, Pablo Pérez Zarazaga, Liisa Koivusalo, Sneha Das, Esteban Gómez Mellado, Marieum Bouafif Mansali, Daniel Ramos, Sudarsana Kadiri, and Paavo Alku. *Introduction to Speech Processing*. 2 edition, 2022. [citation on page 11, 12, 14, 81]

[75] Keith Ito and Linda Johnson. The LJ Speech Dataset. `https://keithito.com/LJ-Speech-Dataset/`, 2017. [citation on page 11, 13, 14, 29]

[76] Md. Sahidullah and Goutam Saha. Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. *Speech Communication*, 54(4):543–565, 2012. [citation on page 13, 14]

[77] Fang Zheng, Guoliang Zhang, and Zhanjiang Song. Comparison of different implementations of mfcc. *Journal of Computer science and Technology*, 16:582–589, 2001. [citation on page 13]

[78] Xinhui Zhou, Daniel Garcia-Romero, Ramani Duraiswami, Carol Espy-Wilson, and Shihab Shamma. Linear versus mel frequency cepstral coefficients for speaker recognition. In *2011 IEEE workshop on automatic speech recognition & understanding*, pages 559–564. IEEE, 2011. [citation on page 14]

[79] Sita Purnama Dewi, Anggunmeka Luhur Prasasti, and Budhi Irawan. The study of baby crying analysis using mfcc and lfcc in different classification methods. In *2019 IEEE International Conference on Signals and Systems (ICSigSys)*, pages 18–23, 2019. [citation on page 14]

[80] Taufiq Hasan, Omid Sadjadi, Gang Liu, Navid Shokouhi, Hynek Boril, and John Hanse. Crss systems for 2012 nist speaker recognition evaluation. 05 2013. [citation on page 14]

[81] Massimiliano Todisco, Héctor Delgado, and Nicholas Evans. Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45:516–535, 2017. [citation on page 14, 80]

[82] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`. [citation on page 14]

[83] Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *ACM Computing Surveys (CSUR)*, 54(1):1–41, 2021. [citation on page 15]

[84] Gan Pei, Jiangning Zhang, Menghan Hu, Zhenyu Zhang, Chengjie Wang, Yunsheng Wu, Guangtao Zhai, Jian Yang, Chunhua Shen, and Dacheng Tao. Deepfake Generation and Detection: A Benchmark and Survey, 2024. [citation on page 15]

[85] deepfakes. faceswap: GitHub repository. `https://github.com/deepfakes/faceswap`. (Accessed on 02/05/2024). [citation on page 15]

[86] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. [citation on page 15]

[87] iperov. DeepFaceLab: GitHub repository. `https://github.com/iperov/DeepFaceLab`. (Accessed on 02/05/2024). [citation on page 15]

[88] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. [citation on page 15]

[89] shaoanlu. faceswap-GAN: GitHub repository. `https://github.com/shaoanlu/faceswap-GAN`. (Accessed on 02/05/2024). [citation on page 15]

[90] Yuval Nirkin, Yosi Keller, and Tal Hassner. Fsgan: Subject agnostic face swapping and reenactment. In *Proceedings of the IEEE/CVF international*

conference on computer vision, pages 7184–7193, 2019. [citation on page 15, 27]

[91] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. [citation on page 15]

[92] Joel Ruben Antony Moniz, Christopher Beckham, Simon Rajotte, Sina Honari, and Chris Pal. Unsupervised depth estimation, 3d face rotation and replacement. *Advances in neural information processing systems*, 31, 2018. [citation on page 15]

[93] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. [citation on page 15]

[94] Yue Han, Junwei Zhu, Keke He, Xu Chen, Yanhao Ge, Wei Li, Xiangtai Li, Jiangning Zhang, Chengjie Wang, and Yong Liu. Face Adapter for Pre-Trained Diffusion Models with Fine-Grained ID and Attribute Control, 2024. [citation on page 15]

[95] Kihong Kim, Yunho Kim, Seokju Cho, Junyoung Seo, Jisu Nam, Kychul Lee, Seungryong Kim, and KwangHee Lee. Diffface: Diffusion-based face swapping with facial guidance. *arXiv preprint arXiv:2212.13344*, 2022. [citation on page 15]

[96] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2Face: Real-time Face Capture and Reenactment of RGB Videos. *Communications of the ACM*, 2018. [citation on page 15]

[97] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019. [citation on page 15]

[98] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9459–9468, 2019. [citation on page 15]

[99] Stella Bounareli, Christos Tzelepis, Vasileios Argyriou, Ioannis Patras, and Georgios Tzimiropoulos. HyperReenact: One-Shot Reenactment via Jointly Learning to Refine and Retarget Faces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [citation on page 15]

[100] Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Thien Huynh-The, Saeid Nahavandi, Thanh Tam Nguyen, Quoc-Viet Pham, and Cuong M. Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, 2022. [citation on page 15]

[101] Choi, Yunjey and Uh, Youngjung and Yoo, Jaejun and Ha, Jung-Woo. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8188–8197, 2020. [citation on page 15]

[102] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. [citation on page 15]

[103] Jianxin Sun, Qiyao Deng, Qi Li, Muyi Sun, Min Ren, and Zhenan Sun. Any-Face: Free-style text-to-face synthesis and manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18687–18696, 2022. [citation on page 15]

[104] Apple. Apple introduces new features for cognitive accessibility, along with Live Speech, Personal Voice, and Point and Speak in Magnifier. `https://www.apple.com/newsroom/2023/05/apple-previews-live-speech-personal-voice-and-more-new-accessibility-features/`, 2023. Accessed: 17.11.2023. [citation on page 16]

[105] Zhang, Wenxuan and Cun, Xiaodong and Wang, Xuan and Zhang, Yong and Shen, Xi and Guo, Yu and Shan, Ying and Wang, Fei. SadTalker: Learning Realistic 3D Motion Coefficients for Stylized Audio-Driven Single Image Talking Face Animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8652–8661, June 2023. [citation on page 16]

[106] Stypułkowski, Michał and Vougioukas, Konstantinos and He, Sen and Zięba, Maciej and Petridis, Stavros and Pantic, Maja. Diffused Heads: Diffusion Models Beat GANs on Talking-Face Generation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5091–5100, January 2024. [citation on page 16]

[107] HeyGen. AI-powered video creation at scale, 2024. `https://www.heygen.com/`, Accessed: 30.01.2024. [citation on page 16]

[108] Vice. European politicians duped into deepfake video calls with mayor of Kyiv, 2023. `https://www.theguardian.com/world/2022/jun/25/`

european-leaders-deepfake-video-calls-mayor-of-kyiv-vitali-klitschko, Accessed: 30.06.2023. [citation on page 16, 95]

[109] Vice. How I Broke Into a Bank Account With an AI-Generated Voice, 2023. https://www.vice.com/en/article/dy7axa/how-i-broke-into-a-bank-account-with-an-ai-generated-voice, Accessed: 30.06.2023. [citation on page 16, 95]

[110] Nicolas M. Müller, Karla Pizzi, and Jennifer Williams. Human perception of audio deepfakes. In *Proceedings of the 1st International Workshop on Deepfake Detection for Audio Multimedia*, pages 85–91, 2022. [citation on page 17, 18]

[111] Massimiliano Todisco, Xin Wang, Ville Vestman, Md. Sahidullah, Héctor Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi H. Kinnunen, and Kong Aik Lee. ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection. In *Proc. Interspeech 2019*, pages 1008–1012, 2019. [citation on page 17, 22, 25, 26, 30, 31, 38, 61, 62, 87]

[112] Daniel Prudký, Anton Firc, and Kamil Malinka. Assessing the human ability to recognize synthetic speech in ordinary conversation. In *2023 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–5, 2023. [citation on page 18]

[113] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1–11, 2019. [citation on page 19]

[114] Anil Jain, Arun Ross, and Karthik Nandakumar. *Introduction to Biometrics.* 01 2011. [citation on page 20]

[115] Serife Kucur Ergünay, Elie Khoury, Alexandros Lazaridis, and Sébastien Marcel. On the vulnerability of speaker verification to realistic voice spoofing. In *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–6. IEEE, 2015. [citation on page 22]

[116] Tomi Kinnunen, Md. Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. The ASVspoof 2017 Challenge: Assessing the Limits of Replay Spoofing Attack Detection. In *Proc. Interspeech 2017*, pages 2–6, 2017. [citation on page 22, 25, 30]

[117] Pavel Korshunov, André R Gonçalves, Ricardo PV Violato, Flávio O Simões, and Sébastien Marcel. On the use of convolutional neural networks for speech presentation attack detection. In *2018 IEEE 4th international*

conference on identity, security, and behavior analysis (ISBA), pages 1–8. IEEE, 2018. [citation on page 22]

[118] Junichi Yamagishi, Xin Wang, Massimiliano Todisco, Md Sahidullah, Jose Patino, Andreas Nautsch, Xuechen Liu, Kong Aik Lee, Tomi Kinnunen, Nicholas Evans, and Héctor Delgado. ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection. In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 47–54, 2021. [citation on page 22, 25, 30, 40, 61, 66, 82]

[119] Zhizheng Wu, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi, Cemal Hanilçi, and Md Sahidullah. ASVspoof 2015: the First Automatic Speaker Verification Spoofing and Countermeasures Challenge. 09 2015. [citation on page 22, 25, 30]

[120] Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009. [citation on page 22]

[121] Bhusan Chettri and Bob Sturm. A deeper look at gaussian mixture model based anti-spoofing systems. pages 5159–5163, 04 2018. [citation on page 23]

[122] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. [citation on page 23]

[123] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A Light CNN for Deep Face Representation With Noisy Labels. *IEEE Transactions on Information Forensics and Security*, 13(11):2884–2896, 2018. [citation on page 23, 24, 40, 41, 48, 49, 64]

[124] Xin Wang and Junichi Yamagishi. A Comparative Study on Recent Neural Spoofing Countermeasures for Synthetic Speech Detection. In *Proc. Interspeech 2021*, pages 4259–4263, 2021. [citation on page 23, 61, 62, 80]

[125] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013. [citation on page 23]

[126] Jee-weon Jung, Seung-bin Kim, Hye-jin Shim, Ju-ho Kim, and Ha-Jin Yu. Improved RawNet with Feature Map Scaling for Text-Independent Speaker Verification Using Raw Waveforms. *Proc. Interspeech 2020*, pages 1496–1500, 2020. [citation on page 24, 48, 49, 50, 64]

[127] Hemlata Tak, Jose Patino, Massimiliano Todisco, Andreas Nautsch, Nicholas Evans, and Anthony Larcher. End-to-End anti-spoofing with RawNet2. *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun 2021. [citation on page 24, 40, 41, 49, 64, 66, 82, 87]

[128] Mirco Ravanelli and Yoshua Bengio. Speaker recognition from raw waveform with sincnet. In *2018 IEEE spoken language technology workshop (SLT)*, pages 1021–1028. IEEE, 2018. [citation on page 24]

[129] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, USA, 1997. [citation on page 24]

[130] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. [citation on page 24, 89]

[131] Hemlata Tak, Jee weon Jung, Jose Patino, Madhu Kamble, Massimiliano Todisco, and Nicholas Evans. End-to-end spectro-temporal graph attention networks for speaker verification anti-spoofing and speech deepfake detection. In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 1–8, 2021. [citation on page 24]

[132] Kaavya Sriskandaraja, Vidhyasaharan Sethu, Phu Ngoc Le, and Eliathamby Ambikairajah. Investigation of Sub-Band Discriminative Information Between Spoofed and Genuine Speech. In *Proc. Interspeech 2016*, pages 1710–1714, 2016. [citation on page 24]

[133] Marcin Witkowski, Stanisław Kacprzak, Piotr Żelasko, Konrad Kowalczyk, and Jakub Gałka. Audio Replay Attack Detection Using High-Frequency Features. In *Proc. Interspeech 2017*, pages 27–31, 2017. [citation on page 24]

[134] Jichen Yang, Rohan Kumar Das, and Haizhou Li. Significance of subband features for synthetic speech detection. *IEEE Transactions on Information Forensics and Security*, 15:2160–2170, 2020. [citation on page 24]

[135] Jee-weon Jung, Hee-Soo Heo, Hemlata Tak, Hye-jin Shim, Joon Son Chung, Bong-Jin Lee, Ha-Jin Yu, and Nicholas Evans. Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6367–6371. IEEE, 2022. [citation on page 24]

[136] Hemlata Tak, Massimiliano Todisco, Xin Wang, Jee weon Jung, Junichi Yamagishi, and Nicholas Evans. Automatic Speaker Verification Spoofing and Deepfake Detection Using Wav2vec 2.0 and Data Augmentation. In

Proc. The Speaker and Language Recognition Workshop (Odyssey 2022), pages 112–119, 2022. [citation on page 24]

[137] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc., 2020. [citation on page 24, 77, 93]

[138] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 2022. [citation on page 25, 77]

[139] Hemlata Tak, Madhu Kamble, Jose Patino, Massimiliano Todisco, and Nicholas Evans. Rawboost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6382–6386, 2022. [citation on page 25, 82, 93]

[140] Nicholas Evans, Tomi Kinnunen, and Junichi Yamagishi. Spoofing and countermeasures for automatic speaker verification. In *Proc. Interspeech 2013*, pages 925–929, 2013. [citation on page 25]

[141] Christophe Veaux, Junichi Yamagishi, and Kirsten MacDonald. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. 2017. [citation on page 26]

[142] Jaime Lorenzo-Trueba, Junichi Yamagishi, Tomoki Toda, Daisuke Saito, Fernando Villavicencio, Tomi Kinnunen, and Zhenhua Ling. The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods. *The Speaker and Language Recognition Workshop (Odyssey 2018)*, Jun 2018. [citation on page 27]

[143] Zhao Yi, Wen-Chin Huang, Xiaohai Tian, Junichi Yamagishi, Rohan Das, Tomi Kinnunen, Zhen-Hua Ling, and Tomoki Toda. Voice conversion challenge 2020 - intra-lingual semi-parallel and cross-lingual voice conversion. pages 80–98, 10 2020. [citation on page 27]

[144] Xuechen Liu, Xin Wang, Md Sahidullah, Jose Patino, Héctor Delgado, Tomi Kinnunen, Massimiliano Todisco, Junichi Yamagishi, Nicholas Evans, Andreas Nautsch, and Kong Aik Lee. Asvspoof 2021: Towards spoofed and

deepfake speech detection in the wild. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2507–2522, 2023. [citation on page 27]

[145] Joel Frank and Lea Schönherr. WaveFake: A Data Set to Facilitate Audio Deepfake Detection. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. [citation on page 27, 30, 38, 40, 41, 55, 58, 60, 61, 66, 82]

[146] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040, 2020. [citation on page 27]

[147] Hasam Khalid, Shahroz Tariq, Minha Kim, and Simon S. Woo. FakeAVCeleb: A Novel Audio-Video Multimodal Deepfake Dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. [citation on page 27, 28, 30, 38, 41, 66]

[148] Iryna Korshunova, Wenzhe Shi, Joni Dambre, and Lucas Theis. Fast face-swap using convolutional neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 3677–3685, 2017. [citation on page 27]

[149] KR Prajwal, Rudrabha Mukhopadhyay, Vinay P Namboodiri, and CV Jawahar. A lip sync expert is all you need for speech to lip generation in the wild. In *Proceedings of the 28th ACM international conference on multimedia*, pages 484–492, 2020. [citation on page 27]

[150] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems*, 31, 2018. [citation on page 27, 45]

[151] IBM. IBM Watson - speech to text. `https://www.ibm.com/cloud/watson-speech-to-text`. Accessed: 17.11.2023. [citation on page 27]

[152] Nicolas Müller, Pavel Czempin, Franziska Diekmann, Adam Froghyar, and Konstantin Böttinger. Does Audio Deepfake Detection Generalize? In *Proc. Interspeech 2022*, pages 2783–2787, 2022. [citation on page 28, 30, 66, 77, 83, 84, 87, 97]

[153] Jiangyan Yi, Ruibo Fu, Jianhua Tao, Shuai Nie, Haoxin Ma, Chenglong Wang, Tao Wang, Zhengkun Tian, Ye Bai, Cunhang Fan, et al. Add 2022:

the first audio deep synthesis detection challenge. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9216–9220. IEEE, 2022. [citation on page 29, 30]

[154] Jiangyan Yi, Jianhua Tao, Ruibo Fu, Xinrui Yan, Chenglong Wang, Tao Wang, Chu Yuan Zhang, Xiaohui Zhang, Yan Zhao, Yong Ren, et al. Add 2023: the second audio deepfake detection challenge. *IJCAI 2023 Workshop on Deepfake Audio Detection (DADA 2023)*, 2023. [citation on page 29, 30]

[155] Ricardo Reimao and Vassilios Tzerpos. For: A dataset for synthetic speech detection. In *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pages 1–10. IEEE, 2019. [citation on page 29, 30]

[156] John Kominek and Alan Black. The cmu arctic speech databases. *SSW5-2004*, 01 2004. [citation on page 29]

[157] Voxforge.org. Free Speech Recognition (Linux, Windows and Mac) - voxforge.org. `http://www.voxforge.org/`. Accessed: 25.06.2023. [citation on page 29]

[158] Zhenyu Zhang, Yewei Gu, Xiaowei Yi, and Xianfeng Zhao. Fmfcc-a: a challenging mandarin dataset for synthetic speech detection. In *International Workshop on Digital Watermarking*, pages 117–131. Springer, 2021. [citation on page 30]

[159] IBM. IBM Watson - speech to text. `https://www.ibm.com/products/text-to-speech`. Accessed: 17.11.2023. [citation on page 30]

[160] Yewei Gu, Zhenyu Zhang, Xiaowei Yi, and Xianfeng Zhao. Mediumvc: Any-to-any voice conversion using synthetic specific-speaker speeches as intermedium features. *arXiv preprint arXiv:2110.02500*, 2021. [citation on page 30]

[161] Jiangyan Yi, Ye Bai, Jianhua Tao, Haoxin Ma, Zhengkun Tian, Chenglong Wang, Tao Wang, and Ruibo Fu. Half-Truth: A Partially Fake Audio Detection Dataset. In *Proc. Interspeech 2021*, pages 1654–1658, 2021. [citation on page 30]

[162] Haoxin Ma, Jiangyan Yi, Chenglong Wang, Xinrui Yan, Jianhua Tao, Tao Wang, Shiming Wang, Le Xu, and Ruibo Fu. Fad: A chinese dataset for fake audio detection. *arXiv preprint arXiv:2207.12308*, 2022. [citation on page 30, 31]

[163] Lin Zhang, Xin Wang, Erica Cooper, Nicholas Evans, and Junichi Yamagishi. The partialspoof database and countermeasures for the detection of

short fake speech segments embedded in an utterance. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:813–825, 2023. [citation on page 30, 31]

[164] Xin Wang and Junichi Yamagishi. Spoofed training data for speech spoofing countermeasure can be efficiently created using neural vocoders. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. [citation on page 30, 31, 93]

[165] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ-Skerry Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A Saurous. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *International conference on machine learning*, pages 5180–5189. PMLR, 2018. [citation on page 30]

[166] Jean-Marc Valin and Jan Skoglund. Lpcnet: Improving neural speech synthesis through linear prediction. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5891–5895. IEEE, 2019. [citation on page 31]

[167] Ahmed Mustafa, Nicola Pia, and Guillaume Fuchs. Stylemelgan: An efficient high-fidelity adversarial vocoder with temporal adaptive normalization. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038. IEEE, 2021. [citation on page 31]

[168] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014. [citation on page 32, 71]

[169] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018. [citation on page 32, 62]

[170] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE security and privacy workshops (SPW)*, pages 1–7. IEEE, 2018. [citation on page 33]

[171] Shashank Kotyan and Danilo Vasconcellos Vargas. Adversarial Robustness Assessment: Why both l_0 and l_∞ Attacks Are Necessary, 2019. [citation on page 33]

[172] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR*, 2015. [citation on page 33, 63]

[173] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*, 2018. [citation on page 34, 63, 89]

[174] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020. [citation on page 34, 63]

[175] Radosław Białobrzeski, Michał Kośmider, Mateusz Matuszewski, Marcin Plata, and Alexander Rakowski. Robust Bayesian and Light Neural Networks for Voice Spoofing Detection. In *Proc. Interspeech 2019*, pages 1028–1032, 2019. [citation on page 38]

[176] Md Sahidullah, Tomi Kinnunen, and Cemal Hanilçi. A comparison of features for synthetic speech detection. ISCA (the International Speech Communication Association), 09 2015. [citation on page 40, 41, 80]

[177] Francois Chollet. Xception: Deep Learning With Depthwise Separable Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. [citation on page 40, 41]

[178] Diederik P. Kingma andJimmy Ba. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [citation on page 41, 51]

[179] Heiga Zen, Andrew Senior, and Mike Schuster. Statistical parametric speech synthesis using deep neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7962–7966, 2013. [citation on page 44]

[180] Masanori MORISE, Fumiya YOKOMORI, and Kenji Ozawa. World: A vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, E99.D:1877–1884, 07 2016. [citation on page 44]

[181] Yannis Agiomyrgiannakis. Vocaine the vocoder and applications in speech synthesis. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4230–4234, 2015. [citation on page 44]

[182] Jason Wise. How Many Videos Are Uploaded To Youtube A Day in 2022?, 2022. `https://earthweb.com/how-many-videos-are-uploaded-to-youtube-a-day/`, Accessed: 30.06.2023. [citation on page 47]

[183] Esma Aïmeur, Sabrine Amri, and Gilles Brassard. Fake news, disinformation and misinformation in social media: a review. *Social Network Analysis and Mining*, 13(1):30, 2023. [citation on page 47]

[184] Stability.ai. Stability.ai Website. `https://stability.ai/`. Accessed: 21.01.2024. [citation on page 47]

[185] OpenAI. ChatGPT. `https://openai.com/chatgpt`. Accessed: 21.01.2024. [citation on page 47]

[186] European Commission. 2022 Strengthened Code of Practice on Disinformation, 2022. [citation on page 47]

[187] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. [citation on page 48]

[188] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. [citation on page 48]

[189] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. [citation on page 49]

[190] Songxiang Liu, Haibin Wu, Hung-Yi Lee, and Helen Meng. Adversarial Attacks on Spoofing Countermeasures of Automatic Speaker Verification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 312–319, 2019. [citation on page 61]

[191] Haibin Wu, Songxiang Liu, Helen Meng, and Hung-yi Lee. Defense Against Adversarial Attacks on Spoofing Countermeasures of ASV. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6564–6568, 2020. [citation on page 61]

[192] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial Attacks Against Automatic Speech Recognition Systems via Psychoacoustic Hiding. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019.* The Internet Society, 2019. [citation on page 61]

[193] Hao Tan, Junjian Zhang, Huan Zhang, Le Wang, Yaguan Qian, and Zhaoquan Gu. NRI-FGSM: An Efficient Transferable Adversarial Attack for Speaker Recognition Systems. In *Proc. Interspeech 2022*, pages 4386–4390, 2022. [citation on page 61]

[194] Sonal Joshi, Saurabh Kataria, Yiwen Shao, Piotr Żelasko, Jesús Villalba, Sanjeev Khudanpur, and Najim Dehak. Defense against Adversarial Attacks on Hybrid Speech Recognition System using Adversarial Finetuning with Denoiser. In *Proc. Interspeech 2022*, pages 5035–5039, 2022. [citation on page 61, 66, 72]

[195] Yiwen Shao, Jesus Villalba, Sonal Joshi, Saurabh Kataria, Sanjeev Khudanpur, and Najim Dehak. Chunking Defense for Adversarial Attacks on ASR. In *Proc. Interspeech 2022*, pages 5045–5049, 2022. [citation on page 61]

[196] Jee weon Jung, Youjin Kim, Hee-Soo Heo, Bong-Jin Lee, Youngki Kwon, and Joon Son Chung. Pushing the limits of raw waveform speaker recognition. In *Proc. Interspeech 2022*, pages 2228–2232, 2022. [citation on page 61, 64, 80]

[197] R. Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, volume 1, pages 125–128 vol.1, 1993. [citation on page 63]

[198] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*, 2017. [citation on page 66]

[199] Kaichao You, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. How Does Learning Rate Decay Help Modern Neural Networks?, 2019. [citation on page 67]

[200] Ayse Elvan Aydemir, Alptekin Temizel, and Tugba Taskaya Temizel. The effects of JPEG and JPEG2000 compression on attacks using adversarial examples. *arXiv preprint arXiv:1803.10418*, 2018. [citation on page 71]

[201] Kui Ren, Tianhang Zheng, Zhan Qin, and Xue Liu. Adversarial Attacks and Defenses in Deep Learning. *Engineering*, 6(3):346–360, 2020. [citation on page 71]

[202] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent Advances in Adversarial Training for Adversarial Robustness. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4312–4321, 8 2021. [citation on page 71]

[203] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble Adversarial Training: Attacks and Defenses. In *International Conference on Learning Representations*, 2018. [citation on page 71]

[204] Yogesh Balaji, Tom Goldstein, and Judy Hoffman. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*, 2019. [citation on page 71]

[205] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent Advances in Adversarial Training for Adversarial Robustness. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4312–4321, 8 2021. [citation on page 71]

[206] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A Cookbook of Self-Supervised Learning, 2023. [citation on page 77]

[207] Shuo Liu, Adria Mallol-Ragolta, Emilia Parada-Cabaleiro, Kun Qian, Xin Jing, Alexander Kathan, Bin Hu, and Bjoern W Schuller. Audio self-supervised learning: A survey. *Patterns*, 3(12), 2022. [citation on page 77]

[208] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022. [citation on page 77]

[209] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised Pre-training for Speech Recognition, 2019. [citation on page 77, 79]

[210] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021. [citation on page 77]

[211] Hemlata Tak, Massimiliano Todisco, Xin Wang, Jee weon Jung, Junichi Yamagishi, and Nicholas Evans. Automatic Speaker Verification Spoofing and Deepfake Detection Using Wav2vec 2.0 and Data Augmentation. In

Proc. The Speaker and Language Recognition Workshop (Odyssey 2022), pages 112–119, 2022. [citation on page 77]

[212] Xin Wang and Junichi Yamagishi. Investigating Self-Supervised Front Ends for Speech Spoofing Countermeasures. In *Proc. The Speaker and Language Recognition Workshop (Odyssey 2022)*, pages 100–106, 2022. [citation on page 77, 80]

[213] OpenAI. Introducing Whisper, 2022. `https://openai.com/research/whisper`, Accessed: 30.03.2024. [citation on page 79]

[214] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. [citation on page 78, 92]

[215] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. [citation on page 78]

[216] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. [citation on page 78]

[217] Tomi Kinnunen, Héctor Delgado, Nicholas Evans, Kong Aik Lee, Ville Vestman, Andreas Nautsch, Massimiliano Todisco, Xin Wang, Md Sahidullah, Junichi Yamagishi, and Douglas A. Reynolds. Tandem Assessment of Spoofing Countermeasures and Automatic Speaker Verification: Fundamentals. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2195–2210, 2020. [citation on page 80]

[218] OpenAI. Whisper Official GitHub Repository, 2024. `https://github.com/openai/whisper`, Accessed: 30.03.2024. [citation on page 83]

[219] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. [citation on page 91]

[220] Nicolas M. Müller, Franziska Dieckmann, Pavel Czempin, Roman Canals, Konstantin Böttinger, and Jennifer Williams. Speech is Silver, Silence is Golden: What do ASVspoof-trained Models Really Learn? In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 55–60, 2021. [citation on page 98]

[221] Nicolas M. Müller, Maximilian Burgert, Pascal Debus, Jennifer Williams, Philip Sperl, and Konstantin Böttinger. Protecting Publicly Available Data

With Machine Learning Shortcuts. In *34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023*. BMVA, 2023. [citation on page 98]

[222] Guang Hua, Jiwu Huang, Yun Q. Shi, Jonathan Goh, and Vrizlynn L.L. Thing. Twenty years of digital audio watermarking—a comprehensive review. *Signal Processing*, 128:222–242, 2016. [citation on page 98]

[223] Lauri Juvela and Xin Wang. Collaborative watermarking for adversarial speech synthesis. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024. [citation on page 98]