POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAMME IN INFORMATION TECHNOLOGY

DATA DRIVEN AND SIGNAL PROCESSING TECHNIQUES FOR AUDIO FORENSICS

Doctoral Dissertation of:
Clara Borrelli

Supervisor:
Prof. Augusto Sarti

Co-Supervisor:
Prof. Fabio Antonacci

Tutor:
Prof. Matteo Cesana

The Chair of the Doctoral Program:
Prof. Luigi Piroddi

Year 2022– XXXIV Cycle

Acknowledgments

I would like to begin by expressing my gratitude to my doctoral supervisor Prof. Augusto Sarti, for his valuable mentoring during my Ph.D. study and for giving me the opportunity to be part of the research group. I would like to extend my gratitude to Prof. Fabio Antonacci and Prof. Paolo Bestagini, who many times devoted their time to guide me during this journey and to share their knowledge and expertise.

The Image and Sound Processing Lab (ISPL) has been a great group to grow up as a researcher and as a person during these years. Despite the time apart, I am very grateful for everything I learnt from ISPL people and all the experiences we shared. Therefore thanks to all current and former members of the research group: Alberto Bernardini, Luca Bondi, Michele Buccoli, Massimiliano Zanoni, Antonio Canclini, Federico Borra, Sebastian Gonzalez, Vincenzo Lipari, Sara Mandelli, Nicolo Bonettini, Alessandro Mezza, Edoardo Cannas, Raffaele Malverme, Marco Olivieri, Riccardo Giampiccolo, Davide Albertini, Davide Salvi, Antonio Giganti and Daniele Leonzio. In particular, I would like to thank my colleagues and dear friends Luca Comanducci, Mirco Pezzoli and Francesco Picetti for our many conversations and cherished time together.

I would like to thank my parents and my sisters, Mari and Emma, for being always encouraging and supportive, practically and emotionally. One final special thanks goes to Paolo, for always standing by my side in this journey.

Abstract

THE recent developments and diffusion of audio recording devices, audio editing tools and speech synthesis techniques have opened questions about how to verify the authenticity and integrity of audio assets. On one side, audio recordings are frequently used as fundamental assets in trials and audio analysis methods are needed to asses their admissibility in court. On the other side, falsification of digital media represents nowadays a menace for modern communication and information ecosystems. Fake news, distributed through social media platforms, are frequently distributed together with forged media content, to acquire credibility at the eyes of deceived users and to increase the engagement. The development of detection methods able to expose fake speech signals is therefore paramount.

In this thesis we propose a set of methods for both authenticity and integrity assessment in audio forensics scenarios. Depending on the context, the analysis aims at retrieving information on the recording acoustic scenario or on the speech signal origin. Authenticity is evaluated by matching the extracted cues with a preliminary hypothesis while manipulations are detected by looking at cue's inconsistencies over time.

In the last years, the audio forensic research community has frequently addressed these two problems, proposing solutions based on digital signal

processing techniques or, more recently, the combination of hand-crafted features with supervised classic machine learning method. In this work we present new methods that expand this approach with the use of recent neural-network-based architectures and, by combining all these different strategies, able to successfully address various different scenarios. If large training audio corpora are available, leveraging deep neural networks allows to extract high-level semantic information and to achieve higher generalisation ability and robustness. On the contrary, if either available data or computational power is reduced, methods based on signal model and low-level descriptors are more suitable and still successful, even if less robust to possible small modifications of the input audio.

With this paradigm in mind, we first focus on the definition of two indicators of the acoustic recording environment and present how to blindly estimate them from single-channel noisy audio signal. Then, we focus on synthetic speech detection and attribution for authenticity assessment, presenting solutions that analyse speech signals at various abstraction levels. Finally, two integrity verification methods are presented, focusing in particular on splicing identification and localisation. All methods are validated trough a set of experiments designed to test at the same time detection performance and robustness in real-world conditions. This thesis represents a preliminary investigation, which we hope will help widening the perspectives of audio forensic research.

Contents

List of Figures

List of Tables

Nomenclature

$*$	Convolution
$\mathbf{f}$	Feature vector
$\mathcal{D}$	Dataset
$h(t)$	Room Impulse Response signal
L_w	Window length
$n(t)$	Noise signal
$s(t)$	Source signal
$w(t)$	Window
$X(m, k)$	Short Time Fourier Transform
$x(t)$	Audio signal

Glossary

A

AAD	Absolute Average Difference. 120, 124
AI	Artificial Intelligence. 4
AI	Articulation Index. 33
AUC	Area Under the Curve. 70, 71, 86, 88, 90, 138, 139, 149

C

CI	Clarity Index. 20
CNN	Convolutional Neural Network. 7, 8, 21–25, 27, 30, 45, 54, 62, 148
CQCC	Constant-Q Cepstral Coefficients. 53
CRNN	Convolutional Recurrent Neural Network. 75
CSII	Coherence Speech Intelligibility Index. 34

D

DAPSE	Direct Acoustic Parameter Similarity Estimation. 25–27, 29, 30
DL	Deep Learning. 4, 6, 7, 48, 91, 147

DRR Direct-to-Reverberant Ratio. 20–23, 26, 28

E

EER Equal Error Rate. 85, 88

ENF Electric Network Frequency. 3, 114

ESII Extended Speech Intelligibility Index. 33

F

FDR Free Decay Region. 118, 119, 124

FPR False Positive Rate. 95, 124

G

GAN Generative Adversarial Network. 52

GMM Gaussian Mixture Model. 52

GRU Gated Recurrent Unit. 54, 79, 135, 137

H

HMM Hidden Markov Model. 34, 50, 51

I

IAPSE Indirect Acoustic Parameter Similarity Estimation. 23, 25–27, 29, 30

IEMOCAP Interactive Emotional Dyadic Motion Capture. 83

L

LCIA Low Complexity Intelligibility Assessment. 34

LFCC Linear-Frequency Cepstral Coefficients. 53

LPC Linear Predictive Coding. 51, 60

LTI Linear Time Invariant. 18

M

MFCC Mel-Frequency Cepstral Coefficients. 36, 53, 58, 80, 84

ML	Machine Learning. 6, 7, 48, 61, 74, 91, 147
MSE	Mean Squared Error. 25, 26

N

NN	Neural Network. 6, 7, 50–52, 54, 56–58, 60, 61, 72–74, 77, 103, 108, 112, 127, 145, 150, 154

P

PMVDR	Perceptual Minimum Variance Distortionless Response. 54

R

RBF	Radial Basis Function. 67, 70, 85
RFC	Random Forest Classifier. 39, 40, 66, 67, 70, 83, 85, 99
RFR	Random Forest Regressor. 39
RIR	Room Impulse Response. 10, 18–20, 22, 26, 33, 116, 121, 122, 125, 145, 151–153
RMS	Root Mean Square. 36
RNN	Recurrent Neural Network. 8, 22, 51, 54, 62, 152
ROC	Receiver Operating Characteristic. 70, 85, 87–89, 95, 123–125, 137–139, 149, 150
RT	Reverberation Time. 116–120, 122, 124, 125, 151

S

SC	Spectral Centroid. 36
SER	Speech Emotion Recognition. 9, 73–75, 82, 85, 86
SF	Spectral Flatness. 36
SII	Speech Intelligibility Index. 33

SNR	Signal-to-Noise Ratio. 8, 21–23, 28–30, 32, 38, 39, 44, 92, 94, 95, 122, 124, 125, 145, 148
Speech SII	Speech-based Speech Intelligibility Index. 34
SPSS	Statistical Parametric Speech Synthesis. 50, 51
SRF	Spectral Roll-Off. 36
SSA	Synthetic Speech Attribution. 48, 49, 109
SSD	Synthetic Speech Detection. 48, 53, 55, 60, 61, 66, 68, 72, 74–76, 81–87, 91, 92, 96, 108
SSM	Self Similarity Matrix. 131, 136
STFT	Short-Time Fourier Transform. 24, 27, 53, 75, 79, 83, 84, 117, 119
STI	Speech Transmission Index. 33
STOI	Short-Time Objective Intelligibility. 34
SVM	Support Vector Machine. 39, 40, 66–68, 70, 81, 84, 89, 99
SVR	Support Vector Regressor. 39, 40

T

TDNN	Time Delay Neural Network. 80, 84
TPR	True Positive Rate. 95, 124
TTS	Text To Speech. 8, 50–52, 55–58, 73–75, 77–79, 83, 87–94, 148–150

V

VAE	Variational Auto-Encoder. 57, 58
VC	Voice Conversion. 50–52, 55, 57, 58, 74, 75, 77, 78, 83, 87–92, 150

Z

ZCR	Zero Crossing Rate. 36

CHAPTER 1

Introduction

Digital audio forensics is a research area that aims at defining a set of methodologies and tools for the analysis and evaluation of audio recordings to be used as evidences in courts of law or in criminal investigations. Audio forensic analysis's objective is to determine both the authenticity and the integrity of the evidences, therefore their admissibility in official investigations or trials [104, 180].

This research field frequently overlaps with digital audio analysis and processing themes, like source or microphone identification, speaker recognition, speech transcription or audio quality enhancement. The recent advances of these research themes, often driven by commercial demand, offer novel techniques to aid audio forensic analysis, leading recently for instance to the adoption of data-driven approaches [14, 18, 32]. Moreover, audio forensic themes often intersect with classic image forensic analysis. In fact, the two research areas share the problems addressed (e.g., authentication, attribution, copy-move or deletion detection), obviously formu-

lated in different domains, and joint analysis of multi-modal assets, for instance video and audio, enables to develop more robust and accurate systems [110].

One of the first historical legal cases in which an audio recording has been considered admissible as evidence is the McKeever case in 1958 (US District Court for the Southern District of New York - 169 F. Supp. 426 (S.D.N.Y. 1958)) [104]. During this trial, the judge officially determined seven principles that must be respected to assess the relevance of an audio evidence:

1. *that the recording device was capable of taking the conversation now offered in evidence*;
2. *that the operator of the device was competent to operate the device*;
3. *that the recording is authentic and correct*;
4. *that changes, additions, or deletions have not been made in the recording*;
5. *that the recording has been preserved in a manner that is shown to the court*;
6. *that the speakers are identified*;
7. *that the conversation elicited was made voluntarily and in good faith, without any kind of inducement*.

Some of these principles have lost their significance since the late 50s. In fact, for instance, the first and the second principles suggest a minor familiarity and diffusion of audio recording devices. On the other side, some other principles still inspire the audio forensic research nowadays and represent the main questions that need answers analysing an audio evidence. How can we establish that an audio recording corresponds to actual events? How can we determine the content and the context (i.e., the location, the recording device, the subjects involved) in which the audio recording took place? Can we ensure that the audio recording has not been manipulated or compromised? And finally, can we reconstruct the history and the processing chain of an audio asset? These questions are still relevant nowadays

and inspired the latest works in the field of audio forensics, often adapted to deal with digital representations of audio.

Digital audio authenticity assessment is a broad and long-term task of audio forensics, which aims at determining if a recording evidence corresponds to real events and if it corresponds to events that are of interest for the investigation or trial, i.e., at a specific location, time and with specific subjects. The definition of the problem already reveals the numerous facets and complexity of digital media authentication. In fact, the definition of authenticity may be declined differently, depending on the discriminative cues on which the evaluation method focuses on. In general, the goal is to blindly verify if a set of declared properties of the audio asset are observed in the actual audio signal. Some hypothesis about the location, the recording device or the processing chain of the audio track are formulated, depending on the context in which the evidence is presented. Forensic analysis methods usually extrapolate this information directly from the media content and either confirm or reject the hypothesis, looking for inconsistencies or anomalies. Part of the literature is devoted to the analysis of Electric Network Frequency (ENF) [66, 67, 135, 178], which is captured by both AC and DC powered recording device and represents a signature of location and time of the recording. Other methods aim at detecting the characteristics of the recording environment, like background noise [119] or reverberation time [106]. Other approaches classify the microphone used for the recording, leveraging on the characteristic recording device signature [32, 130]. Another strategy consists in looking for traces left from MP3 double compression, present in compressed audio files that have been tampered or modified [14, 95]. Moreover, some works address the problem combining audio content analysis and container or metadata inspection [125]. All these works attempt to address the initial forensic investigation questions that we presented, but in the last few years audio forensic research community has started addressing also different problems that affect modern society and communication, e.g., fake news and deep fake media diffusion. In this unpredictable and uncontrolled scenarios, classic handcrafted features may reveal their limited generalisation abilities and their difficulties in dealing with unknown audio assets. Therefore, new approaches should be investigated, less dependent on the signal low-level characteristics.

Digital media has established itself as the dominant communication strategy in nowadays society. In fact, it has been observed that re-posting of news containing video, images or audio is on average 11 times the re-postings of only text news [74]. Moreover, social networks and search engines play as aggregation platforms of news, often tailoring the recommendations on user's preferences [137]. This affects the possibility to control the news cycle and the diffusion of information is lacking of intermediation of professional figures. Therefore, social media helps distributing multi-medial news but, at the same time, veridicity of the content is not guaranteed. In this new information environment, the spreading of falsified media has flourished [64], and its diffusion is boosted by the "echo chamber" effect [27]. Often, fake news are crafted to damage the reputation of a public personality or institution, while gaining money through advertising, and they represent a serious threat to key areas of our society, like politics or economics [9, 186]. The creation of falsified media is facilitated by the availability of free video and voice editing software and the recent advances of Artificial Intelligence (AI) techniques make it possible to create deep-fake in completely automatic fashion for all modalities. For instance, it is possible to generate realistic images or videos learning the parameters and sampling from a distribution [10, 82], modify its context to fit a new one [83, 187] and generate speech with end-to-end architectures [164, 170]. In parallel, research in multimedia forensics has proposed several new methods to address the problem and detect falsified media. Impressive progresses have been done in audio, image and video analysis, detecting manipulation using single-modal input [13, 16, 18, 58, 99] or exploiting multi-modal data [63, 94, 110]. Nonetheless, fake media detectors are often challenged by the rapid evolution of attacks and anti-forensics methods. Common limitations are the lack of generalisation ability, necessary to address new synthesis and manipulation attacks, and the lack of robustness to different acquisition conditions or media compression and coding, operations that are frequently applied in social media sharing. For these reasons, recent trends of audio forensics focus on the extraction of high semantic information, like for instance emotion expressed in the speech, rather than analysing the signal at a lower level [63]. The analysis of semantic inconsistencies can be aided with the usage of recent Deep Learning (DL) architec-

tures, exploiting their flexibility and generalisation power. This approach can help understanding not only if the media has been forged, but also having a better insight of how the attack has been executed and what is the purpose of the attacker. For instance, let us imagine a scenario in which a speech of a public person has been manipulated to match a specific possibly dangerous content, different from the one originally expressed. The extraction of prosodic or emotional cues from the synthetic media may highlight inconsistencies at the semantic level and detect the falsification. This approach would allow to exploit one weakness of recent synthesis methods, which focus more on increasing the intelligibility or matching a specific voice identity rather than on conveying a specific emotion. Moreover, the development of these tools may assist manual inspection of falsified media, providing intuitive descriptors to the forensic analyst.

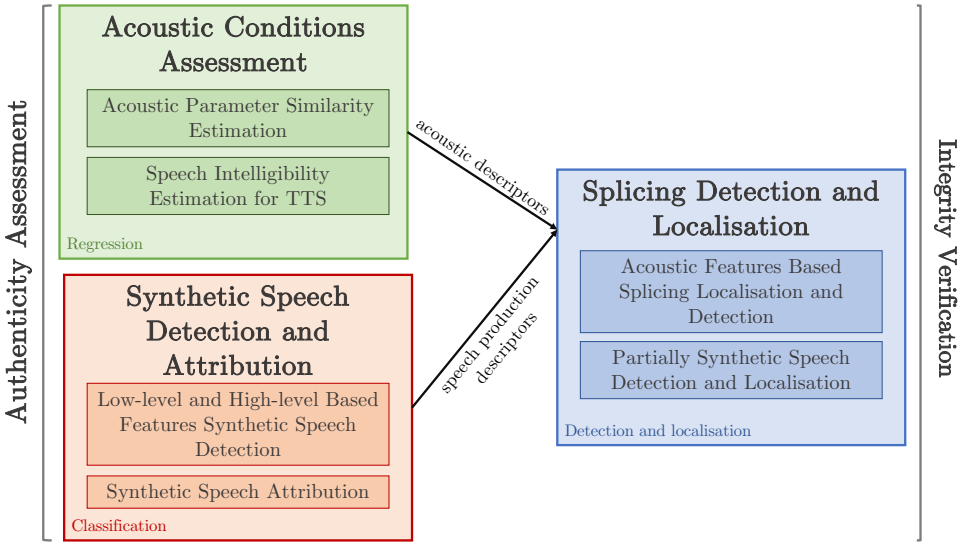


Figure 1.1: *Scheme representing the thesis structure.*

In this thesis we tackle three classic audio forensics topics applying novel methodologies and perspectives. In Figure 1.1 we report an overview of the thesis topics and structure. The first problem we address is acoustic condition assessment. The main objective is to estimate from a single-channel audio signal an acoustic indicator able to express the characteristics of an acoustic environment in a compact fashion. We do not focus simply

on classic acoustic parameter extraction, but we aim at expressing at an higher semantic level the overall acoustic and noise properties of the recording location. This strategy finds several applications in the audio forensics investigation, since it allows to evaluate the authenticity of an audio recording by matching on different acoustic levels the hypothesised recording location of the audio evidence with the actual one. From a methodological standpoint, we adopt for both methods data-driven regression algorithms, exploiting the potentialities of Machine Learning (ML) and DL methods.

The second problem we face is synthetic speech detection and attribution for authenticity evaluation. In this case we decline the theme of authenticity verification looking directly at the source signal, i.e., at speech level, and not at the environment signature. We develop different methods to identify and classify synthetic speech samples, taking into account the recent advances in speech synthesis. Regarding the detection problem, we propose two strategies. The first one envisages the use of low-level features, defined starting from the voice source-filter model. The second technique aims at exploiting more high semantic level features, exploiting recent Neural Network (NN) architectures that allows to describe emotional and prosodic characteristics of voice. The two methods can be applied for the same task, i.e., voice authenticity verification, but they differ in the complexity and training data required.

Finally we address the problem of integrity verification, taking advantage of the descriptors used for authenticity assessment. In particular, we focus on splicing operation detection and localisation. As shown in Figure 1.1, we propose two methods that start from different assumptions on the splicing operation. In the first one we assume that the splicing is a combination of two real recordings performed in two different environments, and we hence exploit reverberation time inconsistencies to detect and localise the splicing point. In the second scenario, we assume that the spliced file is a combination of synthetic and real speech. Therefore, we extract locally a descriptor of the audio signal strictly related to the origin of the speech signal, i.e., if it is real or fake. Again, by looking at the behaviour over time of this representation we are able to spot partially synthetic audio files and locate the point in which the concatenation happened.

From a methodological standpoint, we choose a common framework

for all methods. In most scenarios, we take advantage of ML and DL architectures, including both classic data-driven methods and more recent NN architectures. In particular, we exploit different NN architectures for extracting meaningful and compact embedding, related to different properties of the input. Usually, these networks take as input a simple time-frequency representation of the input and they are trained to learn a feature space related to a specific contextual attribute, which can range from the acoustic conditions of the environment to the emotional content of the speech. The fast development of new deep architectures and their increasing ability to model high semantic level concepts open up to new exciting solutions to audio forensics applications, preliminary investigated in this thesis. Obviously, deep-learning strategies require the availability of large training data corpora. Whenever this requirement is not met, the audio forensic analyst must rely on different tools, able to operate in limited training data scenario. In this case completely deterministic algorithms or systems based on handcrafted features and classic ML algorithms are preferable, even if less robust to signal-level changes. In this thesis we considered both scenarios, and we therefore propose solutions spanning different abstraction and semantic levels and requiring different resources.

1.1 Contributions

In the following we give details about the single contributions included in this thesis work. We follow the order in which each contribute is presented in the thesis. More specifically, the first two works address the task of acoustic conditions estimation for authenticity assessment, aiming at extracting compact and meaningful descriptors of the overall acoustic environment. The third, four and fifth presented contributions are related again to authenticity assessment, this time analysing the speech properties. In fact, we tackle the problems of synthetic speech detection and attribution with two different strategies, using low-level and high-level speech features. Finally, the sixth and seventh contributions address the problem of integrity verification exploiting the acoustic and speech descriptors defined for authenticity assessment. In particular, we focus on detection and localisation of splicing operations in audio excerpts.

Acoustic Parameter Similarity Estimation
In a forensic scenario, an estimate of acoustic parameter similarity between two tracks can be used to verify whether the recordings have been likely acquired in the same environment or not. We propose two methods to estimate acoustic parameter similarity between a speech recording under analysis and a reference one. The first method relies on the estimation of channel-based acoustic indicators that are then compared to extract a similarity measure. The second method directly learns a parameter similarity measure trough siamese neural networks. Both methods take advantage of a Convolutional Neural Network (CNN) state-of-the-art architecture, pretrained on a huge dataset of audio tracks, to compress a time-frequency transform of the input in a compact feature vector. For the evaluation setup, we train and test the two methods on a dataset including different room and noise configurations. The results we obtained show that both methods are able to estimate the defined similarity and they highlight the success of this novel application of metric-learning to this scenario.

Intelligibility Estimation for TTS systems Being able to monitor communications through environmental recordings is an important asset for a forensic investigator, e.g., to prevent terrorist attacks. On one hand, this is becoming easier thanks to the availability of cheaper and smaller audio recordings devices. On the other hand, the automatic analysis of large audio collections of recording is still far from being an easy task. We propose a method to analyze speech audio recordings to establish how reliable they are in terms of automatic transcription capability. This can be used to automatically select relevant non-corrupted portions from huge corpora of recordings for analysts to focus on. This can also be used to help an investigator getting a quick feedback about the quality of his / her recording while deploying a system in a noisy environment. To achieve this goal in a non-intrusive fashion, we use a rich set of time-frequency descriptors as input to a supervised data-driven regressor. We train the method on a large dataset of speech corrupted with different type of noise, comparing the transcriptions of a recent TTS system and the actual transcription to compute the ground-truth reliability metric. At inference stage, the method is able to predict the reliability index by simply analysing frames of the audio signal.

We present the results for different SNR levels and for different time granularity. The numerical analysis of the evaluation shows promising results, both on simulated and real world data.

Synthetic Speech Detection through Low-Level features Several methods for synthetic audio speech generation have been developed in the literature through the years. With the great technological advances brought by deep learning, many novel synthetic speech techniques achieving incredible realistic results have been recently proposed. Nonetheless, several speech synthesis methods still exploit the source-filter model for speech signals. Even methods that do not explicitly use this model (e.g., CNN, Recurrent Neural Network (RNN), etc.) create a speech signal through operations in the temporal domain (e.g., temporal convolutions, recursion, etc.). For this reason, we propose a set of features, starting from a short-term and long-term analysis of the input, able to capture salient information about the speech under analysis over time. The feature set is then used in combination with a simple supervised classification algorithm. The proposed detector is validated on a publicly available dataset consisting of 17 synthetic speech generation algorithms ranging from old fashioned vocoders to modern deep learning solutions. Results show that the proposed method outperforms recently proposed detectors based on signal-level features in the forensics literature.

Synthetic Speech Detection through High-Level features In this work we address the problem of synthetic speech detection with a different angle. We first propose a new audio spoofing detection system leveraging emotional features. The rationale behind this proposed method is that audio deepfake techniques cannot correctly synthesize natural emotional behavior. Therefore, we feed a detector with high-level features obtained from a state-of-the-art Speech Emotion Recognition (SER)) system. As the used descriptors capture semantic audio information, the proposed system proves robust in cross-dataset scenarios, outperforming the considered baseline on multiple datasets. This emotional-cues based method is effective only for a subset of synthetic speech generation methods. To overcome this limitation, we propose a second semantic approach which focuses on prosodic and speaker identity features. In fact, on one side we analyse voice

prosody, in the sense of variations in rhythm, pitch or accent, extracted through a specialized encoder. On the other side, we use a state-of-the-art network for automatic speaker verification to extract a speaker embedding vector, able to distillate all individual voices attributes. We show that the fusion of these two embeddings, fed to a simple binary classifier, allows the detection of speech generated with a larger set of algorithms. Also in this case, our results show improvements over baseline methods and good generalization properties over multiple datasets.

Synthetic Speech Attribution In this scenario we aim at predicting not only if the speech input is synthetic but also which algorithm has been used to create the audio input. In fact, the ability to recognise what synthesis technique has been adopted would allow us to answer questions about the origin, authorship, or diffusion record of individual media assets. To achieve this goal we make use of low-level features, based on short-term and long-term analysis, combined with a multi-class supervised classification algorithm. To increase the robustness of the proposed method, we formulate two different scenarios, closed-set and open-set. We test the proposed method in both cases using a large dataset which includes several different synthesis techniques. Results show very good classification performances, especially in the closed set scenario. The performances on the open-set case are encouraging, but more challenging for the proposed classification method.

Speech Audio Splicing Detection and Localisation through Reverberation Cues Manipulating speech audio recordings through splicing is a task within everyone's reach. Indeed, it is very easy to collect through social media multiple audio recordings from well-known public figures (e.g., actors, politicians, etc.). These can be cut into smaller excerpts that can be concatenated in order to generate new audio content. The ability of detecting whether a speech recording has been manipulated is a task of great interest in the forensics community. In particular, we focus on speech audio splicing detection and localization. We leverage the idea that distinct recordings may be acquired in different environments, which are typically characterized by distinctive reverberation cues. Exploiting this property, our method estimates inconsistencies in the reverberation time throughout

a speech recording. If reverberation inconsistencies are detected, the audio track is tagged as manipulated and the splicing point time instant is estimated. The method is evaluated on a dataset of single spliced tracks, using fragments generated with both simulated and real Room Impulse Response (RIR) with different noise levels. We present the results for both detection and localisation task, varying the noise level and the reverberation time difference between the audio segments used for the splicing operation. The evaluation stage shows that the proposed method is particularly successful in lower presence of noise and with a more pronounced variation of the reverberation time.

Partially Synthetic Speech Identification and Splicing Localisation We investigate the detection of partially synthetic speech and the localisation of the splicing point, a problem rarely addressed in the literature. Nonetheless, it is easy to imagine a malicious attack where only single words or utterances of a longer talk are substituted with synthetically generated ones. The content expressed by a spliced audio may change drastically with respect to the original one, even modifying only small portions. To address the problem, the proposed method makes use of an end-to-end network to extract embeddings related to the speech origin (real or synthetic) on sliding short time frames. The distance between the sequence of embeddings define a self-similarity matrix, from which we extract a novelty function. When the novelty function shows peaks of significant prominence, a splicing point is detected and localized. To force the embedding extractor to learn a feature space in which the distance measure is meaningful, we adopt a metric-learning approach, using the triplet-loss during its training stage. We tested the system on two datasets of same-speaker spliced tracks: the first dataset manipulated tracks have a single splicing point, while in the second one they have two splicing points. We evaluate detection and localisation task on both datasets, obtaining good results in both scenarios, in particular when the same set of algorithms is used for generating the synthetic fragments and for training the embedding extractor.

1.2 Thesis Outline

In this thesis we include the contributions just illustrated in the previous section and organised in three different chapters, corresponding to three different tasks of the audio forensic analysis. In the following, we give detail about the organisation of the thesis In Chapter 2 we present the problem of acoustic conditions assessment. The reader may find an introduction to the reverberation model and the definition of common acoustic indicators in Section 2.1. Then, in Section 2.2 we propose and evaluate a data-driven method for acoustic similarity estimation of speech recording. In Section 2.3 we introduce a reliability metric for speech-to-text systems, designed to automatically analyse big corpora from audio surveillance recordings. In Chapter 3, we tackle the problems of synthetic speech detection and attribution. The reader may find a review of the state of the art on synthetic speech generation and synthetic speech detection in Section 3.1. We then first investigate the problem of synthetic speech detection in Section 3.2, proposing several methods, based on low-level features, in Section 3.2.2, and on high-level features, in Section 3.2.3. Then, we propose a method for synthetic speech attribution in Section 3.3.

In Chapter 4 we analyse the splicing detection and localisation problem from two standpoints. We first report the latest works in the literature in Section 4.2. Then, we propose a method for splicing detection and localisation exploiting discontinuities in the estimate of reverberation time in Section 4.3. In the second part of the chapter, in Section 4.4 we propose a metric-learning based framework to detect partially synthetic speech samples and localise the splicing position.

Finally, in Chapter 5 we comment the overall results obtained for the different tasks, we highlight strengths and weaknesses of the adopted methodologies and we propose some future works in the field.

1.3 List of Publications

In the following, the list of publications presented in the main corpus of the thesis:

- [17] C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro.

Automatic reliability estimation for speech audio surveillance recordings. In *IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2019

- [22] D. Capoferri, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Speech audio splicing detection and localization exploiting reverberation cues. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2020

- [18] C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Synthetic speech detection through short-term and long-term prediction traces. *EURASIP Journal on Information Security*, 2021:1–14, 2021

- [29] E. Conti, D. Salvi, C. Borrelli, B. Hosler, P. Bestagini, F. Antonacci, A. Sarti, M. C. Stamm, and S. Tubaro. Deepfake speech detection through emotion recognition: A semantic approach. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (Accepted)*, 2022

- [122] M. Papa, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. A data-driven approach for acoustic parameter similarity estimation of speech recording. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (Accepted)*, 2022

- [7]L. Attorresi, D. Salvi, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Combining speaker identification and prosody analysis for synthetic speech detection. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (Submitted)*, 2022

- [23] F. Castelli, D. Salvi, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. A metric learning approach to synthetic speech splicing detection and localisation. In *European Signal Processing Conference (EUSIPCO) (Submitted)*, 2022

Here a list of other publications not included in this thesis:

- [19] C. Borrelli, A. Canclini, F. Antonacci, A. Sarti, and S. Tubaro. A denoising methodology for higher order ambisonics recordings. In

IEEE International Workshop on Acoustic Signal Enhancement (IWAENC), 2018

- [71] R. T. Irene, C. Borrelli, M. Zanoni, M. Buccoli, and A. Sarti. Automatic playlist generation using convolutional neural networks and recurrent neural networks. In *European Signal Processing Conference (EUSIPCO)*, 2019

- [26] S. Cherubin, C. Borrelli, M. Zanoni, M. Buccoli, A. Sarti, and S. Tubaro. Three-dimensional mapping of high-level music features for music browsing. In *IEEE International Workshop on Multilayer Music Representation and Processing (MMRP)*, 2019

- [128] G. Picardi, C. Borrelli, A. Sarti, G. Chimienti, and M. Calisti. A minimal metric for the characterization of acoustic noise emitted by underwater vehicles. *Sensors*, 20(22):6644, 2020

In [17] and [18] the author has performed the theoretical formulation, the design, the implementation and the results analysis. The works [22,29,122] are derived from master thesis works supervised by the author and therefore the first author has contributed to the implementation of the methods.

CHAPTER 2

Acoustic Conditions Assessment

Nowadays, thanks to the diffusion of cheap and small recording devices, speech and audio signals are acquired and analysed for several applications. Examples are speaker recognition for authentication, speech recognition for voice controlled user interfaces, automatic transcription or automatic language translation. The correct functioning of the mentioned applications strongly depends on the audio recording environment. In fact, the presence of external noises or strong reverberation components can compromise their performances [49, 141]. For these reasons, the advancements in speech analysis has been followed by analogous progresses in room acoustic analysis. The most recent methods are usually assuming a real-world scenario, where only one microphone is available and several noise sources may be interfering with the main source. The main focus is to blindly infer one or more acoustic properties of the recording environment, like room volume, reverberation time or noise level estimation, starting from a single-channel audio signal [53, 102].

Similar scenarios can be observed in the field of audio forensics. Often audio forensic analysis is based on audio acquisitions or wiretaps, that are usually crucial in the subsequent investigations. Unfortunately, these audio signals are often acquired in far from optimal recording conditions. The strong presence of noise or reverberation can compromise the legal validity of the audio evidence. Therefore, automatic estimation of acoustic indicators may assist the forensic analysis in two phases. Firstly, such systems can be used during the setup of an acquisition system for wiretapping. An operators would be able to asses the acoustic conditions and noise characteristics and, for example, may subsequently reconsider the microphone position in the ambient. Secondly, such systems may be used when analysing the forensics audio evidences. In fact, when an evidence is exhibited in court it is always presented in a specific context. The context is defined surely by the number and the identity of speakers, but also by the environment in which the recording has taken place, e.g., in open air, in a small room or in a noisy hallway. The possibility to automatically infer the acoustic context allows the analyst to match it to the assumed or declared acoustic context.

In this chapter we propose two novel methods for acoustic condition assessment starting from single-channel microphone signal. In both cases the goal is to estimate a general indicator that describe in a compact fashion the environment properties of the analysed recording.

In the first section we introduce the concept of acoustic parameter similarity and present two methods able to estimate it. Given two environments, one reference and one under analysis, the proposed systems allows to estimate the similarity between the two, in terms of five different acoustic indicators.

In the second section we focus on intelligibility estimation, tailored to automatic transcription applications. We first define a metric able to describe the reliability of text-to-speech in a specific acoustic context. The mentioned metric is computed starting from the result of transcription and the actual transcripts. We then propose a framework to automatically predict the reliability metric value, starting solely from the audio input.

These two methods address two different problems but share some characteristics. Both methods define and estimate indicators that belong to high-

level semantic. In fact, they are not directly linked to a specific acoustic phenomenon, but they aim at representing the acoustic condition information in a more abstract fashion.

Moreover, in both cases we consider the task as a regression problem, using a mixed signal-processing and data-driven approach. After defining each acoustic metric, we first select a meaningful feature representation of the audio input. These features are fed to a machine learning or deep learning algorithm, which during the training step learns the mapping between the input signal and the desired output. Therefore, our methods on one side exploit signal processing algorithms to extract a compact representation of raw audio signal. On the other side, they exploit the generalisation ability of data driven methods to learn the semantically high-level relationship between recorded audio and acoustic context.

In this chapter we first introduce the reader to some background notions of acoustic. Then, in the Section 2.2 we introduce two metric-learning based methods for acoustic parameter similarity estimation. In Section 2.3 we present a machine learning method for estimating intelligibility for text-to-speech applications. Finally, in Section 2.4 we draw some final conclusions.

2.1 Background

In this section we introduce some room acoustics basic concepts that will help the reader in the following. We first introduce a model for reverberation behaviour in enclosed spaces. We then define a set of acoustic parameters commonly used to describe compactly the reverberation properties of environments.

2.1.1 Reverberation model

Let us consider an indoor environment enclosed by walls. An omnidirectional acoustic source (e.g., a speaker, a loudspeaker, etc.) and a receiver (e.g., a listener, a microphone, etc.) are present in the room. When the source emits an audio signal, the receiver receives multiple delayed and attenuated copies of the signal. Indeed, the microphone is hit by waves propagating directly from the source to the receiver, i.e., direct path components,

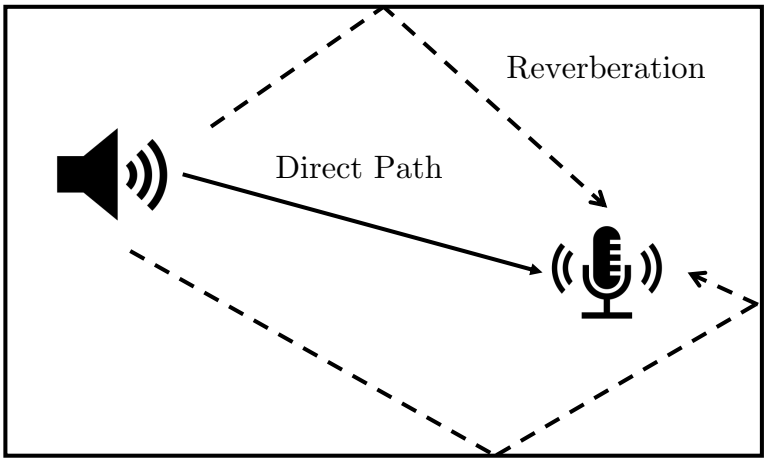


Figure 2.1: *Schematic of room reverberation setup.*

as well as waves reflected by the ground, the walls and other surfaces, i.e., reverberant component. In Fig 2.1 an example of the described setup is reported. The propagation of the signal from the source to the microphone within the environment can be then well approximately by a Linear Time Invariant (LTI) system. Therefore, the signal acquired at the microphone can be modeled as

$$x(t) = s(t) * h(t) = \int_{-\infty}^{\infty} s(t-\tau)h(\tau)d\tau, \tag{2.1}$$

where $s(t)$ is the source signal, $h(t)$ is the system impulse response known as Room Impulse Response (RIR), and the operator $*$ represents convolution. As the RIR depends on the environment geometry and the source and receiver position, it contains valuable information about the recording setup.

Let us assume that $s(t)$ is a Dirac function, which can be approximated as a short sound impulse emitted by an omnidirectional point source. The recorded $x(t)$ corresponds to $h(t)$, which is typically composed by a series of attenuated and delayed pulses as shown in Fig. 2.2. A spherical wave propagates from the source in all directions and the wave-front that first reaches the receiver is the one that follows the direct path from the source to the receiver. Therefore, the first pulse of a RIR represents the *direct signal* propagation. This direct signal is followed by weaker components, i.e.,

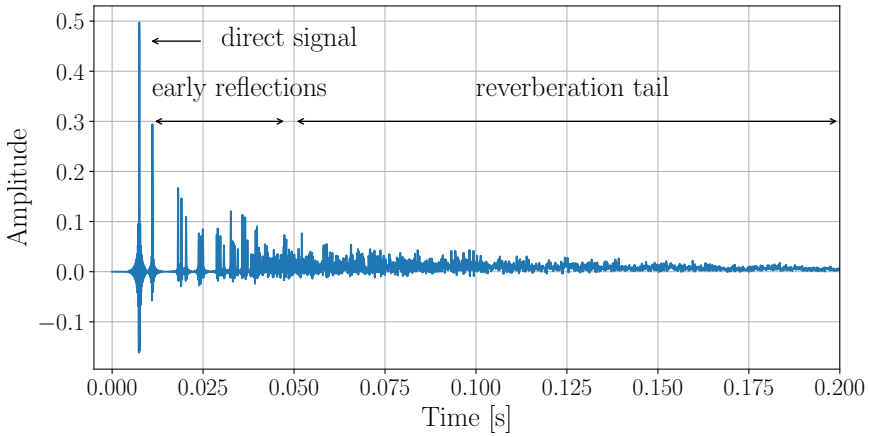


Figure 2.2: *Components of a typical room impulse response characterizing acoustic propagation from a source to a receiver within a closed environment.*

waves that have been reflected by the room walls one or multiple times before reaching the receiver. These reflections, called *early reflections*, have lower intensity because of the increased area of the spherical wave-front as time increases and because of the sound-absorbing property of the walls or objects in the room. As the number of reflections increases, the waves continue to travel in all directions until all the energy has been absorbed. The density of these later reflections increases with time, while the intensity decreases. This decaying *reverberation tail* is often perceived by the listener as the room reverberation.

2.1.2 Acoustic indicators

The focus of this chapter is the assessment of acoustic conditions and context given a single-microphone acquisition. We present here a set of acoustic indicators that are broadly used to describe in a compact fashion the reverberant behaviour of a recording environment. This set includes three different channel-based objective measures that depend on the RIR $h(t)$ from the source to the receiver in the considered setup [116]. In the following we list the definitions of the considered parameters.

- The T_{60} is a compact descriptor of the room reverberant behaviour. It is estimated as the time in seconds the energy decay curve (i.e., the tail integral of the squared RIR) takes to drop by 60 dB [116].

The higher the T_{60}, the longer the reverberation. It is worth noticing this parameter does not strictly depends on the specific source-receiver positions of the measured room, so usually multiple RIR measures are combined to extract the final T_{60} value.

- The Direct-to-Reverberant Ratio (DRR) measures the ratio between the energy contained in the direct arrival and that of the rest of the reverberant tail. It is defined as [116]

$$\text{DRR} = 10 \log_{10} \frac{\sum_{t=0}^{t=t_d} h(t)^2}{\sum_{t=t_d+1}^{\infty} h(t)^2}, \tag{2.2}$$

  where the samples $h(t)$ of the impulse response from index 0 up to t_d correspond to the direct source-receiver propagation, whereas samples from $t_d + 1$ correspond to the reflection paths. Usually t_d corresponds to a time interval of 10 ms starting from the arrival time of the direct sound.

- The Clarity Index (CI) is a compact descriptor that can be linked to the way speech signals can be well perceived and understood. It is defined as [116]

$$\text{CI} = 10 \log_{10} \frac{\sum_{t=0}^{t=t_e} h(t)^2}{\sum_{t=t_e+1}^{\infty} h(t)^2}, \tag{2.3}$$

  where t_e/F_s usually corresponds circa to either 50 ms or 80 ms, leading to two different indices, C_{50} and C_{80}, respectively.

2.2 Acoustic Similarity Estimation

Recently speech signal has been extensively used for multiple applications (e.g., speech recognition, voice user interfaces) and generally the input speech signal is acquired with a single microphone in a surrounding environment which may present unpredictable acoustic characteristics [117].

Depending on the recording context, noise level and reverberation behaviour may change drastically, thus the ability to monitor the acoustic characteristics of the environment is crucial for the effectiveness of speech

analysis systems [49,141]. For this reason, the possibility to evaluate acoustic parameter similarity between a reference audio recording and a track under analysis is interesting in different contexts.

As an example, a method that estimates acoustic similarity can be embedded in data-driven robust speech recognition systems to improve their performances. To overcome the problem of domain mismatch between clean training data and noisy in-the-wild speech signals, data augmentation techniques are often used to increase the robustness of the systems [60,91]. Assessing the similarity between real-world audio signal under analysis and a reference track from the training/evaluation set can help reducing potential errors by possibly re-defining training data. Estimating acoustic parameter similarity can also help in a preliminary phase to select the most suitable speech analysis system or parameter tuning depending on the acoustic context [175].

Acoustic similarity estimation can help in facing challenges encountered in audio forensics as well. A forensic investigator often verifies not only the content and the speaker identity of a speech audio evidence, but also the environment in which the recordings took place [112]. The analysis of the similarity between the audio track under analysis and a reference one allows to verify the match between the claimed environment and the actual one in which the recording has been performed. Moreover, audio evidences can be maliciously manipulated applying splicing, i.e., concatenating multiple segments from different audio tracks [185]. If the acoustic recording conditions of the spliced segments are different, the analysis of acoustic parameter similarity can highlight inconsistencies and localize splicing points [22].

In this work we propose two data-driven methods to assess acoustic parameter similarity between two single-channel speech audio recordings.

The first method employs a CNN that maps a time-frequency representation of the two inputs to five different acoustic indicators: SNR, reverberation time (T_{60}), DRR, and two different clarity indexes (C_{50} and C_{80}) [116]. Acoustic similarity is then defined as the Euclidean distances between the parameters estimated from the reference signal and the signal under analysis.

The problem of non-intrusive acoustic parameter estimation from monau-

ral speech signals has been explored in the audio analysis research community. In [33] sub-band decomposition is combined with statistical analysis to extract T_{60} and DRR directly from speech signal. In [50], a deep learning approach is used for reverberation time approximation. In [123] the authors estimate C_{50}, proved to be highly correlated to performances of phoneme recognition systems, using short-term features and decision tree learning. Another popular strategy is to jointly estimate several different parameters with a single estimator, rather than using one different estimator for each parameter. In [175] a set of features based on Gabor filters is fed to a feed-forward neural network to estimate both T_{60} and DRR. In [124] a set of frame-based features are extracted and used as input for a RNN, able to model the temporal correlation between features and outputs, i.e., DRR and T_{60}. Recently, the authors of [102] proposed to jointly estimate three different acoustic indicators (i.e., T_{60}, SNR and DRR) using a CNN trained on simulated reverberant speech samples. The authors assert that approximating multiple outputs helps data-driven approaches to be robust in noisy conditions. The results are interesting and inspired the multi-task learning methodology adopted in this work.

The second data-driven method we propose addresses directly the problem of acoustic parameter similarity estimation, without explicitly extracting acoustic parameters indicators values. In this case a metric learning approach is tested, defining a siamese architecture for each acoustic parameter which is then trained using a contrastive loss.

The two methods are evaluated on a large dataset of simulated RIRs convolved with speech signals corrupted by noise. Results show that for each considered acoustic indicator the second method outperforms the first one in estimating acoustic similarity between the two audio inputs. However, the first method provides more interpretable responses, being expressed in terms of acoustic parameters values differences.

2.2.1 Signal Model

Let us consider a sampled audio signal $x(t)$ acquired with sampling frequency F_s in a reverberant and noisy environment with a single micro-

phone. We can express $x(t)$ as

$$x(t) = s(t) * h(t) + n(t), \tag{2.4}$$

where $s(t)$ is the source signal, $h(t)$ is the RIR between the source and the receiver, and $n(t)$ is an additive background noise term. In this work the source $s(t)$ is assumed to be a speech signal produced by a source randomly positioned in the considered room. Also the position of the used microphone is randomly selected. With these definitions at hand, the SNR between the main source $s(t)$ and the additive noise $n(t)$ can be written as

$$\mathrm{SNR} = 10 \log_{10} \frac{\sum_t x(t)^2}{\sum_t n(t)^2}. \tag{2.5}$$

2.2.2 Proposed Method

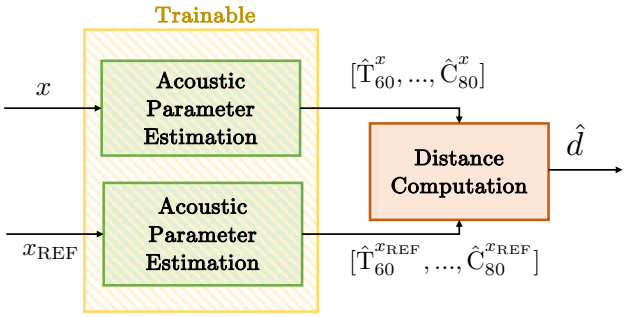
In this work we propose a method to blindly estimate acoustic parameter similarity between a reference audio signal and a signal under analysis by means of a distance measure. To do so, we consider a set of acoustic parameters which includes SNR and four channel based acoustic parameters. In particular, we select T_{60}, DRR, C_{50} and C_{80} as defined in Section 2.1

For each acoustic parameter, the distance measure is defined to have low values if the considered signals have been recorded in two environments with similar values for the considered acoustic indicator, e.g., similar reverberation behaviour or noise level, high values otherwise.

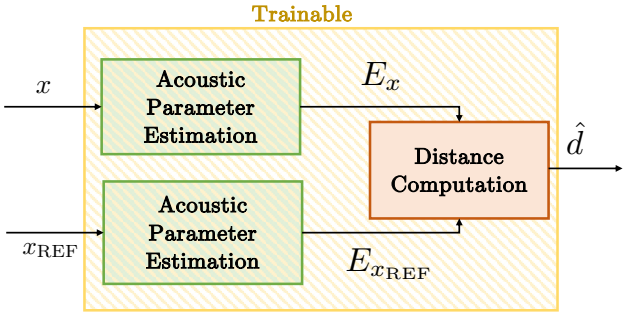
Formally, let us consider an audio speech signal under analysis $x(t)$ associated to the acoustic parameters $\mathrm{SNR}^x, \mathrm{T}^x_{60}, \mathrm{DRR}^x, \mathrm{C}^x_{50}$ and C^x_{80}. Let us consider a reference signal $x_{\mathrm{REF}}(t)$ associated to the acoustic parameters $\mathrm{SNR}^{x_{\mathrm{REF}}}, \mathrm{T}^{x_{\mathrm{REF}}}_{60}, \mathrm{DRR}^{x_{\mathrm{REF}}}, \mathrm{C}^{x_{\mathrm{REF}}}_{50}$ and $\mathrm{C}^{x_{\mathrm{REF}}}_{80}$. The goal of our work is to propose a method that takes x and x_{REF} as input, and returns the Euclidean distance d between each pair of acoustic parameters (i.e., $d_{\mathrm{T}_{60}} = \sqrt{(\mathrm{T}^x_{60} - \mathrm{T}^{x_{\mathrm{REF}}}_{60})^2}$ if T_{60} is considered).

To do so, we explore two possible data driven-strategies that are detailed in the following.

Indirect Acoustic Parameter Similarity Estimation (IAPSE).
Figure 2.3a shows the pipeline of the first method, named IAPSE. This method first estimates the acoustic parameters, and then estimates the sim-



(a) *Indirect Acoustic Parameter Similarity Estimation (IAPSE)*



(b) *Direct Acoustic Parameter Similarity Estimation (DAPSE)*

Figure 2.3: *Acoustic similarity estimation pipelines: (a) IAPSE pipeline: first acoustic parameters are estimated, then acoustic parameter similarity is computed. (b) DAPSE pipeline: similarity is directly estimated bypassing acoustic parameters.*

ilarity based on them. Each input (i.e., $x(t)$ and $x_{\mathrm{REF}}(t)$) is separately processed by the acoustic parameter estimation block consisting of a CNN that estimates acoustic parameters. Parameters are then compared in the distance computation block that returns the estimated Euclidean distances $\hat{d}$ for each parameter. Notice that in this method, only the acoustic parameters estimation block is data-driven, thus trainable.

The acoustic parameter estimation block predicts the set of acoustic parameters associated to its input signal. Following the approach proposed in [102], the parameters are jointly estimated using a CNN fed with a time-frequency representation of the input. In particular, considering the input $x(t)$, we compute its log-mel spectrogram. To do so, Short-Time Fourier Transform (STFT) is applied to $x(t)$, its magnitude is integrated over mel-spaced bins and a logarithmic function is applied to the magnitude of each

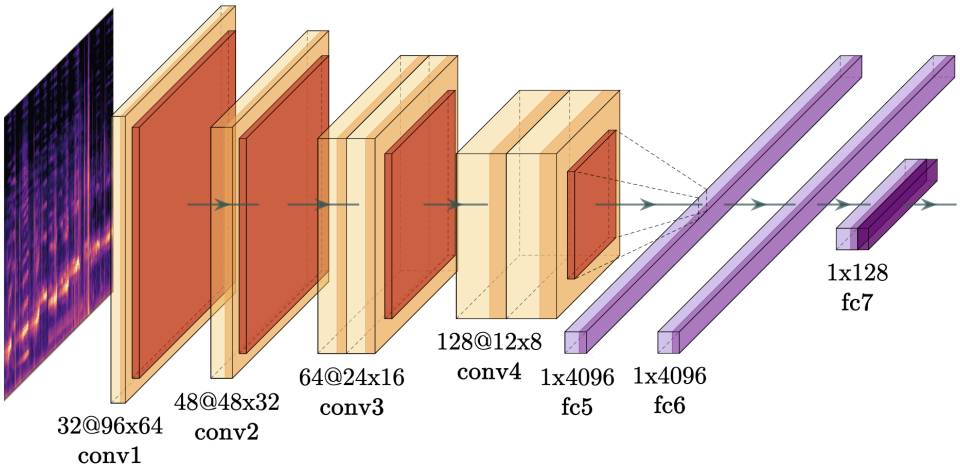


Figure 2.4: *VGGish architecture for embedding extraction.*

bin as explained in [62]. This transformation produces a 2D representation of the input $X(m,k)$ where the index k indicates the frequency bin while the index m indicates the correspondent time window. This processed input is fed to a CNN. We decided to use the popular VGGish network [62], whose architecture is reported in Figure 2.4 and detailed in [62]. Additionally, a dense layer of five neurons is concatenated to the final VGGish layer to estimate the five parameters $\hat{\text{SNR}}^{x}$, $\hat{\text{T}}_{60}^{x}$, $\hat{\text{DRR}}^{x}$, $\hat{\text{C}}_{50}^{x}$ and $\hat{\text{C}}_{80}^{x}$. The loss function used for training the model is the Mean Squared Error (MSE) between the predicted and ground-truth acoustic parameters values. The same process is applied independently to the second input $x_{\text{REF}}(t)$.

Once the network is trained, the distance computation block takes all estimated acoustic parameters as input, and returns the Euclidean distance $\hat{d}$ for each parameters pair. For instance, if T_{60} is considered, we obtain $\hat{d}_{\text{T}_{60}} = \sqrt{(\hat{\text{T}}_{60}^{x} - \hat{\text{T}}_{60}^{x_{\text{REF}}})^2}$. The same applies to the other parameters.

Direct Acoustic Parameter Similarity Estimation (DAPSE).
Figure 2.3b reports the pipeline for the second method, named DAPSE. This method directly estimates similarity bypassing acoustic parameters. Differently from IAPSE method, the inputs $x(t)$ and $x_{\text{REF}}(t)$ are jointly processed by a siamese CNN [93], which extracts audio embeddings and learns the desired distance measure $\hat{d}$ for the considered parameter. This is an end-to-end trainable method that is completely data-driven compared to

IAPSE approach. It focuses on learning a distance measure on one single acoustic parameter at a time, rather than learning jointly an estimate for all parameters and then separately computing their distances.

The siamese CNN is composed of two twin networks that share the weights, jointly process two distinct inputs (i.e., $x(t)$ and $x_{\text{REF}}(t)$) and are joined at the top by a specific function. In our scenario, each one of the twin networks is a VGGish model [62] presented in Figure 2.4. Each VGGish acts as an embedding extractor that maps the inputs x and x_{ref} into the embedding vectors E_x and $E_{x_{\text{REF}}}$, respectively. Embeddings are fed to a layer that computes the Euclidean distance $\hat{d}$ between them. As the entire system is trained at once, the loss function is built such that the network minimizes the difference between the learnt distance $\hat{d}$ and the ground truth distance d by means of MSE. This architecture and loss are inspired by well-known siamese architecture and contrastive loss [59].

It is worth noticing that in DAPSE method the configuration of the embedding space and the subsequent Euclidean distance is learnt through training, while in IAPSE the learning aims only at estimating directly the acoustic parameters. Moreover, to the best of our knowledge, the choice of deep metric learning in acoustic parameter similarity estimation is completely novel.

2.2.3 Dataset

For evaluating the proposed method we created an ad-hoc dataset. This consists of several speech signals corrupted by noise and convolved with RIRs obtained simulating a large number of rooms with different acoustic properties.

Following the signal model introduced in Section 2.2.1, we used the TIMIT dataset [51], consisting of 6300 different utterances from different speakers, as clean speech signals $s(t)$ sampled with $Fs = 16000$ Hz. As additive noise $n(t)$, we considered both white noise and babble noise. With the term babble noise we indicate noise present in a multi-speaker environment, therefore the interference corresponds to one or a combination of multiple speech signals [96]. The considered noise levels are SNR $\in \{10, 15, 25, 35\}$ dB. The RIRs $h(t)$ have been simulated using Pyroomacoustic toolbox [131], allowing the direct control of reverberation

parameters. In particular, we defined a set of shoe-box rooms with volumes spanning between 27 m^3 and 256 m^3 and T_{60} values between 200 ms and 1200 ms. From the simulated RIRs we computed C_{50}, C_{80} and DRR values following (2.2) and (2.3). We obtained DRR $\in [-21.67, 15.37]$ dB, $C_{50} \in [-12.40, 20.66]$ dB and $C_{80} \in [-8.66, 25.59]$ dB. The total number of room configurations is approximately 100. The final dataset considering all speeches, noises and RIRs counts 104000 tracks, which is suitable for our data driven approach.

2.2.4 Training and metrics

The CNNs presented in Section 2.2.2 are trained using the proposed dataset. The training set is composed of 90000 audio tracks while the test set is composed of 14000 audio tracks. To ensure the generalization capability of the networks, we divide training and test sets such that the subset of rooms considered during the training phase is disjointed from the subset used for testing.

For log-melspectrogram computation we consider a window of 0.96 s for each track and STFT is applied using Hanning window of length $L_w = 0.025$ s and hop size $L_h = 0.010$ s. The magnitude of the result is mapped in mel scale, using 64 bins spanning from $F_{min} = 125$ Hz up to $F_{max} = 7500$ Hz. Finally the natural logarithm function is applied. The final 2D matrix has dimension 96x64 samples, as required by VGGish. To improve the overall performances of the proposed system, the actual training consists in fine tuning the original VGGish network pre-trained for audio classification task on a very large dataset [52]. For both configurations, the selected optimizer is Adam with learning rate set to 0.001. Training is performed for 100 epochs using the early-stopping mechanism with patience 10 (i.e., if validation loss does not improve for 10 epochs, the training is stopped and the best validation model is saved).

To evaluate the proposed systems we choose as evaluation metric the Pearson correlation coefficient ρ, as in [102]. This is used to compare estimated acoustic parameters or acoustic similarity distribution against the ground truth.

2.2.5 Results

In this section we present the achieved results. First, we analyse the performances of the acoustic parameter estimation block of IAPSE method. Then, we present the results on the task of acoustic parameter similarity estimation obtained using IAPSE and DAPSE methods.

Acoustic Parameter Estimation

These experiments validate the use of VGGish to estimate five acoustic parameters jointly in the acoustic parameter estimation block of IAPSE. First, we compare the proposed acoustic parameter estimation method exploiting VGGish against a baseline. As baseline, we use an extended version of the work presented in [102], adapted to jointly estimate the five acoustic parameters used in this work rather than the three parameters used in [102]. To this purpose, Table 2.1 shows a comparison between the baseline and the proposed acoustic parameter estimation method. The results are expressed in terms of ρ computed between the predicted acoustic parameters and the ground-truth values, for each one of the five parameters. We can observe that performances are satisfactory for all the estimated parameters. Some parameters, like DRR, C_{50} and C_{80} are more challenging for both networks, while the noise level is often easily predicted, reaching almost $\rho = 0.98$. The proposed method has on average a slight improvement over the baseline, which motivates us in keeping VGGish as back-end for the proposed system.

As second experiment, we investigate the effects of jointly estimating all five parameters compared to the estimate of each parameter separately. In Table 2.2 we report the prediction results for each parameter estimated with two different configurations. In the first one, indicated as single-output, each parameter is estimated separately with a specific network trained for one output. In the second configuration, indicated with multi-output, all parameters are jointly estimated from a single network as proposed in Section 2.2.2. As shown in [102], this strategy allows to exploit the information shared between the different acoustic parameters and helps the learning phase in achieving higher generalization capacity. The experiment confirms that the multi-output strategy improves over the single-output one, apart for SNR estimation, for which there are no sensible differences.

Table 2.1: *ρ values for the proposed acoustic parameter estimation method and the baseline.*

	DRR	T60	SNR	C50	C80
Proposed	0.857	0.932	0.990	0.900	0.896
Baseline	0.841	0.923	0.979	0.901	0.912

Table 2.2: *ρ values obtained with single-output and multi-output acoustic parameter estimation configurations.*

	DRR	T60	SNR	C50	C80
Multi-output	0.857	0.932	0.990	0.900	0.896
Single-output	0.856	0.923	0.992	0.895	0.889

Finally, we want to evaluate the robustness of the method to different noise conditions. To this purpose, in Figure 2.5 we present the results for the multi-output proposed architecture for different SNRs. As expected, ρ values decrease for lower SNR values, but also in the worst scenario the values are acceptable and the system is effective.

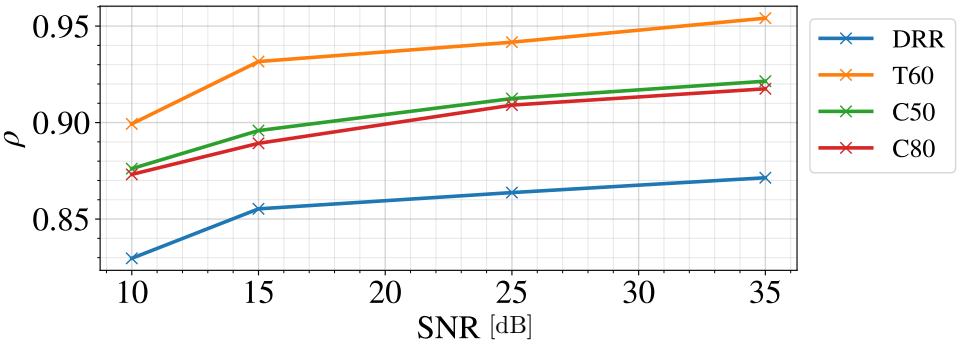


Figure 2.5: *ρ values for proposed acoustic parameter estimation method varying SNR values.*

Acoustic Parameter Similarity Estimation.

In this section we compare the IAPSE and DAPSE approaches. As mentioned, in these experiments we evaluate the estimation of acoustic parameter similarity between two environments rather than acoustic parameter value estimation, as in the previous section. In the first experiment we com-

pare the performances between the two methods in terms of ρ between predicted and real distance for each acoustic indicator. For IAPSE approach, we test a single network performing a joint estimation of parameters, for maximizing prediction accuracy. About DAPSE method, one siamese network is trained for each parameter. As evident from Table 2.3, the distance estimation is more accurate using DAPSE strategy, i.e., when the network is specifically trained for the task of distance estimation rather than for parameter estimation. We believe that the use of metric learning and siamese configuration helps in learning a meaningful embedding space, strictly related to the considered acoustic properties. Two audio tracks that correspond to acoustically similar environments correspond to two close points in the embedding space. For this reason, distance prediction reaches higher accuracy. On the other side, IAPSE system provides an intermediate direct estimation of all the acoustic indicators, that can be easily interpreted by an analyst.

Table 2.3: *ρ values for IAPSE and DAPSE methods.*

	DRR	**T60**	**SNR**	**C50**	**C80**
IAPSE	0.706	0.837	0.964	0.785	0.784
DAPSE	0.735	0.871	0.979	0.834	0.840

In Figure 2.6, we present the performances of the two systems showing ρ for different SNR values. We can observe that, for both methods, the approximation accuracy increases for higher SNR values. We can also observe that DAPSE method outperforms IAPSE one for any SNR value, showing a good prediction robustness even when dealing with noisy recordings.

2.2.6 Conclusions

In this section we presented two data-driven methods to estimate acoustic parameter similarity between two speech recordings. Both methods are based on CNN architectures fed with a time-frequency representation of the audio signal. The first method preliminary estimates a set of acoustic parameters on which a distance measure is defined. The second method learns an embedding space where the distance is highly correlated to the

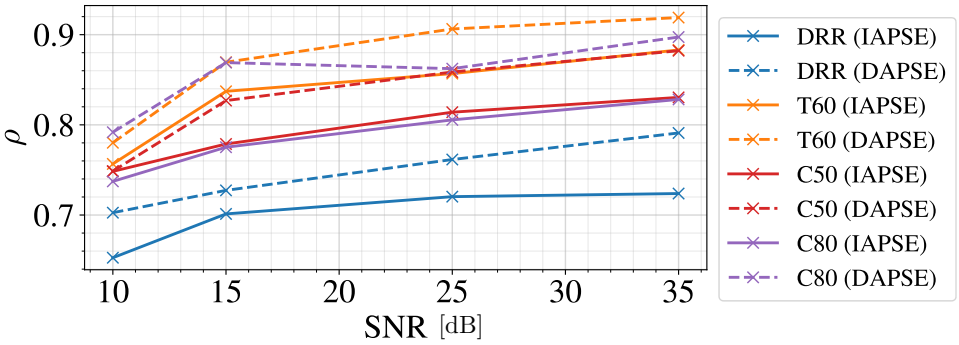


Figure 2.6: *ρ values for IAPSE and DAPSE methods at different SNR values for acoustic parameter similarity estimation.*

acoustic parameter similarity.

We evaluated both methods using a large dataset of reverberant noisy speech signals. The second method outperforms the first one in terms of Pearson correlation coefficient in the acoustic parameter similarity estimation task. The first method, on the other side, reaches performances comparable with the state of the art in the approximation of the acoustic parameter. Moreover, it offers an easier interpretation of the acoustic conditions of the analysed setups.

2.3 Intelligibility Estimation for STT Systems

Thanks to the recent technological advances, audio recording systems are becoming increasingly smaller and cheaper despite the high quality they can guarantee [101, 172]. This is good news for forensic investigators, as the ability of monitoring environmental, digital, or phone communications has become an urgent necessity for national security. Indeed, being able to deploy effective audio surveillance systems tailored to speech recording is of paramount importance to assist law enforcement agencies in foiling terrorist attacks or revealing harmful intents [98, 162].

On one hand, due to the availability of this technology, information gathering has become easier and ubiquitous. As far as environmental monitoring is concerned, audio surveillance devices are getting easier to deploy also thanks to the rise of wireless sensor networks [11, 118]. Therefore, the number of control spots in a scene can be further increased, ensuring higher

coverage of large public spaces.

On the other hand, blind and massive data collection can result in huge databases whose manual inspection might become infeasible [57,155]. Tools like automatic transcription agents would be of great help to quickly analyze databases for both investigative and surveillance purpose. However, given the diversity of contexts in which data is collected, audio excerpts are often corrupted by several types of noise (e.g., other speakers, ambient noise, reverberations, etc.). This can strongly degrade the quality of the recordings and compromise the intelligibility and transcription reliability of a possible relevant conversation.

For these reasons, it is important to develop automatic and intelligent methods that allow to speed up the analysis of these huge corpora of recorded data, but at the same time take into account the variety of the involved devices and environments [127, 153]. Depending on the characteristics of the collected audio excerpts, these systems should be able to extract relevant information speeding up the analysis of a human user.

In this work, we perform one step forward to meet these needs by proposing a framework to evaluate the reliability of noisy speech recordings tailored to automatic transcription. In particular, we estimate the likelihood of obtaining reliable automatic transcripts through speech-to-text engines, based on the analysis of recorded audio signals. Despite in the literature many full-reference [48, 97, 149] and no-reference [42, 81, 142, 143] algorithms to estimate speech intelligibility have been proposed, we specifically focus on the ability of obtaining valid transcriptions.

Our method can be employed in different application scenarios. As an example, this system can be a useful tool in managing data already blindly acquired by audio surveillance systems at scale. The model provides a qualitative feedback to investigators who can then decide to focus their attention only on reliable portions of the whole corpora that can be correctly transcribed. Alternatively, it can help an investigator in quickly understanding how to physically deploy the acquisition system within an environment characterized by a given kind of noise. Indeed, through a quick set of recordings at setup time, the analyst can obtain an indicator of how reliable the actual recordings will be for the investigation.

The proposed solution is based on a data-driven approach, and it is com-

posed by two main blocks. In the first one, a suitable set of features is extracted from the recorded audio signal. These features provide a numerical description of the fundamental characteristics of the signal under analysis. Then, a regressor (or a classifier) is designed and trained to predict the reliability level (or a discrete label) of noisy speech audio signals.

The implemented model has been trained and tested on a large dataset of transcribed speech signals corrupted by several types of noises at different SNR levels. The achieved results show the effectiveness of the proposed model in estimating how reliable a speech audio recording is. A preliminary test also shows that the method can generalize to recordings affected by previously unseen perturbations.

2.3.1 Problem formulation

Let us consider a speech audio excerpt $x(t)$ recorded in a noisy environment. Let us define it as

$$x(t) = s(t) * h(t) + n(t) = y(t) + n(t), \tag{2.6}$$

where $s(t)$ is the dry speech signal, $n(t)$ is an additive noise term, $h(t)$ is the RIR between source and receiver and $y(t)$ is the reverberant speech signal.

Our goal is to estimate a reliability score that quantifies how much the noise term introduced by the disturbed environment has compromised the possibility of correctly understanding the speech through an automatic speech-to-text transcriptor. Formally, we want to either estimate a binary score $\hat{\eta}_{\text{bool}} \in \mathbb{N}^{[0,1]}$ (i.e., classification problem) or a real score $\hat{\eta}_{\text{real}} \in \mathbb{R}^{[0,1]}$ (i.e., regression problem). A low score indicates that the speech is hard to be correctly automatically transcribed. A high score indicates that the speech can be easily automatically transcribed in a correct way.

Note that we are not interested in defining a classic speech intelligibility score, as our work is tailored to the performance of the specific transcription engine used by the analyst. Nonetheless, in the next subsection we introduce some background on generic speech intelligibility scores available in the literature.

2.3.2 Background

Speech intelligibility estimation has always drawn attention in the signal processing community due to its wide applicability in many research areas, from the study of audio communication standard to the development of speech enhancement techniques.

The first historically proposed indexes are the Speech Transmission Index (STI) [149] and the Articulation Index (AI), firstly defined in [48, 97] and later refined in the Speech Intelligibility Index (SII). These methods have been the foundations for all the solutions developed in the following years, but their effectiveness is valid under specific strict conditions, like presence of additive stationary noise only. To overcome this issue, a number of extensions has been proposed, like the Extended Speech Intelligibility Index (ESII) [134], the Coherence Speech Intelligibility Index (CSII) [84] and the Speech-based Speech Intelligibility Index (Speech SII) [54]. Alternatively, a different approach has been proposed in [152], where the authors defined the Short-Time Objective Intelligibility (STOI) measure that is able to deal with reverberation as well. STOI has proven to be effective in several different contexts, like telecommunications [75] or hearing aid systems [41].

All these methods have a common drawback: they all require the knowledge of both the degraded signal and the clean one. For this reason they are called intrusive (or full-reference) methods and they are not useful if an estimate of intelligibility is needed in contexts for which the reference signal is not available.

For this reason, non-intrusive (or no-reference) intelligibility estimation methods for which the knowledge of the clean reference signal is not required have been proposed in the literature. As an example, in [42], a speech to reverberation modulation energy ratio measure is proposed for estimating intelligibility of reverberated speech. This methodology is effective only when the degradation is given by reverberating environment. Other recent works have tried to apply data-driven approaches through machine learning algorithms to tackle the problem. For example, the Low Complexity Intelligibility Assessment (LCIA) methods [142, 143] extract a set of features from noisy audio signals and use a tree-regression or classi-

fication model to approximate intrusive measures of intelligibility (specifically STOI) and perceptual quality. Alternatively, in [81], a Hidden Markov Model (HMM) is used to extract relevant features of the clean signal from the noisy ones. In this case the goal is not to directly estimate an intelligibility index, but to synthesize the reference signal features to be used in any intrusive intelligibility estimation method.

In this work, we propose a data-driven approach whose goal is to extract from noisy monaural signal a score that predicts how much environmental noise affects the performances of an automatic transcription agent. Despite the methodology is similar to non-intrusive intelligibility estimation methods, the intent is different. Indeed, we want to provide a framework specifically tailored to audio forensics applications embedding an audio transcriber, hence we are not interested in predicting a subjective quality score. Our system should be flexible enough to be used in very different contexts with different types of degradation. Moreover, since it can be used by investigators during preliminary inspections of environments for preparing a bugging system, the computational cost and run time should be small enough for fast execution on portable devices.

2.3.3 Method

In order to evaluate the transcription reliability of a speech audio excerpt, we resort to a data-driven approach as shown in Figure 2.7. Specifically, we objectively define the reliability score based on audio transcripts. We extract feature vectors from the audio excerpts under analysis. We finally train either a regression or classification model using the extracted features and scores as labels. In the following we provide a detailed description of each step of this pipeline.

Objective reliability score

Let us consider a speech audio excerpt $x(t)$, whose ground truth transcript is $\mathcal{T}$, i.e., the set of words pronounced by the speaker. We define the speech-to-text engine as the function that transcribes an input audio excerpt $x(t)$ as

$$\hat{\mathcal{T}} = \mathrm{STT}(x(t)), \tag{2.7}$$

where $\hat{\mathcal{T}}$ is the estimated transcript. Given the dictionary (or collection of

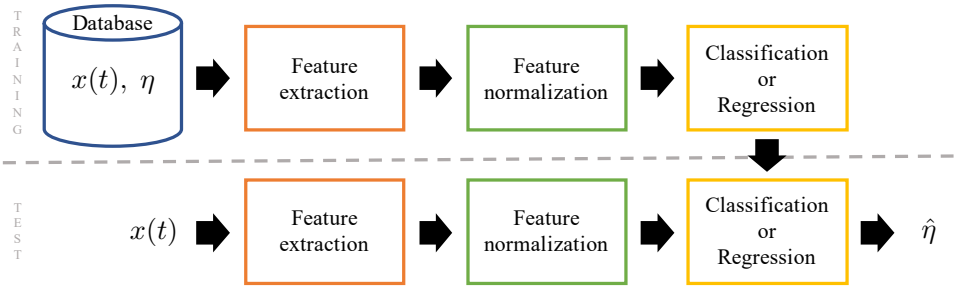


Figure 2.7: *Pipeline of the proposed transcription reliability estimation method, split into training and test phases. From each excerpt $x(t)$, features are extracted, normalized, and fed to a classifier or regressor. During training, ground truth reliability scores η are also used. During test, the reliability score $\hat{\eta}$ is predicted.*

terms) of the considered language $\mathcal{D}$, we consider that both $\mathcal{T} \subset \mathcal{D}$ and $\hat{\mathcal{T}} \subset \mathcal{D}$ hold.

In our context, a good reliability score should measure how similar the transcripts $\mathcal{T}$ and $\hat{\mathcal{T}}$ are. In the literature, many techniques have been proposed for this task [68]. In this work, we resort to Jaccard Similarity defined as the ratio between the cardinality of the union and the cardinality of the intersection of the two considered sets. Formally,

$$\eta = \frac{|\mathcal{T} \cap \hat{\mathcal{T}}|}{|\mathcal{T} \cup \hat{\mathcal{T}}|}, \tag{2.8}$$

where $|\cdot|$ is the cardinality of a set, $\mathcal{T}$ os the ground truth transcription and $\hat{\mathcal{T}}$ is the estimated transcription. Note that the score η assumes values ranging from 0 to 1. The value 0 is assumed if the two sets $\mathcal{T}$ and $\hat{\mathcal{T}}$ are disjoint (i.e., the transcriber could not extract a single correct word from the recording). The value 1 is assumed if the two sets $\mathcal{T}$ and $\hat{\mathcal{T}}$ are coincident (i.e., the transcriber was able to fully transcribe each pronounced word).

As a final remark, the selected score does not consider errors due to wrong ordering of the detected words (i.e., two sentences with swapped words are considered equal). This could be an issue if very short sentences would be considered. However, if we work with meaningful audio excerpts of a few seconds, it is very unlikely that the transcriber understands the correct set of words, but in a different order. Therefore, we can reasonably consider this issue as negligible.

Feature extraction

Given an audio excerpt $x(t)$ and a L_w long sample causal window $w(t)$, we define the i-th window of the signal as

$$x_i(t) = x(t) \cdot w(t - iL_w). \tag{2.9}$$

From each window $x_i(t)$, we extract a feature vector

$$\mathbf{f}_i = \mathrm{FEAT}(x_i(t)), \tag{2.10}$$

where $\mathrm{FEAT}(\cdot)$ computes Mel-Frequency Cepstral Coefficients (MFCC), Zero Crossing Rate (ZCR), Root Mean Square (RMS), Spectral Centroid (SC), Spectral Flatness (SF) and Spectral Roll-Off (SRF) as defined in [109]. This rich set of descriptors is very popular in mixed audio machine learning applications, especially for speaker and speech recognition [115]. The vector $\mathbf{f}_i$ is thus composed by 27 elements. Notice that the length L_w of the window must be chosen correctly in order to extract relevant features from a speech signal.

Given a series of feature vectors in time (i.e., one per window), we compute the per-feature average, standard deviation, maximum and minimum value over groups of J vectors as

$$\mathbf{f}_j^{\mu} = \frac{1}{J} \sum_{i=jJ}^{(j+1)J-1} \mathbf{f}_i, \tag{2.11}$$

$$\mathbf{f}_j^{\sigma} = \sqrt{\frac{1}{J} \sum_{i=jJ}^{(j+1)J-1} (\mathbf{f}_i - \mathbf{f}_j^{\mu})^2}, \tag{2.12}$$

$$\mathbf{f}_j^{\mathrm{M}} = \max_{i \in \{jJ,\,(j+1)J-1\}} \mathbf{f}_i, \tag{2.13}$$

$$\mathbf{f}_j^{\mathrm{m}} = \min_{i \in \{jJ,\,(j+1)J-1\}} \mathbf{f}_i, \tag{2.14}$$

where all operations are performed element-wise on each entry of $\mathbf{f}_i$. This step is needed to extract temporal statistics from the extracted features, which provide additional value and robustness for the subsequent learning step.

Finally, we concatenate all the aggregated feature statistics in a single feature vector as

$$\mathbf{f}_j^{\mathrm{tot}} = [\mathbf{f}_j^{\mu},\ \mathbf{f}_j^{\sigma},\ \mathbf{f}_j^{\mathrm{M}},\ \mathbf{f}_j^{\mathrm{m}}]. \tag{2.15}$$

The total length of the feature vector is of $27 \times 4 = 108$ elements. This vector is used as compact representation of a burst of $J \times L_w$ samples of $x(t)$.

Learning step

Once we extract a feature vector $\mathbf{f}_j^{\text{tot}}$ from an audio excerpt, we make use of either a classifier or regressor to predict the desired score $\hat{\eta}_{\text{bool}}$ or $\hat{\eta}_{\text{real}}$, respectively.

At training time, we select a training set of audio recordings whose transcript is known. From each track, we compute the score η according to Equation (2.8) and the features according to Equation (2.15). We then normalize each feature vector using either z-score (i.e., we subtract the mean from each feature and divide by its standard deviation) or min-max procedure (i.e., we scale each feature to span the range $[0, 1]$).

If regression is used, we simply feed the pairs of features and scores to the regressor. If binary classification is used, we first discretize the scores in two classes (i.e., $\eta \geq 0.5$ and $\eta < 0.5$), then we feed the pairs of features and scores to the classifier.

At test time, we extract the feature vector from the audio track under analysis, we normalize it, and feed to either the classifier ot regressor to predict $\hat{\eta}_{\text{bool}}$ or $\hat{\eta}_{\text{real}}$, respectively.

2.3.4 Dataset

To create the speech recording dataset used to test our framework, we started from LibriSpeech corpus [120]. LibriSpeech is a dataset of read speech extracted from English audiobooks from several different male and female readers. It includes 1000 hours of speech recordings sampled at 16 kHz, split into chunks that measures from 10 to 90 seconds approximately. All relative transcriptions are also available. The dataset is divided in three sets: training, development and test. The authors of this dataset guarantee that each speaker is included in only one of these three partitions and that male and female speaker presence is balanced in each partition. Unfortunately, we have no control over the distribution of the texts used over each dataset and, for instance, repetitions of book excerpts with different speakers may be present. These are are further divided in two subsets,

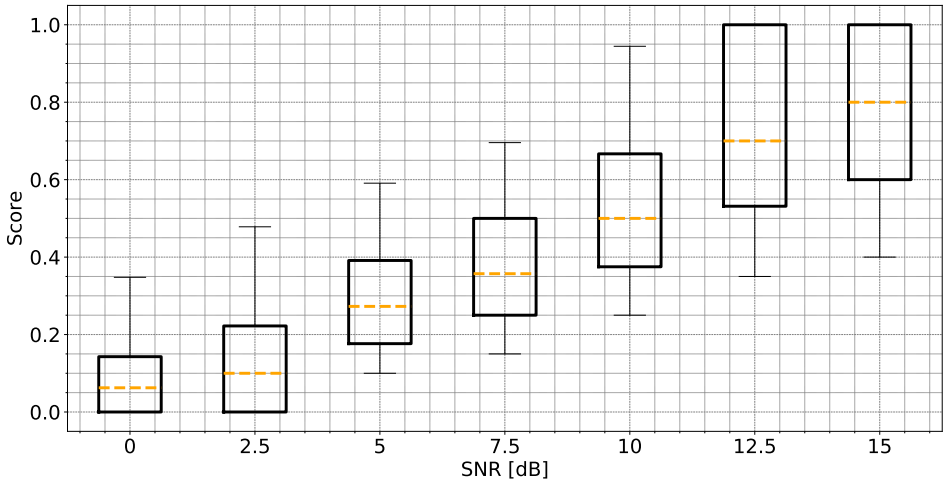


Figure 2.8: *Boxplot representing the distribution of ground truth scores η for different SNR values for the proposed transcription reliability estimation method.*

corresponding to clean and noisy audio.

We merged into a single set all the training, development and test clean files, since we want to have complete control over the noise conditions. For each clean recording, we estimated the transcription using a speech-to-text agent. We then selected only the audio files that produced a score $\eta = 1$, i.e., perfectly interpretable and intelligible.

In order to generate less interpretable recordings, we enriched the dataset by adding 10 different noises from everyday life at 7 different SNR levels to each selected audio track. The noise realizations are: traffic noise; crowd noise; hall noise; restaurant noise; train noise; industrial noise; three different speakers to create a cocktail party effect. The 7 SNR values range between 0 and 15 dB and equally spaced. Figure 2.8 shows the distribution of ground truth scores for each considered SNR. It is possible to notice that all η score values ranging from 0 to 1 are represented.

2.3.5 Experimental setup

In our experiments, we selected as $w(t)$ a Hanning window of length $L_w = 2048$ samples, which corresponds to 128 ms on the considered dataset at sampling frequency $F_s = 16000$ Hz. This parameter is selected considering the minimum duration of words in English [8]. The parameter J, which

corresponds to the number of windows on which feature statistics are computed, differs from test to test and ranges from a minimum value of 10, up to the value that covers the whole length of a recording. Feature extraction has been implemented through LibROSA [109] using default values if not otherwise specified.

As speech-to-text engine we relied on PocketSphinx [69]. This choice has been driven by four main motivations: it is a lightweight transcriptor that can easily run on low-power devices; it runs very quickly, thus making the analysis of huge corpora of recordings possible; its implementation is open and free; it does not constrain the user to a limited amount of transcriptions unlike other third-party engines (e.g., Google, Amazon, IBM, Microsoft, etc.).

Considering the learning part, we used as classifiers a Support Vector Machine (SVM) with radial basis function kernel, and a Random Forest Classifier (RFC). As mentioned above, in the classification setup the score η values are discretized over two classes (i.e., $\eta \geq 0.5$ and $\eta < 0.5$). As regressors, we used a Support Vector Regressor (SVR) with radial basis function kernel, and a Random Forest Regressor (RFR). Everything has been implemented using Scikit-learn Python library [126] using default parameters.

Experiments report the average result obtained performing 5-fold cross-validation on the prepared dataset. Specifically, for classification we rely on balanced accuracy (to take into account possible unbalanced classes) and F1 score, whereas for regression we rely on R2 score.

2.3.6 Numerical analysis of results

In the following, we report all performed experiments, from the ones conducted on our synthetically corrupted data, to those performed on a use case.

Classification and regression

In this experiment we set J equal to the maximum possible value in order to obtain a single feature vector for each different recording. Therefore, in this experiment, being L the total length of the input $x(t)$, $J = \lfloor L/L_w \rfloor$. We then trained and tested each classifier and regressor on the whole dataset

Table 2.4: *Classification results in terms of accuracy and F1 score using different classifiers and feature normalization techniques for the transcription reliability estimation method.*

Algorithm	Normalization	Accuracy score	F1 score
SVM	min-max	0.7819	0.7183
	z-score	0.8754	0.8388
RFC	min-max	0.8434	0.8030
	z-score	0.8434	0.8030

Table 2.5: *Regression results in terms of R2 score with different regressors and feature normalization techniques for the transcription reliability estimation method.*

Algorithm	Normalization	R2 score
SVR	min-max	0.5857
	z-score	0.8268
RFR	min-max	0.7482
	z-score	0.7482

according to the aforementioned 5-fold cross-validation setup.

Table 2.4 reports the results obtained in terms of balanced accuracy and F1 score obtained with the different classifiers and feature normalization techniques. It is possible to notice that RFC results correctly do not change depending on the normalization technique as expected. The best results are obtained with SVM classifier applied to z-score-normalized features. In this scenario, more than 87% of the excerpts are correctly classified as possible to transcribe or not.

Table 2.5 reports regression results in terms of R2 scores using different regressors and feature normalization policies. Also in this case the best results are achieved by the SVR paired with z-score normalization. In particular, it is possible to achieve R2 score of 0.83, showing a good correlation between estimated and ground truth scores.

To provide the reader with a better insight on the regressor results, we report in Figure 2.9 a boxplot showing the distribution of the achieved regressor scores against the ground truth ones, only considering the best trained model. This was obtained by quantizing the ground truth scores to 10 possible output values for visualization purpose. On one hand, it is interesting

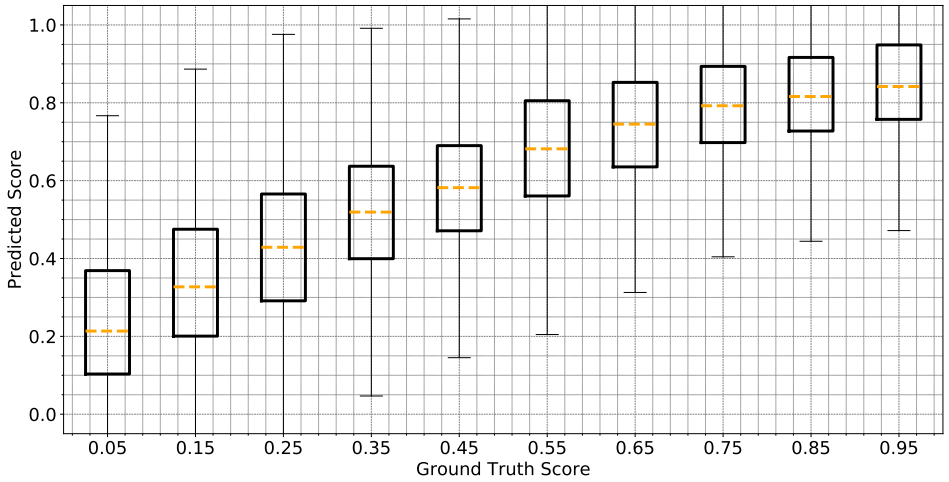


Figure 2.9: *Boxplot representing the distribution of estimated $\hat{\eta}_{real}$ against the ground truth scores η for the proposed transcription reliability estimation method.*

to notice the correct linear trend between real and estimated scores. On the other hand, this graph highlights one limitations of our regressor. As we did not constrain the predicted score $\hat{\eta}_{\text{real}}$ to lie in the range $[0, 1]$, we may obtain predictions that are outside this range. Therefore, despite the high R2 score (i.e., 0.83), the metric is surely penalized by this lack of constraints.

Moreover, the regressor seems to suffer from a scaling issue. The trend is correct, but the estimated values are slightly shifted from the ground truth ones. By knowing this systematic error, we may think about a possible post-processing solution to enhance even more the achieved performances.

Analysis window length J

We tested the effect of using smaller J values at test time. Please notice that J corresponds to the number of time windows on which feature vectors' statistics are computed and with this experiment we aim to evaluate how fine grained results can be obtained. As an example, this can be useful to an analyst that needs to extract very small excerpts. To this purpose, we considered the best classification model obtained so far trained using the largest possible J value, and we tested it against speech recordings analysed using $J \in [10, 12, 14, 16, 18, 20]$. Figure 2.10 reports the accuracy

and F1 score obtained for different J values according to two classification strategies:

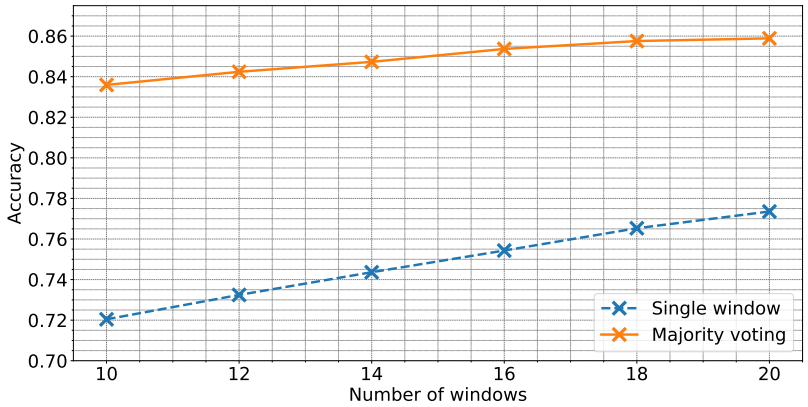
- *Single window*: results are reported computing evaluation metrics on a per-window base. This measures classifier performances in classifying excerpts of length $J \times T$ samples each.
- *Majority voting*: results are reported aggregating through majority voting classification results on all windows belonging the the same recording. This evaluates whether it is possible to analyze windows separately and then aggregate them, rather than using the complete recording, thus enabling real-time applications.

It is possible to notice that results obtained through majority voting are always better than the per-window ones as expected. Moreover, by aggregating even just 20 windows, it is possible to achieve an accuracy comparable to the one obtained with the largest possible J. This means that it is possible to achieve an accuracy of almost 86% on excerpts as short as 2.5 seconds.

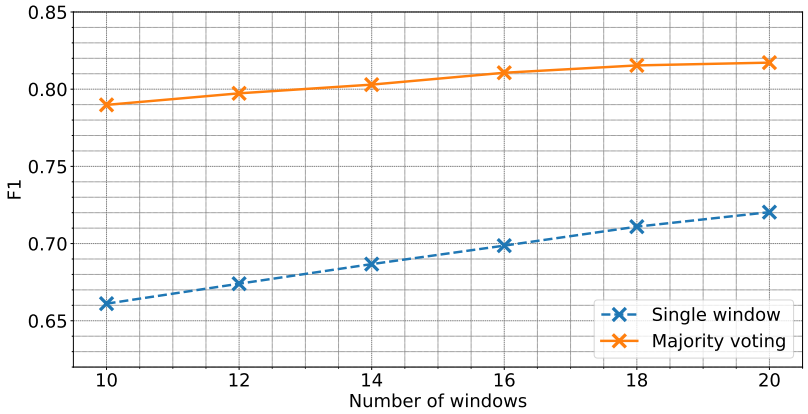
Case study

Finally, we tested the proposed system on an additional dataset obtained from a real case study. We collected a set of speech audio recordings acquired in a TV studio. We split the audio tracks into excerpts characterized by the presence of a single speaker or of multiple overlapped speakers. Recordings of the single speaker were completely intelligible, thus they could all be correctly transcribed (i.e., high η value). Conversely, all recordings containing multiple overlapped speakers could not be correctly transcribed due to the high mutual interference (i.e., low η value).

We tested the best model achieved so far trained on the ad-hoc created synthetic dataset on this new dataset. Interestingly, we were able to correctly estimate reliable speech excerpts with a balanced accuracy of 76% and F1 score of 78%. This is an interesting result considering that this case study dataset was acquired in a completely different setup. This opens the possibility of training the proposed system offline on synthetically corrupted data in all scenarios in which the environmental noise condition are known in advance.



(a) *Accuracy*



(b) *F1*

Figure 2.10: *Classification results in terms of accuracy and F1 score changing the number of windows J used at test time for the proposed transcription reliability estimation method.*

2.3.7 Conclusions

In this section we proposed a methodology to estimate whether a speech audio recording is reliable for automatic transcription. The goal is to help forensic investigators that need to analyse large corpora of environmental audio recordings that might be often corrupted by noise. The proposed procedure can then be used to quickly and automatically extract audio excerpts that are worth to be investigated due to their high intelligibility.

The proposed method exploits a supervised learning framework based

on feature extraction followed by classification or regression. Classification is used to simply predict a binary label (i.e., the audio recording can be correctly transcribed or not). Regression is used to provide a "soft" output value that better represent the likelihood of the recording to be correctly transcribed.

The presented solution has been tested on a corpus of speech recordings corrupted by different types of noise at different SNR levels. Finally, we have also shown the possibility of using the algorithm trained with synthetically corrupted data on real audio speech recording featuring multiple speakers. Future work will be devoted to study the effect of different text-to-speech engines, as well as to embed a denoising filter procedure within the transcription chain. Another possible development is the inclusion of multiple speech dataset at training stage to increase the robustness of the system.

2.4 Final remarks

In this Chapter we proposed two methods to automatically and blindly extract acoustic indicators from single-channel speech signals. In both cases, the problem is addressed combining signal processing and data driven methods. Moreover, the proposed acoustic indicators are not directly linked to specific acoustic behaviours but aim at describing the acoustic and noise conditions at a higher semantic level. Both estimation problems are mainly formulated as regression problems. In the first part of the chapter, the proposed indicator describes the similarity between two recording setups. To do so, two frameworks are implemented and evaluated, both exploiting a CNN for the feature extraction step. The first one define the similarity as a simple distance measure, while the second one exploit siamese networks to directly express the similarity measure in the embedding space. The evaluation proves that the second strategy reaches higher prediction accuracy, while the first one provides more a more interpretable similarity measure. The second method proposes a reliability measure of automatic transcriptions in noisy conditions, strictly related to speech intelligibility but tailored to the specific application. The goal is achieved through the extraction of a rich set of features used as input to regression supervised algorithm. Pre-

dictions from different time frames are fused with two different techniques. The system is tested with a dataset which includes speech recording corrupted by different type of noise. The overall performances are promising and they have been studied for different noise level and time granularity.

CHAPTER 3

Synthetic Speech Detection and Attribution for Authenticity Verification

In this chapter we analyse and propose solutions to the problem of synthetic speech detection and attribution.

Thanks to the constant development of new technologies and the massive evolution of neural networks, synthetic speech generation is nowadays an effortless operation and it is becoming increasingly difficult to distinguish the synthetic audio material from original one. While this opens the door to new challenging and stimulating scenarios, it can also lead to problematic situations.

For example, recently deepfakes have been used with malicious intents in several cases, especially in mass and social media. Some examples concern the spreading of fake news [64] and fraud cases [9], which led to some ethical considerations regarding the use of artificial intelligence [20].

Moreover, impersonation attacks may be dangerous also in every day

life scenarios. Often voice signal is used to assess the identity or control devices through speech human interfaces. The availability of synthesis techniques able to reproduce any voice put at risk the reliability of such systems. In the last few years, speech analysis research community has recognised the centrality of this problem. This has led to the organisation of challenges which specifically addressed the problem of automatic speaker verification in case of spoofing, or impersonation, attacks, like ASVSpoof 2019 [160] and ASVSpoof 2021 [177].

Simulated speech may be problematic also in a forensics scenario. Among other evidences, voice recordings, their transcriptions and the recording context may result crucial. It is easy to imagine a scenario where an audio evidence is maliciously forged to simulate, for example, a conversation that never happened. A forensic analyst needs tools able to verify the identity of the speaker and the authenticity of the recordings.

In this chapter we address the problem of synthetic speech detection and attribution. We define Synthetic Speech Detection (SSD) as the task of estimating whether a speech signal under analysis has been synthetically created or it is bonafide, i.e., real. On the other side, Synthetic Speech Attribution (SSA) is the problem of understanding which specific synthesis algorithm has been used to generate the fake speech samples.

Regarding SSD , we propose two different strategies, both adopting a data-driven approach. The first one exploits speech analysis and signal processing technique to define a set of features which are then fed to a ML classifier system. The goal is to detect traces in the signal left from the synthesis processing, by modelling speech signal as an auto-regressive process. This approach is suitable when training data is reduced and computational complexity needs to be low.

The second set of methods exploits high-level features. We aim at extracting from the speech signal high-semantic content and context, like the expressed emotion or the prosody style. This information may be used to detect falsified speech samples, exploiting the fact that also most recent synthesis techniques are not able to perfectly convey these aspects of human communication. To be able to describe such abstract information, we need to exploit the modelling potentialities of DL systems. Therefore larger training set are required to train such systems, compared to low-level ones.

Regarding synthetic speech attribution SSA, the task consists in estimating which specific algorithm has been used for the synthesis of the analysed fake speech signal. In this case we choose a supervised multiclass classification system, combined with low-level features. In particular, we present two possible scenarios, closed-set and open-set. In the first one it is assumed that during inference the set of possible classes, i.e., synthesis algorithms, is the same used during train. In the open-set scenario, the SSA system must be adapted to be able to classify correctly also new synthesis algorithms, using an additional unknown class. This second case is, of course, more realistic in a real-world not controlled scenario.

In this chapter, we first introduce the state of the art relative to both synthetic speech generation and synthetic speech detection. Then we focus on synthetic speech detection problem, presenting both low-level and high-level feature based approaches. In the final part of the chapter we address the problem of synthetic speech attribution, presenting a method for both closed and open set scenario.

3.1 Related Work

In this section we illustrate the state of the art relative to both synthetic speech generation and synthetic speech detection. The first part allows the reader to understand the different approaches and latest trends in the field of synthetic speech generation. In the second part, we investigate the literature regarding the problem in this chapter, i.e., synthetic speech detection. We also present a set of datasets that will help us in the evaluation experiments of the proposed methods.

3.1.1 Synthetic Speech Generation

Synthetic speech generation task, or speech synthesis, aims at creating automatically speech samples which sound natural and perfectly intelligible. It has several applications in everyday communication and it has been a crucial research topic in both natural language processing, speech processing and artificial intelligence community. In the literature we can find a large number of techniques that achieve natural sounding results, recently also thanks to the advances of neural networks architectures. We first present

TTS methods, and then review some recent Voice Conversion (VC) techniques.

The family of Text-To-Speech (TTS) methods start from a textual representation of the speech and aim at creating the correspondent waveform signal. In the past, TTS synthesis was largely based on concatenative waveform synthesis, i.e., given a text as input, the output audio is produced by selecting the correct diphone units from a large dataset of diphone waveforms and concatenating them so that intelligibility is ensured [15, 70, 114]. Additional post-processing steps allow to increase smoothness in transition between diphones, simulate human prosody and retain a good degree of naturalness [121]. The main drawback of concatenative synthesis is the need of huge recording databases and the difficulty of modifying the voice timbral characteristics, e.g., to change speaker or embed emotional and prosodic content in the voice.

To increase the variety and naturalness of generated speech, Statistical Parametric Speech Synthesis (SPSS) has been proposed. These methods avoid to directly generate the final waveform, but they aim at modelling first the sequence of acoustic features. Therefore, given an input text, these models first process it into a sequence of phonemes and other linguistic features (pauses, grammatical tags, ecc..). Then, an acoustic model is in charge of learning and predicting the mapping between linguistic features and acoustic features, like fundamental frequency, spectral envelope and excitation signal. The final step is a vocoder synthesizer, which is defined as a system able to transform a spectral representation of the audio in the raw waveform. Therefore, the final vocoder system transforms the acoustic features, derived from the textual input, into the final waveform. Historically, the selected acoustic model is an HMM, trained on large datasets of acoustic features extracted from diphones and triphones [107, 133, 161]. Also the choice of vocoder system, originally proposed in [37], contributes to the final quality of the synthesised voice. Examples of recent SPSS vocoders are STRAIGHT [85, 86], WORLD [113] and VOCAINE [2]. The simplicity of the SPSS approach allows to obtain good results at a reduced computational cost, suitable for real-time scenarios.

The advent of NNs has broke new ground for the generation of realistic and flexible synthesised voices. In particular, neural networks have

been firstly employed to replace only portions of the SPSS systems, like the acoustic models or the vocoder. Regarding the acoustic model, RNNs [166, 182] have substituted HMMs in sequence modelling. On the other side, traditional vocoders have been replaced with neural vocoders. Examples are WaveNet [144, 164], which predict samples of the waveform using convolutional layers in an auto-regressive setup, or LPCNet [163], which combines Linear Predictive Coding (LPC) analysis and RNNs to predict sample by sample a speech waveform.

To overcome the problem of synchronisation between the acoustic and linguistic features, first end-to-end models have been proposed. The increasing modelling potentialities of NN has allowed to use simpler linguistic features, like simple phoneme or characters, and more complex and less compact acoustic features, like mel-spectrograms.

One example is Tacotron [170], based on seq2seq [151] architecture and attention paradigm. The first version takes as input a sequence of characters and produces the corresponding raw spectrogram, which is then transformed in a waveform using the Griffin-Lim algorithm [55]. A second version, named Tacotron2 [144], improves the reconstruction of the waveform by predicting mel-spectrograms and using WaveNet as vocoder. This combination has allowed to greatly improve the quality of the speech signal, which sounds really natural if compared to the one produced with SPSS systems. During the years, several improved versions of Tacotron have been proposed, i.e., to convey specific prosody styles or emotions [146, 171].

Another example of end-to-end TTS systems are Deep Voice [5], which roughly follows the structure of SPSS systems, up to Deep Voice 3 [129], which proposes a fully convolutional network architecture.

These end-to-end speech synthesis architectures stand out with respect to classic methods in terms of timbre, prosody and general naturalness of the results, and further highlight the necessity of developing fake speech detection methods.

Another class of speech synthesis methods are the so-called Voice Conversion (VC) methods. In this case, a voice signal is manipulated such that the final target identity is different from the original one. Therefore, differently from TTS, the input is not text but a speech waveform. VC pipelines are usually split in three components [145]: speech analysis and

feature extraction, which transform the input speech signal into a suitable intermediate representation; feature mapping, which concretely applies the modifications necessary to match the target speaker; speech reconstruction, that re-construct the raw waveform from the modified feature maps. Each VC method combines different techniques and strategies for each pipeline's block. For the speech-analysis part, popular approaches in the past were based on Pitch Synchronous Overlap and Add (PSOLA) [6] or on the source-filter speech model, i.e., the intermediate representation corresponds to the set of parameters required by a vocoder synthesizer like STRAIGHT [85]. The use of vocoder parameters in analysis guarantees good quality in the final speech reconstruction, but it is not easy to adapt these parameters to match the target voice characteristics. For this reason, alternative spectral representations are often adopted, like mel-spectrograms or linear predictive spectral coefficients. About the mapping function, it can be learnt either using parallel training, i.e., on the pairs of utterances of original and target speaker with the same content, on with non-parallel training data. Parallel training methods can be performed in a parametric fashion, using Gaussian Mixture Model (GMM) [150], or adopting more recent NN architectures [34, 111]. Moreover, recently encoder-decoder architectures with attention mechanism has been proposed for allowing the network to implicitly learn the alignment between the input and the output [103, 158]. On the other side, non-parallel training of the mapping function is an exciting perspective for voice conversion applications, giving more flexibility on the choice of the training data corpora. Similarly to what has been done for image-to-image translation, preliminary solutions adopt Generative Adversarial Networks (GANs) architectures for the purpose [79, 80]. As mentioned, the final step of VC pipeline is speech reconstruction, which can be implemented using a vocoder system [85, 164], similarly to TTS methods.

3.1.2 Synthetic Speech Detection and Attribution

Detecting whether a speech recording belongs to a real person or is synthetically generated is far from being an easy task. Indeed, synthetic speeches can be generated through a wide variety of different methodologies, each one characterized by its peculiar aspects. For this reason, it is hard to find

a general forensic model that detect all possible synthetic speech methods. Moreover, due to the rise of deep learning solutions, new and better ways of generating fake speech tracks are proposed very frequently, as mentioned above. It is therefore also challenging to keep pace with the speech synthesis literature development.

Despite these difficulties, the forensic community has proposed a series of detectors to combat the spread of fake speech recordings.

Traditional approaches focus on extracting meaningful features from speech samples, able to discriminate between fake and real audio tracks. Specifically, the common belief in the community was that methods that choose effective and spoof-aware features usually outperform more complex classifiers. Moreover, long term features should be preferred with respect to short time features [78]. Examples are the Constant-Q Cepstral Coefficients (CQCC) [159], based on a perceptually inspired time-frequency analysis, magnitude-based features like Log Magnitude Spectrum or phase-based features like Group Delay [174]. Moreover, it has been noticed that traces of synthetic speech algorithms are distributed unevenly across the frequency bands. For this reason, sub-band analysis was exploited for SSD, presenting features like Linear-Frequency Cepstral Coefficients (LFCC) or MFCC [136]. In [72], the feature extraction step is based on a linear prediction analysis of the signals. These features are usually fed to simple supervised classifiers, often based on Gaussian Mixture Models. One of the most recently proposed methods to detect audio deepfakes based on hand-crafted features is [3], where the bicoherence matrix is used for the task and that which we consider as one of our baselines. Given the signal $x(t)$ under analysis, the authors compute the STFT of the input, obtaining $X(m, k)$, where m is the time window index and k is the frequency bin index. The bicoherence is then defined for each couple of frequency bin indexes k_1, k_2 as:

$$B(k_1, k_2) = \frac{\sum_{m=0}^{M-1} X(m, k_1)X(m, k_2)X^*(m, k_1 + k_2)}{\sqrt{\sum_{m=0}^{M-1} |X(m, k_1)X(m, k_2)|^2 \sum_{m=0}^{M-1} |X^*(m, k_1 + k_2)|^2}}. \tag{3.1}$$

Finally, the authors extract the first four moments of the bicoherence magnitude and phase and concatenate them in a feature vector which is fed to a

simple supervised classifier to distinguish whether a speech is synthetic or bonafide.

More recent methods explore deep learning approaches, inspired by the success of these strategies in speech synthesis as well as in other classification tasks. NN have been proposed both for feature learning and classification steps. For example, in [99] a time frequency representation of the speech signal is presented at the input of a shallow CNN architecture. A similar framework is tested in [183]. In this case the CNN is used solely for the feature learning step, whereas a RNN able to capture long terms dependencies is used as a classifier. In this case, several inputs have been tested, ranging from classic spectrograms to more complex novel features like Perceptual Minimum Variance Distortionless Response (PMVDR). The authors of [25] feed linear filter banks into a Resnet to generate embeddings used as input of a neural network classifier, and in [78] long-term features are used to discriminate fake and real audio tracks. Also end-to-end strategies have been proposed for spoofing detection [36]. These avoid any pre- or post-processing of the data and fuse the classification and feature learning step in a unique sleek process. An example of end-to-end spoofing detection systems is Rawnet2 [156]. This network works directly on the raw speech waveform, overcoming the classic back-end/feature extraction and front-end/classification structure. More specifically, the first layers corresponds to SincNet [132], a novel convolutional network that transforms the raw input with a band-pass filter bank for which the set of parameters is learnt during training. The following layers are three residual blocks, followed by a Gated Recurrent Unit (GRU) and a fully connected layer. This architecture has been proved to be successful not only for speaker verification, i.e., the original task for which it has been proposed, but also for synthetic speech detection. For this reason it has been proposed as baseline in the recent ASVSpoof 2021 challenge [177]. In the high-level feature based methods we will use Rawnet2 as a baseline for evaluating the performances of our methods.

3.1.3 Datasets

In this Section we present the datasets that we used for the evaluation setup of the presented methods. These datasets include speech samples produced

with TTS and VC algorithms and speech samples recorded from real speakers. The variety of synthetic speech generation algorithms allow us to test our algorithms in a real-world scenario.

ASVSpoof 2019

The most recent and complete dataset that we used in the following experiments is the ASVSpoof 2019 dataset described in [160, 169]. This dataset has been proposed to evaluate a wide variety of tasks related to speech verification, from spoofing detection to countermeasures to replay attacks. For this reason we only considered the part of the dataset consistent with the SSD problem considered in our work, defined as logical access dataset in [160].

This dataset is derived from the VCTK base corpus [176] that includes bonafide speech data captured from 107 native speakers of English with various accents (46 males, 61 females), and it is enriched with synthetic speech tracks obtained through 17 different methods. The data is partitioned into three separate sets: the training set $\mathcal{D}_{\text{ASV tr}}$; the validation set $\mathcal{D}_{\text{ASV dev}}$; the evaluation set $\mathcal{D}_{\text{ASV eval}}$. The three partitions are disjoint in terms of speakers, and the recording conditions for all source data are identical. The sampling frequency is equal to 16000Hz and the dataset is distributed in a lossless audio coding format.

The training set $\mathcal{D}_{\text{ASV tr}}$ contains bonafide speech from 20 (8 male, 12 female) subjects and synthetic speech generated from 6 methods (i.e., from A01 to A06 using the convention proposed in [169]). The development set $\mathcal{D}_{\text{ASV dev}}$ contains bonafide speech from 10 (4 male, 6 female) subjects and synthetic speech generated with the same 6 methods used in $\mathcal{D}_{\text{ASV tr}}$ (i.e., from A01 to A06). The evaluation set $\mathcal{D}_{\text{ASV eval}}$ contains bonafide speech from 48 (21 male, 27 female) speakers and synthetic speech generated from 13 methods (i.e., from A07 to A19). Notice that A16 and A19 actually coincide with A04 and A06, respectively. Therefore $\mathcal{D}_{\text{ASV eval}}$ only shares 2 synthetic speech generation methods with $\mathcal{D}_{\text{ASV tr}}$ and $\mathcal{D}_{\text{ASV dev}}$, whereas 11 methods are completely new. The complete breakdown of ASVSpoof2019 dataset is reported in Table 3.1.

The synthetic speech generation algorithms considered in this dataset

		$\mathcal{D}_{\text{ASV tr}}$	$\mathcal{D}_{\text{ASV dev}}$	$\mathcal{D}_{\text{ASV eval}}$
Samples	**Bonafide**	2580	2548	7355
	Synthetic	22800	22296	63882
Speakers	**Bonafide**	20	10	48
Synthetic Methods	**A01**	✓	✓	
	A02	✓	✓	
	A03	✓	✓	
	A04 = A16	✓	✓	✓
	A05	✓	✓	
	A06 = A19	✓	✓	✓
	A07			✓
	A08			✓
	A09			✓
	A10			✓
	A11			✓
	A12			✓
	A13			✓
	A14			✓
	A15			✓
	A17			✓
	A18			✓

Table 3.1: *Breakdown of the ASVSpoof2019 dataset showing the training, development and evaluation splits composition per number of samples, speakers, and synthesis methods.*

have different nature and characteristics. Indeed, some make use of vocoders, others of waveform concatenation, and many others of NN. In the following, a brief description of each one of them [169]:

A01 is a NN-based TTS system that uses a powerful neural waveform generator called WaveNet [164]. The WaveNet vocoder follows the recipe reported in [167].

A02 is a NN-based TTS system similar to A01 except that the WORLD vocoder [113] is used to generate waveforms rather than WaveNet.

A03 is a NN-based TTS system similar to A02 exploiting the open-source

TTS toolkit called Merlin [173].

A04 A waveform concatenation TTS system based on the MaryTTS platform [138].

A05 is a NN-based VC system that uses a Variational Auto-Encoder (VAE) [65] and WORLD vocoder for waveform generation.

A06 is a transfer-function-based VC system [108]. This method uses source-signal model to turn a speaker voice into another speaker voice. The signal is synthesized using a vocoder and overlap-and-add technique.

A07 is a NN-based TTS system. The waveform is synthesized using the WORLD vocoder, and it is then processed by WaveCycleGAN2 [157], a time-domain neural filter that makes the speech more natural-sounding.

A08 is a NN-based TTS system similar to A01. However, A08 uses a neural-source-filter waveform model [168], which is faster than WaveNet.

A09 is a NN-based TTS system [181] that uses Vocaine vocoder [2] to generate waveforms.

A10 is an end-to-end NN-based TTS system [73] that applies transfer learning from speaker verification to the neural TTS system Tacotron 2 [144]. The synthesis is performed through WaveRNN neural vocoder [77].

A11 is a neural TTS system that is the same as A10 except that it uses the Griffin-Lim algorithm [55] to generate waveforms.

A12 is a neural TTS system based on WaveNet.

A13 is a combined NN-based VC and TTS system that directly modifies the input waveform to obtain the output synthetic speech of a target speaker [92].

A14 is another combined VC and TTS system that uses the STRAIGHT vocoder [86] for waveform reconstruction.

A15 is another combined combined VC and TTS system similar to A14. However, A15 generate waveforms through speaker-dependent WaveNet vocoders rather than the STRAIGHT vocoder.

A16 is a waveform concatenation TTS system that uses the same algorithm as A04. However, A16 was built from a different training set than A04.

A17 is a NN-based VC system that uses the same VAE-based framework as A05. However, rather than using the WORLD vocoder, A17uses a generalized direct waveform modification method [92].

A18 is a non-parallel VC system [89] that uses a vocoder to generate speech from MFCCs.

A19 is a transfer-function-based VC system using the same algorithm as A06. However, A19 is built starting from a different training set than A06.

In the following we are going to use the notation $\mathcal{D}_{\text{ASV part alg}}$ to indicate the subset of the dataset corresponding to a specific algorithm, i.e., $\mathcal{D}_{\text{ASV tr A01}}$ corresponds to the training subset created using algorithm A01. This notation will be useful in the evaluation phase of our proposed methods. In fact, we are going to report the results not only on the complete test set but also on each dataset singularly and, when synthetic speech is present, on each algorithm.

Cloud2019

Cloud2019 is a dataset of synthetic speech samples originally introduced in [99]. It includes 11785 tracks generated using five different TTS cloud services: Amazon AWS Polly $\mathcal{D}_{\text{CL PO}}$, Google Cloud Standard $\mathcal{D}_{\text{CL GS}}$, Google Cloud WaveNet $\mathcal{D}_{\text{CL GW}}$, Microsoft Azure $\mathcal{D}_{\text{CL AZ}}$ and IBM Watson $\mathcal{D}_{\text{CL WA}}$ [45].

LibriSpeech

LibriSpeech is an open-source dataset, firstly presented in [120]. It contains about 1000 hours of speech recording, distributed at sampling frequency $F_s = 16000$ Hz. It is based on LibriVox, a collection of audio books freely available online. For this reason, each speech sample is associated to its transcription extracted from the corresponding book, by performing some

alignment operations, as described in [120]. The dataset is splitted in 3 partitions: training $\mathcal{D}_{\text{LS tr}}$, development $\mathcal{D}_{\text{LS dev}}$ and evaluation $\mathcal{D}_{\text{LS eval}}$ subset. Moreover, each subset is divided in two groups, "clean" and "other", depending on the matching score between the automatic transcription and the original text. Voice recordings have been performed by 2338 different speakers and each speaker appears in only one subset. Given the large dimensions of this dataset, in our experiments we considered only one part of the train clean subset and we are going to indicate it with $\mathcal{D}_{\text{LS}}$. The reader may notice this dataset has been used also in the evaluation setup for the method presented in Section 2.3.

LJSpeech

LJSpeech [87] contains audio clips of a single speaker reciting pieces from public domain non-fiction books. It counts 13100 audio clips of variable duration for a total length of 24 hours. We are going to refer to this dataset with $\mathcal{D}_{\text{LJ}}$.

IEMOCAP

IEMOCAP (Interactive Emotional dyadic MOtion CAPture database) [21] is a multi-modal dataset, which includes audio, video and motion capture recordings of 5 acting sessions. Each session is composed of segments of scripted and improvised dialogues performed by actors emphasizing a particular emotion. In our experiments we use obviously only the audio content, selecting tracks which are associated to a specific emotion class among anger, frustration, happiness, sadness and neutral. The total length is of approximately 12 hours. In the following we use only the improvised dialogues subset and we are going to indicate this dataset with $\mathcal{D}_{\text{IEM}}$.

3.2 Synthetic Speech Detection

In the following we present two different strategies to tackle the problem of synthetic speech detection.

We propose two data-driven strategies that take advantage of two different methodological approaches. In the first one we use a low-level feature

set, defined through a speech processing analysis technique. In the second one, on the other side, we employ recent NN networks to extract contextual and semantic information from the raw audio signal. The choice between one methodology or the other is driven by the availability of training data and computational power. Hand-crafted features allow us to operate in a reduced-data scenario, but may lack of generalisation ability. On the other side, high-level methods require higher efforts in terms of training data and time, but allow us to reach better results in not-controlled scenarios.

Nonetheless, both methods share the detection strategy, i.e. exploit inconsistencies present in the speech signal produced by synthesis algorithms, even though they operate in two different domains.

In the first part of this section we give a formal definition of the described problem. After that, we present the two SSD methodologies and experiments separately, given the just presented differences between the two approaches.

3.2.1 Problem formulation

Let us consider a speech signal $x(t)$ sampled at sampling frequency F_s. The speech signal is associated to a label

$$y \in [\mathrm{REAL}, \mathrm{DF}] \tag{3.2}$$

where the label REAL is associated to real, or bonafide, speech samples, while the label DF is associated to synthetic, or deep fake, speech samples.

Given an audio speech signal, the proposed frameworks aim at producing and estimate $\hat{y}$ of the ground-truth label y, i.e., whether the speech sample is pristine or it has been generated synthetically.

3.2.2 Low Level Feature Based Synthetic Speech Detection

In this Section we introduce a low-level feature based synthetic speech detection method.

In this first method we propose a set of hand-crafted features, inspired by the speech processing literature. In particular, we combine a series of features derived from a LPC analysis in order to capture traces from different kinds of synthetically generated speech tracks, By modelling speech as an auto-regressive process, we create a feature representation from the

residuals of both short-term and long-term analysis for different prediction orders. The use of classic signal processing knowledge helps overcoming the lack of training data, which on the contrary would affect SSD methods based on NN.

We first present the method, giving details about both back-end and front-end systems. We then report in detail the setup used for the evaluation phase. Finally we report the binary classification results obtained on a recent dataset of fake speech samples.

Method

In Figure 3.1 the pipeline of the proposed method is illustrated. Like most of the ML based systems, it is composed of two main steps. In the first one, a set of meaningful features are extracted from the raw audio signal. We propose a set of audio descriptors based on short term and long term analysis of the signal temporal evolution. Indeed, speech signals can be well modelled as processes with memory. It is therefore possible to extract salient information by studying the relationship between past and current audio samples. In the second step of the pipeline, a simple supervised classification algorithm learns to predict whether the speech sample is pristine or fake starting from the low-level feature representation.

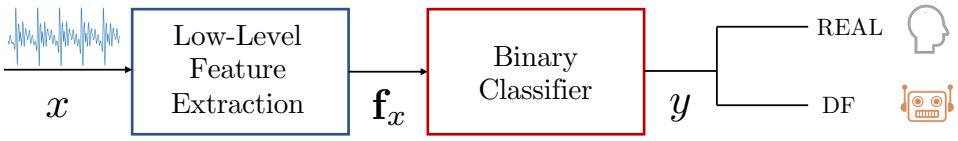


Figure 3.1: *Pipeline of the low-level feature based method.*

To correctly illustrate the feature extraction procedure, it is necessary to introduce the speech source-filter model. Speech is physically produced by an excitation emitted by the vocal folds that propagates through the vocal tract. This is mathematically well represented by the source-filter model that expresses speech as a source signal simulating the vocal folds, filtered by an all-poles filter approximating the effect of the vocal tract [43, 148].

Formally, the speech signal can be modelled as

$$x(t) = \sum_{i=1}^{L} a_i x(t-i) + e(t), \tag{3.3}$$

where $a_i,\ i = 1, ..., L$ are the coefficients of the all-poles filter, and $e(t)$ is the source excitation signal. This means that we can well estimate one sample of $x(t)$ with a L-order short-memory process (i.e., with a weighted sum of neighboring samples in time) as

$$\hat{x}(t) = \sum_{i=1}^{L} a_i x(t-i), \tag{3.4}$$

where the filter coefficients $a_i,\ i = 1, ..., L$ are also called short term prediction coefficients. By combining (3.3) and (3.4) it is possible to notice that the short term prediction residual $x(t) - \hat{x}(t)$ is exactly $e(t)$ if the model and predictor filter coefficients a_i are coincident.

For all voiced sounds (e.g., vowels), the excitation signal $e(t)$ is characterized by a periodicity of k samples, describing the voice fundamental pitch. It is therefore possible to model $e(t)$ as

$$e(t) = \beta_k e(t-k) + q(t), \tag{3.5}$$

where $k \in [k_{\text{min}}, k_{\text{max}}]$ is the fundamental pitch period ranging in a set of possible human pitches, β_k is a gain factor, and $q(t)$ is a wide-band noise component. According to this model, we can predict a sample of $e(t)$ with a long term predictor that looks at k samples back in time as

$$\hat{e}(t) = \beta_k e(t-k). \tag{3.6}$$

By combining (3.5) and (3.6) it is possible to notice that the long term prediction residual $e(t) - \hat{e}(t)$ is exactly $q(t)$ if the delay k and the gain β_k are correctly estimated.

According to this model, a speech signal can be well parameterized by the coefficients $a_i,\ i = 1, ..., L$ and the residual $e(t)$, which on its turn can be parameterized by β_k and the noisy residual $q(t)$. As already mentioned, several speech synthesis methods exploit this model. Even methods that do not explicitly exploit this model (e.g., CNN, RNN, etc.) generate a speech

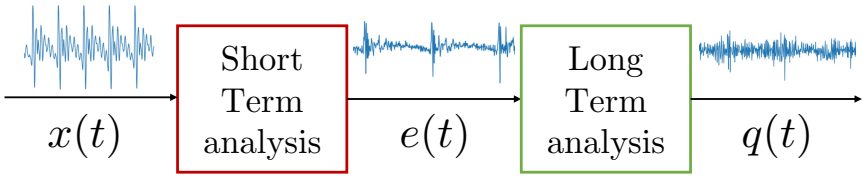


Figure 3.2: *STLT feature extraction for low-level feature based method.*

signal through operations in the temporal domain (e.g., temporal convolutions, recursion, etc.). It is therefore reasonable to expect that features within this model parameters domain capture salient information about the speech under analysis [72].

Motivated by the idea just illustrated, we propose a set of features based on the aforementioned set of parameters computed as follows. Given a speech signal under analysis $x(t)$ of length N, the feature extraction is divided in two steps, as shown in Figure 3.2.

In the short term analysis phase, prediction weights a_i, $i = 1, ..., L$ are estimated in order to minimize the energy of $e(t)$. Formally, this is achieved by minimizing the cost function

$$J_{\mathrm{ST}}(a_i) = \mathbf{E}[e^2(t)] = \mathbf{E}\left[\left(x(t) - \sum_{i=1}^{L} a_i x(t-i)\right)^2\right], \tag{3.7}$$

where $\mathbf{E}$ is the expected value operator. By imposing $\delta J_{\mathrm{ST}}/\delta a_i = 0$ for $i = 1, 2, ..., L$, we obtain a set of well-known equations at the base of linear predictive coding [148], i.e.,

$$r(m) - \sum_{i=1}^{L} a_i r(m-i) = 0,\ m = 1, 2, ..., L, \tag{3.8}$$

where $r(m)$ is the autocorrelation of the signal $x(t)$. By expressing (3.8) in matrix form, we obtain

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_L \end{bmatrix} = \begin{bmatrix} r(0) & r(-1) & \dots r(1-L) \\ r(1) & r(0) & \dots r(2-L) \\ \vdots & \vdots & \dots & \vdots \\ r(L) & r(L-1) & \dots & r(0) \end{bmatrix}^{-1} \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(L) \end{bmatrix} \tag{3.9}$$

or $\mathbf{a} = \mathbf{R}^{-1}\mathbf{r}$ where $\mathbf{a}$ is the coefficient vectors, $\mathbf{R}$ is the autocorrelation matrix and $\mathbf{r}$ is the autocorrelation vector. The inversion of $\mathbf{R}$ is usually performed using the Levinson-Durbin recursive algorithm [47]. Once the set of prediction coefficients are estimated, the short term prediction error $e(t)$ is obtained as

$$e(t) = x(t) - \sum_{i=1}^{L} a_i x(t-i). \tag{3.10}$$

Long term analysis aims at capturing long term correlations in the signal by estimating the two parameters k and β_k. As already mentioned, the delay k ranges between $k_{\min}$ and $k_{\max}$, determined by the lowest and the highest possible pitches of the human voice. The parameter k is obtained minimizing the energy of the long term prediction error $q(t)$. This is done by minimizing the cost function

$$J_{LT}(k) = \mathbf{E}[q^2(t)] = \mathbf{E}\left[(e(t) - \beta_k e(t-k))^2\right], \tag{3.11}$$

where β_k is approximated as $\beta_k = r(k)/r(0)$ [148]. As for the short time step, the long term prediction error $q(t)$ can be obtained as

$$q(t) = e(t) - \beta_k e(t-k). \tag{3.12}$$

In the proposed system we set $k_{\min} = 0.004\text{s}$, correspondent to a speech fundamental frequency of $f_0 = 250\text{Hz}$, $k_{\max} = 0.0125\text{s}$, correspondent to $f_0 = 80\text{Hz}$.

The features employed in the proposed method are directly derived from $e(t)$ and $q(t)$. In particular, we extract the prediction error energy (E) and prediction gain (G) for both short term (ST) and long term (LT) analysis,

defined as

$$\begin{aligned} E_{\mathrm{ST}} &= \frac{1}{N}\sum_{i=0}^{N-1} e(i)^2, \\ E_{\mathrm{LT}} &= \frac{1}{N}\sum_{i=0}^{N-1} q(i)^2, \\ G_{\mathrm{ST}} &= \frac{\frac{1}{N}\sum_{i=0}^{N-1} s(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2}, \\ G_{\mathrm{LT}} &= \frac{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} q(i)^2}. \end{aligned} \tag{3.13}$$

Rather than computing the prediction error energy and prediction gain on the whole signal as just described, the short term and long term analysis is applied to a speech signal segmented using rectangular windows. The quantities defined in (3.13) for each window w define the vectors

$$\begin{aligned} \mathbf{E}_{\mathrm{ST}} &= [E^0_{\mathrm{ST}},\, E^1_{\mathrm{ST}},\, \ldots,\, E^{W-1}_{\mathrm{ST}}], \\ \mathbf{E}_{\mathrm{LT}} &= [E^0_{\mathrm{LT}},\, E^1_{\mathrm{LT}},\, \ldots,\, E^{W-1}_{\mathrm{LT}}], \\ \mathbf{G}_{\mathrm{ST}} &= [G^0_{\mathrm{ST}},\, G^1_{\mathrm{ST}},\, \ldots,\, G^{W-1}_{\mathrm{ST}}], \\ \mathbf{G}_{\mathrm{LT}} &= [G^0_{\mathrm{LT}},\, G^1_{\mathrm{LT}},\, \ldots,\, G^{W-1}_{\mathrm{LT}}], \end{aligned} \tag{3.14}$$

where W is total number of windows. In the proposed method we used a boxcar window of length equal to $0.025\mathrm{ms}$.

To obtain a compact description for each speech signal, mean value, standard deviation, minimum value and maximum value across the windows are extracted, obtaining a vector

$$\begin{aligned} \mathbf{f} = [&\mu_{\mathbf{E}_{\mathrm{ST}}}, \sigma_{\mathbf{E}_{\mathrm{ST}}}, \max(\mathbf{E}_{\mathrm{ST}}), \min(\mathbf{E}_{\mathrm{ST}}), \\ &\mu_{\mathbf{E}_{\mathrm{LT}}}, \sigma_{\mathbf{E}_{\mathrm{LT}}}, \max(\mathbf{E}_{\mathrm{LT}}), \min(\mathbf{E}_{\mathrm{LT}}), \\ &\mu_{\mathbf{G}_{\mathrm{ST}}}, \sigma_{\mathbf{G}_{\mathrm{ST}}}, \max(\mathbf{G}_{\mathrm{ST}}), \min(\mathbf{G}_{\mathrm{ST}}), \\ &\mu_{\mathbf{G}_{\mathrm{LT}}}, \sigma_{\mathbf{G}_{\mathrm{LT}}}, \max(\mathbf{G}_{\mathrm{LT}}), \min(\mathbf{G}_{\mathrm{LT}})]. \end{aligned} \tag{3.15}$$

The entire procedure described up to this point assumes that a specific prediction order L is used. However, a good prediction order to be applied may change from signal to signal. Moreover, also this parameter L may be characteristic of some specific speech synthesis methods. For this reason,

the entire feature extraction procedure is repeated with different short time prediction orders $L \in L_{\min}, \ldots, L_{\max}$. Given the audio input $x(t)$, the resulting $\mathbf{f}_l$ feature vectors, where l is the considered order, are concatenated to obtain the final feature vector

$$\mathbf{f}_x^{\mathrm{STLT}} = [\mathbf{f}_{L_{\min}}, \mathbf{f}_{L_{\min+1}}, \ldots, \mathbf{f}_{L_{\max}}]. \tag{3.16}$$

In the proposed implementation $L_{\min} = 1$ and $L_{\max} = 50$, hence we obtain a feature vector of total length equal to $16 \times 50 = 800$ elements.

As already mentioned, during the classification step a supervised classifier is used to associate a label y to the feature vector $\mathbf{f}_x^{\mathrm{STLT}}$. Please note that in the proposed method we do not rely on a specific classification method since any supervised classification method can be used, i.e., SVM or RFC.

Experimental Setup

In this Section we report the technical details related to our experiments for SSD based on the presented low-level features.

Baseline

As baseline, we use another method based on hand-crafted features, i.e., bicoherence as presented in [3]. In particular, the authors propose as feature vector $\mathbf{f}_x^{\mathrm{BICOH}}$ defined as the first four statistical moments of magnitude and phase of bicoherence matrix B_x, as defined in Equation 3.1, computed on the audio signal input $x(t)$.

In the following experiments the bicoherence-based methodology is analysed in two scenarios. On one side, bicoherence method serves as a baseline to our proposed methodology. On the other side, we decided to combine bicoherence features with the newly proposed short-term and long-term analysis based features, to verify if this mix allows us to achieve good robustness and classification accuracy. This fusion strategy is implemented by simply concatenating the two feature vectors, obtaining

$$\mathbf{f}_x = [\mathbf{f}_x^{\mathrm{STLT}}, \mathbf{f}_x^{\mathrm{BICOH}}]. \tag{3.17}$$

Bicoherence is computed using a window length $L_w^{\mathrm{BICOH}} = 512$ and hop size $L_h^{\mathrm{BICOH}} = 256$, both in the baseline and fusion case.

Training

The proposed features can be used with any supervised classifier. In our experimental campaign we focus on simple and classical classifiers in order to study the amount of information captured by the proposed features. Specifically we use a RFC, a linear SVM and a Radial Basis Function (RBF) SVM.

In each experiment we always consider a training set used for training and parameters tuning, and a disjoint test set. Parameters tuning is performed by grid-searching the following set of parameters:

- RFC: the number of trees is searched in $[10, 100, 500, 1000]$; both Gini Index and Entropy split criteria are tested.
- Linear SVM: the margin parameter (often denoted as C) is searched in $[0.1, 1, 10, 100, 1000]$
- RBF SVM: same values of C for the linear SVM are searched. The RBF γ parameter, i.e. kernel coefficient, is searched in $[1, 0.1, 0.01]$.

In additional to the classifiers parameters, also different feature normalization techniques are used. In particular, we use min-max normalization (i.e., we scale features in the range from 0 to 1) and z-score normalization (i.e., we normalize the features to have zero mean and unitary standard deviation).

After all parameters have been selected based on grid-search on a small portion of the training set, results are always presented on the used test set. The implementation of all classification-related steps have been done through the Scikit-Learn [126] Python library.

Dataset

For this experiment we use ASVSpoof2019 dataset, previously described in Section 3.1.3. We split the train partition $\mathcal{D}_{\text{ASV tr}}$ in two subsets, using the 80% for the actual training stage and 20% for fine tuning. Then we test the method on both $\mathcal{D}_{\text{ASV dev}}$, which contains samples obtained using the same algorithms present in the training subset, and $\mathcal{D}_{\text{ASV eval}}$, which includes 13 new synthesis algorithm.

Results

In this section we collect and comment the results achieved through the performed experimental campaign for the low-level feature based method illustrated in Section 3.2.2. We first report an analysis that justifies the use of multiple prediction orders in the feature extraction procedure. Then, we report the results on SSD based on low-level features. Finally, we conclude the section with a preliminary experiment on encoded audio tracks.

Impact of the prediction order

As mentioned in Section 3.1.2, other methods proposed in the literature make use of the source-filter model to extract characteristic features [72]. However, these techniques typically exploit a single prediction order. Conversely, we propose to aggregate features computed considering multiple prediction orders.

To verify the effectiveness of our choice, we run an experiment considering the binary classification scenario while spanning multiple amounts of prediction orders ranging from 1 to 50. Let us define $\mathcal{L}$ as the set of used prediction orders such that $L \in \mathcal{L}$. This experiment can be interpreted as a feature selection step. In practice, we have iteratively trained and tested a RBF SVM, adding at each iteration the short-term and long-term features obtained from an additional order L.

Figure 3.3 reports the best accuracy obtained on $\mathcal{D}_{\text{ASV eval}}$ and $\mathcal{D}_{\text{ASV dev}}$ for each possible cardinality of $\mathcal{L}$. It is possible to notice that the use of a higher number of orders in the short-term analysis improves the detection ability of the system, enabling acceptable results also on $\mathcal{D}_{\text{ASV eval}}$.

Synthetic Speech Detection Results

In this experiment we report the performances of the main binary classification problem, i.e. the SSD task.

For this test we used $\mathcal{D}_{\text{ASV tr}}$ as training set. As features, we compared the baseline bicoherence-based ones [3] (Bicoherence), the proposed features (STLT), and the combination of both (STLT + Bicoherence). As bicoherence features can be computed with different window sizes affecting the resolution in the frequency domain, we tested windows of size 512, 256 and 128 samples with overlap half of the window length. For this reason

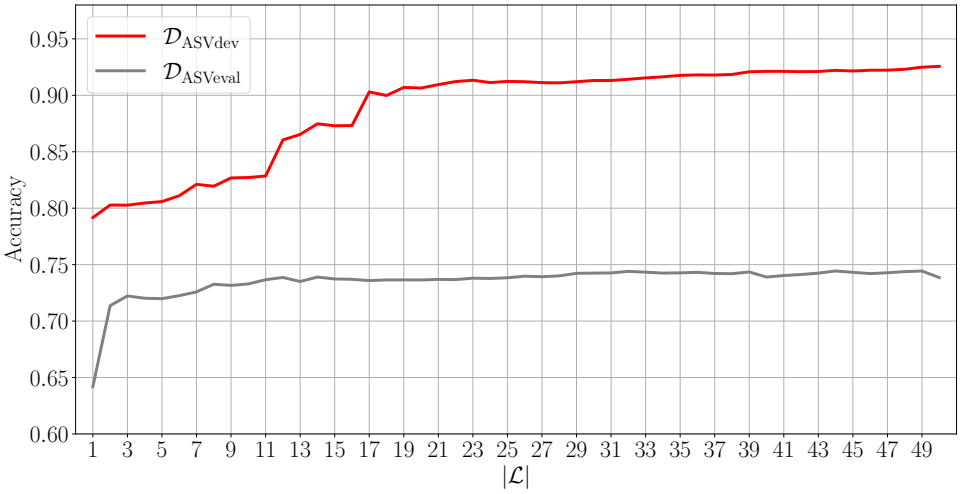


Figure 3.3: *Accuracy achieved on $\mathcal{D}_{ASV\,dev}$ and $\mathcal{D}_{ASV\,eval}$ for different cardinalities of $\mathcal{L}$ using the low-level feature based method.*

we have three different Bicoherence results, and three different STLT + Bicoherence results.

Table 3.2 shows the results achieved considering the best classifier and preprocessing combination for each feature set. In particular, we report the accuracy in detecting synthetic tracks depending on the used algorithm, as well as the average accuracy considering all synthetic algorithms together. It is possible to notice that classifiers based on Bicoherence alone perform reasonably, but are always outperformed by the proposed STLT. The best result is always achieved in the STLT + Bicoherence case, where windows have a 128 sample length. Specifically, it is possible to achieve an average accuracy of 0.94, and none of the synthetic speech generation is detected with accuracy lower than 0.91. It is interesting to notice that the best window length using only Bicoherence is larger than the one using Bicoherence + STLT features. Probably, short windows of Bicoherence do not capture enough temporal information to discriminate between bonafide and spoof samples, hence when only Bicoherence is used, a larger window length is more suitable. On the other side, when Bicoherence is combined with STLT features, the temporal traces and dependencies are properly and entirely described by the proposed STLT features, while Bicoherence on short windows captures finer information, that contributes positively to the final

classification accuracy.

	Bicoherence			**STLT**	**STLT + Bicoherence**		
	512	**256**	**128**		**512**	**256**	**128**
A01	0.615	0.526	0.570	0.929	0.917	0.919	**0.941**
A02	0.881	0.873	0.863	0.940	0.940	0.939	**0.946**
A03	0.859	0.846	0.847	0.952	0.948	0.950	**0.962**
A04	0.546	0.505	0.499	0.886	0.827	0.879	**0.915**
A05	0.805	0.801	0.778	0.946	0.943	0.945	**0.955**
A06	0.655	0.628	0.609	0.898	0.868	0.898	**0.932**
All	0.726	0.695	0.687	0.926	0.907	0.921	**0.942**

Table 3.2: *Bonafide vs. synthetic accuracy on dataset $\mathcal{D}_{ASV\ dev}$ for each synthetic speech algorithm using the low-level feature based method.*

Table 3.3 shows the same results breakdown when the trained classifiers are tested on the $\mathcal{D}_{\text{ASV eval}}$ dataset. This scenario is far more challenging, as only two synthetic methods used in training are also present in the test set (i.e., A04 and A06 being A16 and A19, respectively). All the other synthetic speech algorithms are completely new to the classifier. In this scenario, some algorithms are better recognized by the Bicoherence features, some by STLT, and some by STLT + Bicoherence fusion. On average, it is still possible to notice that STLT outperforms Bicoherence. The best results are obtained by the fusion STLT + Bicoherence, which provides an accuracy of 0.90 on known algorithms at training time, and 0.74 accuracy on average also considering unknown algorithms.

Concerning the choice of the classifier, the SVMs always outperforms the RFCs. The grid search has highlighted that RBF kernels are often more effective on Bicoherence methods, whereas STLT + Bicoherence and STLT methods work better with linear kernels.

To further analyse the results, we present also the ROC curve and correspondent AUC obtained on the $\mathcal{D}_{\text{ASV dev}}$. In this case we use the best parameters for both feature computation and classification. In Figure 3.4 each ROC curve corresponds to the three methods presented. We can confirm that the method using STLT features is able to reach very satisfactory performances, similarly to the method using both STLT + Bicoherence.

	Bicoherence			STLT	STLT + Bicoherence		
	512	**256**	**128**		**512**	**256**	**128**
A07	0.541	0.505	0.501	0.865	0.813	0.864	**0.905**
A08	0.693	0.627	0.591	0.951	**0.955**	0.955	0.954
A09	0.543	0.508	0.508	0.835	**0.882**	0.865	0.835
A10	**0.534**	0.516	0.504	0.511	0.492	0.487	0.493
A11	0.617	0.685	**0.762**	0.629	0.489	0.481	0.474
A12	**0.547**	0.524	0.511	0.509	0.504	0.498	0.487
A13	0.768	0.779	0.767	0.948	**0.955**	**0.955**	0.945
A14	0.718	0.708	0.726	0.882	**0.916**	0.906	0.880
A15	**0.567**	0.514	0.507	0.466	0.479	0.473	0.465
A16	0.544	0.516	0.509	0.872	0.833	0.871	**0.908**
A17	0.510	0.532	0.578	0.656	0.649	**0.660**	0.653
A18	0.515	0.534	0.537	**0.869**	0.849	0.843	0.849
A19	0.611	0.586	0.575	0.882	0.863	0.885	**0.906**
All	0.592	0.578	0.578	0.739	**0.741**	0.737	0.735

Table 3.3: *Bonafide vs. synthetic accuracy on dataset $\mathcal{D}_{ASV\ eval}$ for each synthetic speech algorithm using the low-level feature based method.*

In Figure 3.5 we present the same metrics on the $\mathcal{D}_{\text{ASV eval}}$ partition. Similarly to what observed in the analysis of accuracy values, on this dataset the fusion of STLT and Bicoherence features is even more effective and outperforms the other two methods, reaching a value of AUC of 0.79.

Preliminary test on encoded audio tracks

Nowadays, audio tracks are often shared through social media and instant messaging applications. This means that audio signals are customarily compressed using lossy standards. This is the case of Whatsapp, which makes use of Opus audio coding scheme.

In order to further assess the robustness of the proposed method on encoded audio tracks, we performed a preliminary simple experiment. We simulated Whatsapp audio sharing by encoding a random selection of 1000 audio tracks of $\mathcal{D}_{\text{ASV dev}}$ dataset using Opus codec with a bitrate compatible with Whatsapp. We tested the system trained on the original audio tracks in the binary configuration using as input the encoded audio files. The results we obtained are interesting and promising. Even tough the lossy coding op-

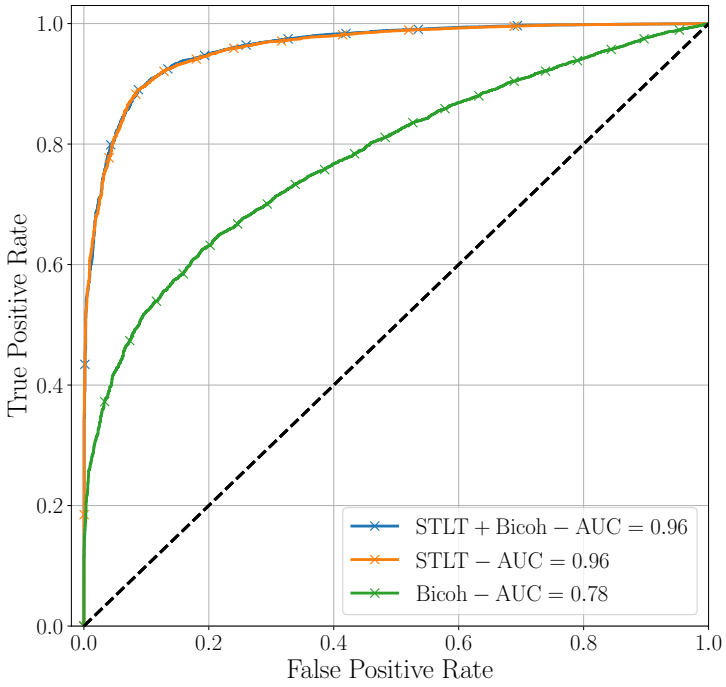


Figure 3.4: *ROC obtained on $\mathcal{D}_{ASV\ dev}$ for the three low-level feature based methods.*

eration has lowered the quality of the audio signals, the proposed system is able to discriminate the synthetic speech from the real speech signals with 79% accuracy. Despite these experiments are just preliminary, we believe they highlight an interesting future research path.

3.2.3 High Level Feature Based Synthetic Speech Detection

In this section we present two novel methods for SSD task, both based on NN architectures and using transfer learning approach.

In the previous section we presented a low-level feature based method, which combines meaningful hand-crafted features with a simple supervised classifier. This approach has shown to be helpful with limited training data and incorporates digital signal processing knowledge for the back-end feature extraction phase.

Nonetheless, in presence of larger dataset and higher computational power, it is worth exploring NN potentialities for SSD task. In particular, the goal is to design a feature space which is able to capture high-level semantic

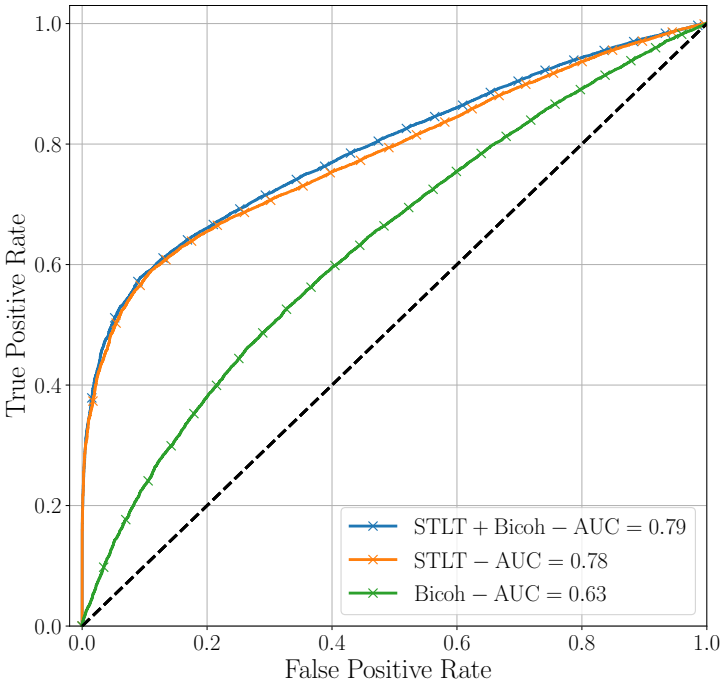


Figure 3.5: *ROC obtained on $\mathcal{D}_{ASV\ eval}$ for the three low-level feature based methods.*

information, exploiting NN modelling potentialities. If we are able to describe correctly the semantic context of the speech acquisition, we are also able to spot inconsistencies and therefore detect manipulated data.

The first method we propose is based on speech emotional cues and it is able to detect only TTS generated speech samples. The rationale behind this proposal is to exploit the fact that recent end-to-end TTS systems, even when reaching really high quality speech, are still not able to embed emotional content in the speech. We use a SER network to produce a compact representation of the emotional content present in natural speech. This feature set is then fed to a simple binary classifier, which predicts whether the input is a bonafide or spoof sample. We indicate this method using the name EmoSSD.

The second method expands and modifies the idea behind the first one, maintaining the same high level semantic approach. In this case we define two set of features, one related to the speaker identity and one related to prosody style. After concatenating the two representations, we are able to

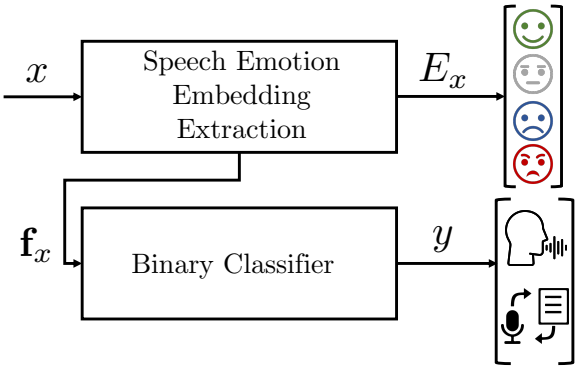


Figure 3.6: *Architecture of the proposed system based on emotional cues.*

detect fake speech using, again, a simple supervised classification system. The use of these two embeddings vectors allow to address the problem for both VC and TTS speech samples. This second method will be indicated as ProsospeakerSSD.

Both methods are based on the transfer learning technique. In fact, back-end feature extraction systems are derived from a NN originally trained for a different task, i.e., emotion recognition, speaker verification and prosody modelling. On the other side, the second step, i.e., front-end classification, is implemented using classic ML algorithms. We are not considering more complex classification schemes because we rely on the rich contextual information expressed in the feature representation.

In the following section we first present emotional cues-based SSD method and prosody and speaker cues-based SSD method. Then we give details about the experiments performed on both methods. Finally we present the results obtained in the evaluation stage.

Emotional Cues Based SSD (EmoSSD)

In this section we propose a method for SSD which exploits sentiment analysis, named EmoSSD. In particular, we use a novel transfer-learning method, using the semantic features extracted from a SER network as input of a deepfake classifier. The method is focused on the detection of TTS and mixture TTS/VC deepfakes, while does not take into account pure VC algorithms. This is because we exploit speech semantic information to de-

tect anomalies, and pure VC fakes do include such content, being generated from a real voice and then altered with style transfer techniques.

Figure 3.6 shows the pipeline of the proposed EmoSSD method. The process is composed of two blocks. The first block is a SER system that exploits the architecture recently proposed in [24]. Starting from an input speech signal x, it estimates the expressed emotion E_x. We exploit this architecture to define a set of features $\mathbf{f}_x$ able to describe the emotional content of input speech. The second block is the actual SSD system, i.e., a supervised classifier that associates a class y to the input features $\mathbf{f}_x$. In the following, we provide details about each block.

Speech Emotion Embedding Extraction

The first part of the proposed pipeline extracts a set of features $\mathbf{f}_x$ able to express the emotional content of the speech audio signal x under analysis. This choice is motivated by the fact that TTS deepfake algorithms reach excellent results in terms of speech naturalness but still fail in modeling the emotional properties of the human voice correctly. We can therefore exploit this weakness in combination with with neural networks' ability to create powerful and flexible embeddings, which should describe not only which emotion is present in the speech signal but also its intensity.

The considered emotional features are computed making use of the 3D-Convolutional Recurrent Neural Network (CRNN) proposed in [24]. The authors address the problem of speech emotion recognition as a classification problem using a categorical approach, i.e., N possible emotion classes are considered [28]. Therefore, given a speech utterance x, the output of the network is

$$E_x \in \{e_1, e_2, ..., e_N\}, \tag{3.18}$$

where e_i is the i-th emotion class (e.g., happy, sad, angry, etc.). As reported in [24], the input signal x must be pre-processed to be fed to the following neural network. We do so by computing the spectrum of x through an STFT in the mel-frequency domain and applying a logarithmic transform to the STFT magnitude. This returns a log-mel spectrogram defined as

$$\mathbf{S}_{\text{mel}} \in \mathbb{R}^{M \times K}, \tag{3.19}$$

where M is the number of windows and K is the number of mel bins. Then, we compute the first and second discrete derivatives of $\mathbf{S}_{\text{mel}}$ along its second

dimension (frequency axis), obtaining $\Delta\mathbf{S}_{\text{mel}}$ and $\Delta\Delta\mathbf{S}_{\text{mel}}$. By stacking the log-mel spectrogram and its derivatives along a third dimension, we obtain the final 3D matrix $\mathbf{X}$ defined as

$$\mathbf{X} = [\mathbf{S}_{\text{mel}}, \Delta\mathbf{S}_{\text{mel}}, \Delta\Delta\mathbf{S}_{\text{mel}}] \in \mathbb{R}^{M\times K\times 3}. \tag{3.20}$$

This matrix is then standardised by means of z-score normalization. The processed input is fed to a set of 3D convolutional layers, followed by a linear layer, a bidirectional LSTM layer and an attention layer. Finally, a sequence of dense layers outputs a probability measure of each emotion class, from which the prediction E_x is extracted. We refer to the complete paper for further details [24]. Adopting a transfer-learning strategy, we extract a feature vector $\mathbf{f}_x$ of dimensionality L from an intermediate network layer. Specifically we consider the output of the final attention layer, which the authors present as the *utterance-level emotional representation*. Formally, we can express the feature extraction block as a function $\mathcal{F}$ such that

$$\mathbf{f}_x = \mathcal{F}(x) \in \mathbb{R}^L. \tag{3.21}$$

As mentioned, the proposed feature vector does not simply have good discriminative power for its original task (i.e., estimating the *quality* of the emotion) but also for the SSD task (i.e., estimating the *intensity/quantity* of the emotions expressed).

Binary Classifier

In the second part of the proposed pipeline, a binary classifier takes as input the feature vector $\mathbf{f}_x$ and estimates the class y to which the input signal x belongs. It is worth noting that we can use any supervised classification method at this stage. However, since this work aims to explore the deepfake discriminatory power of the selected semantic features, we decided to use well-known classical classifiers. Among others supervised classification methods, our experiments show that a Random Forest Classifier is capable of discriminating between real and fake audio with high accuracy.

Prosody and Speaker Cues Based SSD (ProsospeakerSSD)

In this section we present a second SSD methodology, which, using the same high semantic based strategy, expands and improves the previous one.

In the following we refer to this method as ProsospeakerSSD method. In particular, we extract two set of high-semantic features that describe, on one side, the identity of the speaker and, on the other side, the prosody properties of the speech. By combining these two representations we are able to detect synthetic speech samples produced both with VC and TTS methods.

The work presented in [1] has partially inspired our proposed approach. In this work the goal is to detect deep-fake video created with face-swap technique, i.e., where only person's face is modified to match another identity. The authors exploit the fact that in a face-swap deep fake the facial behaviors are not affected and are the ones of the original individual, while the facial identity is modified to match the one of a different individual. By matching bio-metrics based on face recognition and expressions/head movements against a set of pristine reference videos, they are able to spot face-swap deep-fake videos.

We translate this approach in the speech domain by combining speaker identity and prosody speaker features to detect synthetic speech samples. Differently from [1], ProsospeakerSSD method does not require any reference set neither tries to match identity and prosody style. We instead exploit the discriminative potential of the two set of features taking into account the main weaknesses of generation algorithms, which we, as already mentioned, roughly divide in two categories, TTS and VC.

On one side, we select a set of prosodic features motivated by the fact that TTS systems fail in creating speech with convincing and natural prosody. To give a precise definition of prosody is not an easy task, therefore a subtractive definition is usually adopted, as in [146]: prosody is the speech variation that remains after considering the content, the speaker identity and the recording environment. Classic prosodic features are fundamental frequency statistics, voicing probability or loudness [40]. In this work, a different approach is adopted and therefore prosodic features are computed through NN-based prosody encoder [146]. We rely on the fact that such prosody embeddings can be helpful in describing not only the *quality* of the prosody, but also the *intensity* of it. It is worth noticing that in this work prosody features play a role similar to the one that emotional cues play in the EmoSSD proposal: they "measure" an intrinsic characteristic

of human voice which TTS engines struggle at recreate. One difference is that, while commonly TTS systems renounce to convey any emotion in the final speech, most recent TTS algorithms openly tries to re-create natural prosody. We rely on the fact that, despite the recent TTS advances, synthetic prosody has different quality and intensity w.r.t. the human one and it can be captured by the proposed prosodic embeddings.

On the other side, VC algorithms use as input a pristine speech sample and aim at modifying the identity of the speaker, i.e., the timbre. We believe that the modification process creates artifacts in the final voice qualities that are not audible but that can be traced using speaker identity embeddings.

These two sets of features are therefore orthogonal and allow our system to detect a broader set of synthesis algorithm. Also here we use a transfer learning approach, using two neural networks originally trained for a different task, i.e., speaker identification and prosody synthesis, as embedding extractor. The actual binary classification is performed with a simple supervised classifier. In Figure 3.7 the pipeline of the proposed system is schematised. It takes as input a speech signal x and two features vectors, $\mathbf{f}_x^{\text{PROS}}$ and $\mathbf{f}_x^{\text{SPKR}}$, are extracted. These two vectors are then concatenated and fed to a classifier, that gives as output a prediction of the label y. In the following we give details about each block of the pipeline.

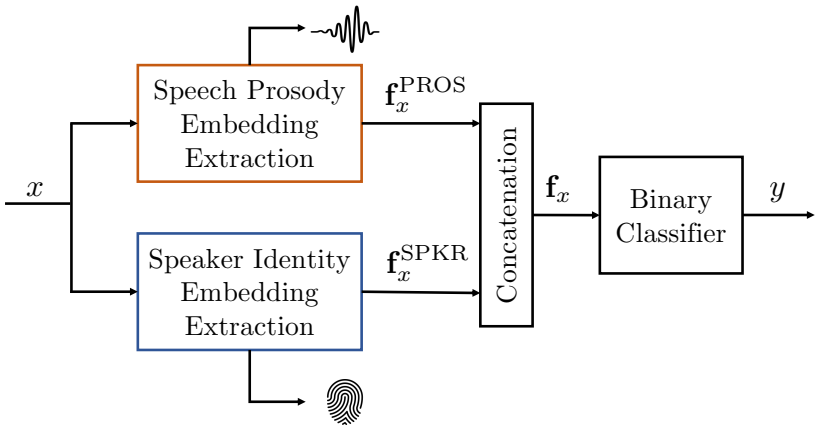


Figure 3.7: *Architecture of the proposed ProsospeakerSSD method.*

Speech Prosody Embedding Extraction

We extract prosody embeddings using the reference encoder of the model

presented in [146], which we will refer to as prosody encoder. Its original role in [146] is to improve the naturalness of the voices synthesized by Tacotron [170], augmenting them with explicit prosody controls. Tacotron receives a textual input and generates speech depending on the speaker identity used for training. In contrast, the encoder receives a reference signal conveying the desired prosody and extracts a fixed-length learned representation. This is used to condition the synthesis process, allowing higher expressiveness and recreating a specific prosody style. The final result matches the prosody of the reference signal with fine time detail even when the reference and synthesis speakers are different. The authors demonstrate that it allows transferring prosody between utterances in an almost speaker-independent fashion. The prosody encoder comprises a 6-layer stack of 2D convolutions with batch normalization, followed by a GRU layer to summarize the variable-length sequence. Finally, a fully-connected layer extracts the embeddings in the desired dimension. It is worth noting that no supervision is needed for training the prosody encoder. This design sufficiently compresses the input information, forcing the encoder to learn a compact representation of prosody. Tacotron and prosody encoder are jointly trained by synthesizing target audio, provided as input to both, and using reconstruction error as loss. We consider as prosody embeddings the output of the prosody encoder.

Therefore, given a speech signal x we first extract a time-frequency representation using STFT in the mel-frequency domain, obtaining

$$\mathbf{S}_{\text{mel}} \in \mathbb{R}^{M \times K} \tag{3.22}$$

where M is the number of time windows and K corresponds to the number of frequency bins. This pre-processed input is fed to the prosody encoder, which can be defined as a function $\mathcal{F}^{\text{PROS}}$ such that:

$$\mathbf{f}_x^{\text{PROS}} = \mathcal{F}^{\text{PROS}}(x) \in \mathbb{R}^{L_{\text{PROS}}}, \tag{3.23}$$

where L_{PROS} corresponds to the embedding dimensionality. This feature vector is able to describe the prosody characteristics of the analysed speech and we prove in the following experiment its potentialities in discriminating between real and fake speech samples, in particular when the employed synthesis method is a TTS system.

Speaker Identity Embedding Extraction

To extract the speaker embedding $\mathbf{f}_x^{\text{SPKR}}$ we rely on a state-of-the-art method originally proposed for speaker verification, i.e., ECAPA-Time Delay Neural Network (TDNN) model described in [35]. This architecture enhances the typical X-vectors architectures [147], widely used for speaker verification tasks. The original X-vector network maps variable length utterances in a fixed length speaker embedding vector, using a TDNN and a statistical pooling layer. Interestingly, a recent work [61] investigates which voice characteristic's variability affects more a x-vector based speaker recognition prediction score. This study can help us understanding roughly what voice characteristics are encoded in the x-vectors embedding (and derived speaker embeddings architectures). The study reveals that the most influential voice quality descriptors are harmonic-to-noise ratio, strictly related to the vocal tract characteristics [44], and spectral tilt.

ECAPA-TDNN, the adopted speaker embedding architecture, further elaborates the X-vector architecture by adding some components, like residual connections and squeeze-excitation blocks, to expand the temporal context and multi-layer feature aggregation. The authors prove that these additions allow the network to generalize better, capture high-level properties, and improve speaker recognition results, while significantly reducing the number of model parameters. More details about the network components may be found in [35].

This network takes as input a variable-length speech signal x and outputs an embedding $\mathbf{f}_x^{\text{SPKR}}$. The network requires a pre-processing step, i.e., speech signal is first transformed in a time-frequency representation and further processed to obtain a set of MFCC, i.e.,

$$\mathbf{MFCC}_x \in \mathbb{R}^{M \times B}, \tag{3.24}$$

where M is the number of time windows and B is the number of mel-frequency cepstrum coefficients.

This feature map is fed to one dilated convolutional layer, followed by three squeeze-excitation residual blocks. The outputs of these three blocks are concatenated and fed to a second dilated convolutional layer and one attention layer. Finally, the embedding is obtained as the output a fully-connected layer of dimensionality L_{SPKR}. Formally, we can define this em-

bedding extraction block as a function $\mathcal{F}_{\text{SPKR}}$ such that

$$\mathbf{f}_x^{\text{SPKR}} = \mathcal{F}_{\text{SPKR}}(x) \in \mathbb{R}^{L_{\text{SPKR}}} \tag{3.25}$$

where L_{SPKR} corresponds to the embedding dimensionality. This embedding vector is a compact description of the voice properties of a speaker and it is also able to capture the traces left by voice conversion processes, as it will be proved later.

Binary Classifier

The two feature vectors obtained in the just described blocks are fused in one single feature vector by applying concatenation, i.e.,

$$\mathbf{f}_x = [\mathbf{f}_x^{\text{SPKR}}, \mathbf{f}_x^{\text{PROS}}] \in \mathbb{R}^{L_{\text{SPKR}}+L_{\text{PROS}}}. \tag{3.26}$$

In the final part of the pipeline the final feature vector is fed to a simple binary classifier that learns to predict the binary label y, i.e., if the input speech x is pristine or synthetic. Even though any supervised classifier can be used at this stage, our experiments show that SVM can be successfully selected for this step.

Experimental Setup

In this section we give details about the experiments performed on both high-level feature based systems. The two methods share some parts of the experimentation setup, like the used dataset and the used baseline.

Dataset

In this section we present the datasets that are used to train and evaluate the two proposed high-level methods.

In Section 3.2.2, only ASVSpoof 2019 dataset is used for training and testing, since the low-level SSD approach allows to obtain satisfactory performances while keeping down the number of samples. On the other side, high-level methods aim at reaching higher generalisation ability and performances but require larger datasets.

For this reason, we decide to combine multiple datasets for the experiments on high-level feature based SSD methods, to ensure that our proposed techniques do not over-fit to one dataset or domain, and is appropriate for real-world conditions.

In Table 3.4 we give details about the composition of each subset. The reader may find more details about the notation and the characteristics of each dataset in Section 3.1.3.

	Real	DF	Number of tracks			
			VC	TTS	Real	TOT
Train	$\mathcal{D}_{\text{ASV tr}}$, $\mathcal{D}_{\text{LS}}$	$\mathcal{D}_{\text{ASV tr}}$	7600	15200	22800	45600
Dev	$\mathcal{D}_{\text{ASV dev}}$	$\mathcal{D}_{\text{ASV dev}}$	7432	14864	2548	24844
Eval	$\mathcal{D}_{\text{ASV eval}}$, $\mathcal{D}_{\text{LJ}}$, $\mathcal{D}_{\text{IEM}}$	$\mathcal{D}_{\text{ASV eval}}$, $\mathcal{D}_{\text{CL}}$	14742	61028	22735	98505

Table 3.4: *Composition of train, development and test sets for high-level feature based experiments.*

Baseline

We compare the performance of our systems to those of another well-established neural network based state of the art method, Rawnet 2 [156]. It is an end-to-end network aimed at audio anti-spoofing detection, which has been proposed as a baseline in the ASVSpoof 2021 challenge [177]. The reader may find more details about Rawnet2 in Section 3.1.2. We trained this architecture using the same strategy originally proposed, i.e., on 4 seconds long windows of speech signal, using ADAM optimisation with learning rate = 0.0001, for 100 epochs using a batch size of 32. As training set, we extended the original $\mathcal{D}_{\text{ASV tr}}$ with $\mathcal{D}_{\text{LS}}$ (which contains only real speech samples), hence using the same training dataset used for our systems.

EmoSSD Setup

In the following we specify the evaluation setup details for the experiments relative to the emotional cues based SSD method, namely EmoSSD.

To avoid detecting dataset-specific artifacts, we pre-process all tracks to make them as uniform as possible. We convert all tracks to mono and, if necessary, down-sample them to a standard sampling frequency $F_s = 16$ kHz. Then, we filter all speech signals using a Butterworth band-pass digital filter with order 6, considering a low-cut frequency $F_l = 250$ Hz and a high-cut frequency $F_h = 3600$ Hz. Finally, we normalize each track using infinity norm. We compute the input of the SER block starting from a time-frequency transform, as detailed in [24]. Each track of the datasets

is reduced to have a common length $L_{cut} = 3$ s, using zero-padding if necessary. Then, we compute the STFT of x using a Hamming windows of length $L_w = 0.025$ s and a hop-size $L_h = 0.01$ s. Only the magnitude of the STFT is considered. The spectrum is then processed using a bank of mel-spaced filters and further scaled using the natural logarithm function. In our implementation we consider $M = 300$ windows and $K = 40$ mel bins.

The EmoSSD method, illustrated in Section 3.2.3, contains 2 parts which are trained independently. First, the feature extractor is trained to perform Speech Emotion Recognition (SER) following the procedure proposed in [24]. Specifically, we use the Interactive Emotional Dyadic Motion Capture (IEMOCAP) dataset, i.e., $\mathcal{D}_{\text{IEM}}$, and we consider the classes *angry*, *happy*, *sad*, *neutral*, hence $N = 4$. Since the $\mathcal{D}_{\text{IEM}}$ dataset is divided in 5 dialogue sessions, we select sessions 1 to 4 for training and session 5 for development and testing. We use Adam optimizer with learning rate $l_r = 10^{-5}$ and categorical cross-entropy as loss function. Our trained feature extractor achieves results comparable to those presented in [24] with a balanced accuracy of 0.6 on the four classes. The dimension of the feature vector $\mathbf{f}_x$ is $L = 256$.

The second stage of EmoSSD pipeline is trained to perform SSD, using features extracted from each dataset. The composition of training, development and test dataset corresponds roughly to the one previously presented in Table 3.4. As mentioned above, only TTS and mixed TTS/VC algorithms are taken in account. Therefore, for the training and fine tuning we consider only algorithms A01, A02, A03 and A04 of $\mathcal{D}_{\text{ASV tr}}$ and $\mathcal{D}_{\text{ASV dev}}$. In the evaluation stage, we consider algorithms from A07 to A17 of $\mathcal{D}_{\text{ASV eval}}$ dataset and the complete data from $\mathcal{D}_{\text{CL}}$.

The hyper-parameters for the RFC have been selected using a grid search on the validation set, using balanced accuracy as a metric. The considered parameters are the criterion of split quality and the number of learners. In particular, we tested as quality criterion functions both Gini impurity and information gain. Regarding the number of learners, we consider $N_{\text{RF}} = [10, 30, 100, 300]$.

In this specific experiment we consider not only the baseline introduced above but also an additional second baseline. We compared the results

of our transfer-learning approach with those obtained by training the first network of our pipeline directly for the SSD task. This test aims at verifying that the use of emotions cues really improves the accuracy of our system and it is relevant in increasing its overall performance.

ProsospeakerSSD Setup

We now present all the details relative to the evaluation setup on the prosody and speaker cues based SSD. ProsospeakerSSD system needs three independent training phases: one for the speaker embedding extractor based on TDNN, one for the prosody encoder and the last one for the final binary classifier.

We first pre-process all tracks consistently, setting the sampling frequency to $F_s = 16$ kHz and normalising the audio signal dynamic.

For the prosody encoder, we first extract the mel-spectrogram using a Hamming window of length $L_w = 0.05$ s and $L_h = 0.0125$ s. The final number of bins is $K = 80$, while the number of time windows M is not fixed, since the prosody encoder is able to compress input signals of any length. The prosody encoder has been trained as detailed in [146], on the Blizzard 2013 dataset. The only detail we change, due to computational issues, is the mini-batch size, that has been set to 8. The dimension of the prosody embedding vector is $L^{\text{PROS}} = 128$.

Regarding the speaker embedding architecture, we first pre-process the input by extracting MFCCs. First, STFT is performed on Hamming windows of length $L_w = 0.025$ s and hop size $L_h = 0.010$ s. The number of mel-frequency cepstrum coefficients is $B = 80$. The number of windows M is not fixed, since this architecture can accept sequences of any length. We use the pre-trained ECAPA-TDNN weights available online, that has been trained with Additive Margin Softmax Loss on the two datasets Voxceleb 1 and Voxceleb 2, as described in [35]. The final dimension of the embedding vector is $L^{\text{SPKR}} = 192$.

Finally, we train the binary classifier for SSD task using as input the fusion of the two embedding vectors. The composition of training, development and test dataset is the one presented in Table 3.4.

As mentioned, the most effective supervised classification algorithm is SVM. The values for the hyper-parameters C and kernel type has been se-

lected using a grid search on the development set, using balanced accuracy as tuning metric. In particular we test $C \in [0.01, 0.1, 1, 10, 100]$ and three type of kernels: sigmoid, RBF and polynomial.

Results

In this section we present together the results of the evaluation stage relative to the two high level feature based methods. We first focus on the emotional cues based SSD method, illustrated in Section 3.2.3. Then we separately present the results relative to prosody and speaker cues based SSD, detailed in Section 3.2.3. We then compare the two methods, EmoSSD and ProsospeakerSSD, highlighting the differences and comparing the evaluation stage findings. Finally, we present an additional experiment aiming at investigating the robustness of one of the proposed SSD systems in the presence of noise.

EmoSSD Results

In this section we present the results relative to the SSD task for the emotional embedding-based method, that we indicated as EmoSSD method.

The best hyper-parameter setup of the RFC classifier corresponds to information gain as quality criterion function and a number of learners $N_{RF} = 300$.

Figure 3.8 compares the ROC curves of our proposed method, EmoSSD, against our 2 baselines, i.e., Rawnet2 and SER architecture trained directly for SSD (SER/SSD). It can be noticed that EmoSSD outperforms RawNet2, reaching a value of AUC $= 0.98$. In Table 3.5 we report also EER and balanced accuracy values for the three methods

	EER	**Balanced Acc**
Rawnet 2	0.165	0.862
SER/SSD	0.201	0.794
EmoSSD	0.061	0.938

Table 3.5: *EER and balanced accuracy of EmoSSD method against Rawnet2 and SER/SSD on $\mathcal{D}_{ASV\ eval}$.*

This first experiment confirms that EmoSSD approach allows to achieve

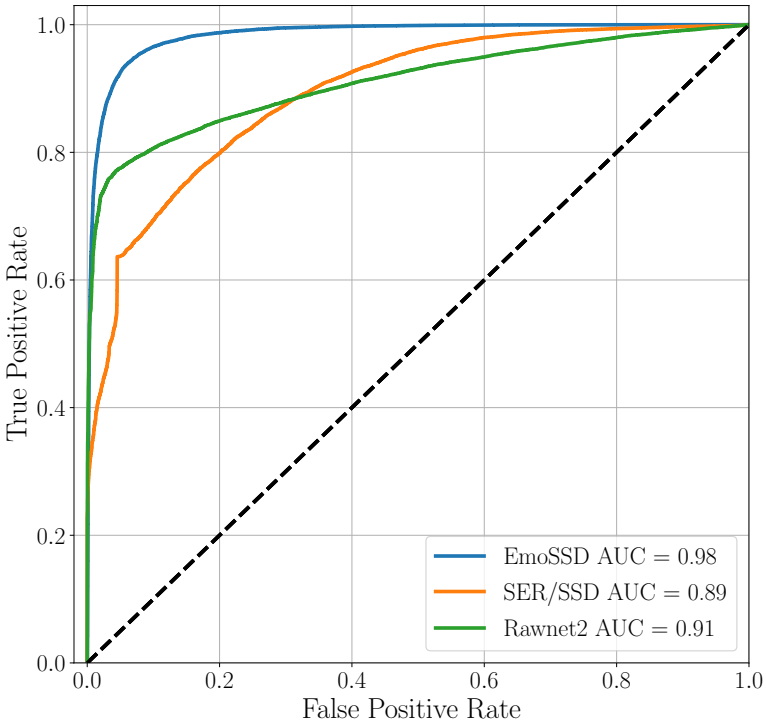


Figure 3.8: *ROC curves for EmoSSD method and the considered baselines on $\mathcal{D}_{ASV\,eval}$ and correspondent AUC values.*

higher discrimination capability even if compared to recent end-to-end methods. Figure 3.8 also shows that training architecture proposed in [24], originally proposed for SER but here used for the task of SSD, achieves worse results than our method. This shows that the effectiveness of our method does not lie in the architecture per-se, but in the knowledge gained in learning to predict the correct emotion label. Therefore, extracting emotional embeddings from an audio track creates a strong feature set, that it is still meaningful in deepfake detection task and it actually improves its accuracy.

To further test the potentials of the proposed method, we consider several additional datasets in the experiments , as already anticipated in Section 3.2.3 and detailed in Table 3.4. We set the binary classification threshold to 0.5 and we extract a predicted label $\hat{y}$ for each samples x. Then we define the detection rate as simply the ratio between the number of samples for which $\hat{y} = y$ over the total number of samples.

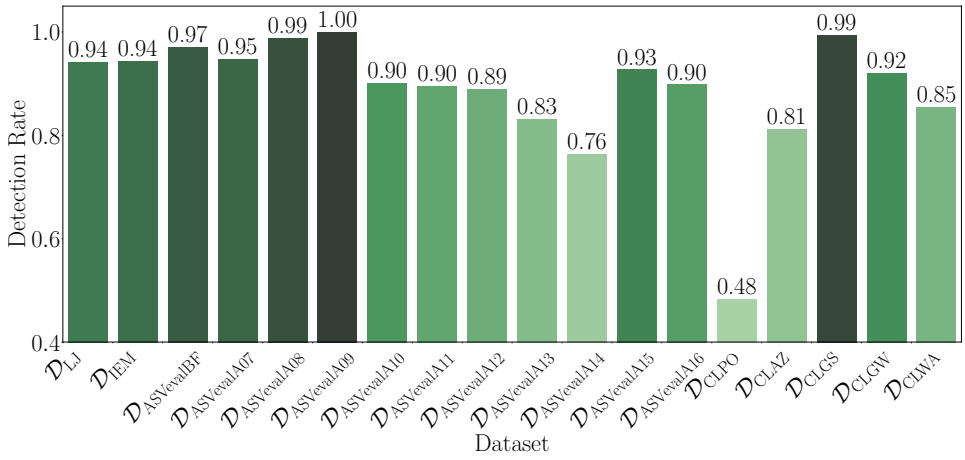


Figure 3.9: *Detection rate values on each subset for the EmoSSD method.*

In Figure 3.9, we present the results in terms of detection rate for each dataset and, if the dataset contains fake speech, for each synthesis algorithm. We can observe that performances are very good for all pristine signal samples and most deepfakes generation algorithms. We remind to the reader that only the algorithm A16 has been seen during the training stage, while all other ASV algorithms are unknown to the system. Algorithm A14 from ASVSpoof2019 and PO from Cloud2019 are the only cases where the detection rate is below 0.8. We suspect that this is because algorithm A14 is a mixed TTS/VC system that has been built starting from a very efficient VC system. Hence real emotional qualities are probably still present in the audio tracks, affecting the efficiency of the proposed system. For all the other deepfake systems, the detection rate accuracy value is close to or greater than 0.9.

ProsospeakerSSD Results

In this section we analyse the results obtained from the evaluation phase of the prosody and speaker cues based SSD method, namely ProsospeakerSSD.

We first compare ProsospeakerSSD method against the baseline Rawnet2. In Figure 3.10 the ROC curves for both the baseline and the proposed method are presented. Please note that here, differently from Figure 3.8, Rawnet2 has been trained on a larger dataset, where samples produced with

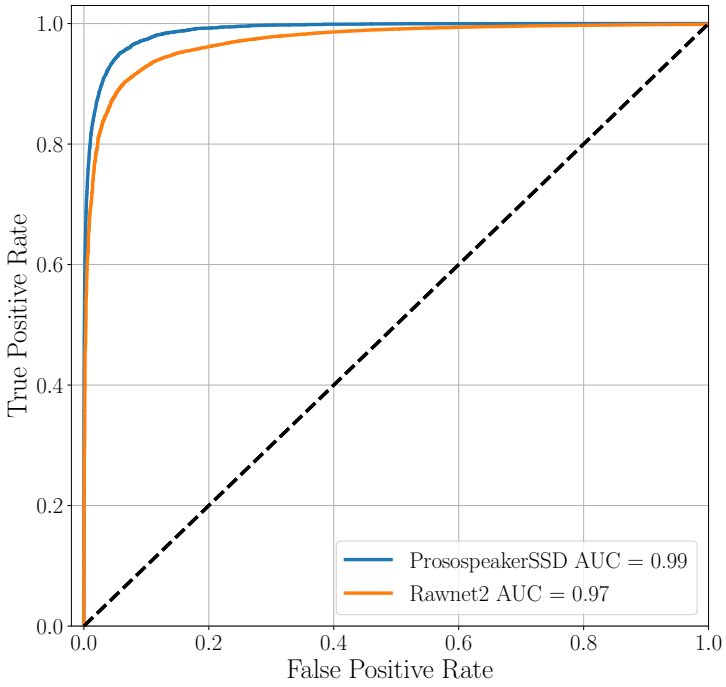


Figure 3.10: *ROC curves for the ProsospeakerSSD method and the considered baseline on $\mathcal{D}_{ASV\ eval}$ and correspondent AUC values.*

both TTS and VC algorithms are present. Our method is able to outperform the baseline, obtaining a AUC = 0.99. To further validate this comparison, we computed balanced accuracy and equal error rate. In Table 3.6 we present two additional metrics, EER and balanced accuracy.

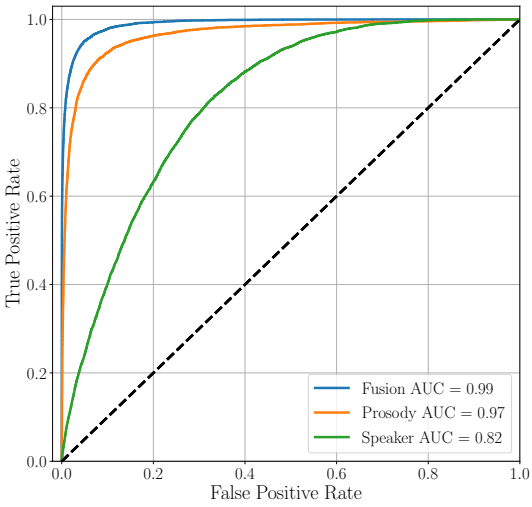
These metrics confirm what inferred from ROC curves and AUC values, i.e., our semantic based method is able to outperform the state-of-the-art baseline.

	EER	**Balanced Acc**
Rawnet2	0.083	0.915
ProsospeakerSSD	0.054	0.944

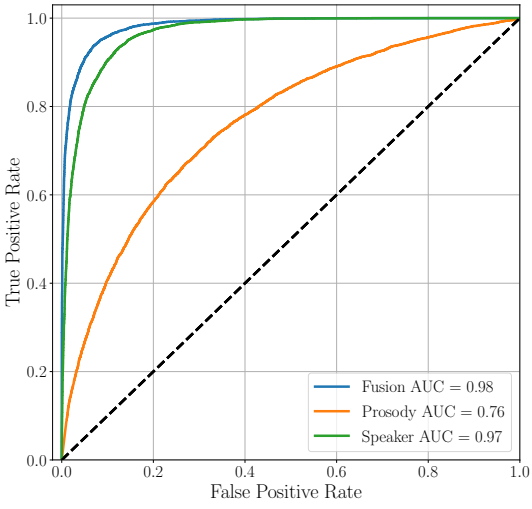
Table 3.6: *EER and balanced accuracy of ProsospeakerSSD vs Rawnet2 (baseline) on $\mathcal{D}_{ASV\ eval}$*

To assess the influence of speaker and prosody features in the final results, we perform an ablation study of the system. We repeat the training

of the SVM classifier using as feature representation only the prosody embedding $\mathbf{f}_x^{\text{PROS}}$ and only the speaker embedding $\mathbf{f}_x^{\text{SPKR}}$.The training and development dataset remain unchanged. In Figure 3.11 we present the ROC curves obtained with the original fusion system, for the system trained only with prosody embeddings and for the system trained only with speaker embedding. In the first plot, we test the three systems against TTS algorithms of $\mathcal{D}_{\text{ASV eval}}$. In the second plot we test on VC algorithms of $\mathcal{D}_{\text{ASV eval}}$.



(a) *Test on TTS systems.*



(b) *Test on VC systems*

Figure 3.11: *Ablation study of ProsospeakerSSD method.*

We can observe that in Figure 3.11a prosody embeddings alone achieve really good results, i.e., AUC = 0.97, compared to the one obtained only with speaker embeddings. Nonetheless, the fusion of the two improves the overall performances, since the original method reaches a value of AUC = 0.99.

On the contrary, when we test our systems only on VC algorithms, in Figure 3.11b, speaker embeddings alone are able to discriminate between synthetic and real speech samples, obtaining AUC = 0.97, while prosody embeddings are less effective. Also in this case, the fusion of the two is the best choice in terms of final results.

This ablation study proves the validity of what we assumed in the design of the system. A feature representation able to describe the prosody properties of the speech is crucial in detecting speech samples that lack of this characteristic, i.e., generated from TTS. On the other side, speaker embedding are effective when the original "human" content is preserved but the timbre and identity is manipulated, i.e., results of VC algorithms. Moreover, these two features are not completely orthogonal, since the fusion of the two allows to optimise detection accuracy of the final system.

To further assess the robustness of the complete ProsospeakerSSD method, we expanded the evaluation setup considering additional datasets, of both real and synthetic speech, as originally mentioned in Section 3.2.3 and similarly to what we have done for the emotional base cues method. In Figure 3.12 we present the values of detection rate for each dataset considered and for each synthetic speech algorithm. As already mentioned, detection rate is simply the ratio between the correctly labeled samples and the total number of samples.

Our system shows good results on the majority of the considered dataset, both for dataset containing only real samples, i.e., $\mathcal{D}_{\text{LJ}}$ and $\mathcal{D}_{\text{IEM}}$, and for datasets composed of synthetic speech examples different from the one in ASVSpoof2019, i.e., $\mathcal{D}_{\text{CL}}$. The dataset which causes most difficulties to our detection system is $\mathcal{D}_{\text{CL WA}}$, for which the prediction is almost random. One possible explanation is the fact that this generation algorithm, presented in [45], is combining an unit selection system with an advanced prosody prediction model, therefore producing natural speech samples in terms of prosody as a concatenation of real speech phoneme. We plan in the future

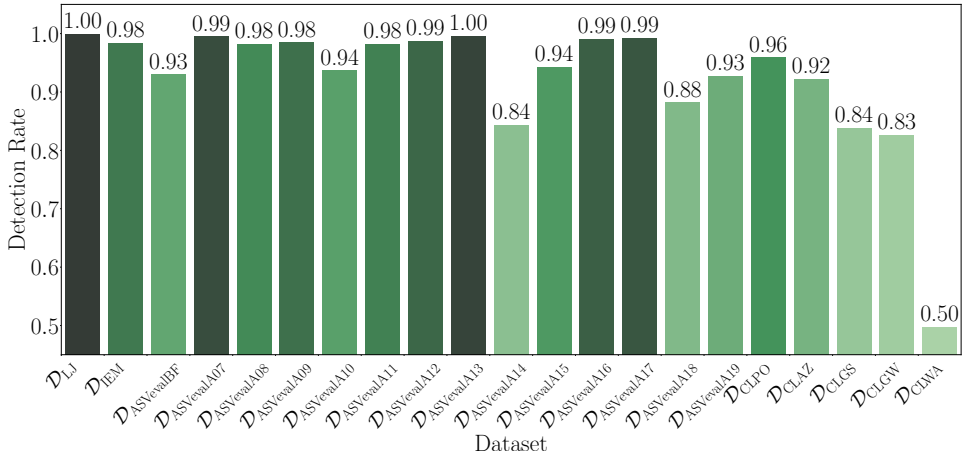


Figure 3.12: *Detection rate values on each subset for the ProsospeakerSSD method.*

to further investigate these techniques. The detection rate for all the other dataset and synthesis technologies is higher than 83%.

Finally, we present the results achieved by manipulating the input speech signal applying compression. In fact, in real-world scenario, audio deepfake may be manipulated to hide the artefacts introduced by the synthesis processing by, for instance, apply lossy compression to the resulting speech signal. Nonetheless, the proposed ProsospeakerSSD method relies on high-level semantic information and therefore be robust to such manipulations of low-level signal properties. We believe the combination of prosody and speaker embedding is still able to discriminate between synthetic and authentic speech. To assess this, we create three versions of $\mathcal{D}_{\text{ASV eval}}$ applying MP3 lossy compression with three different bitrates, namely $[32, 64, 128]$ kBits/s. In Table 3.7 we present the results obtained in terms of EER, balanced accuracy and AUC of ROC curve values.

The detector's performance deteriorates as we increase the compression factor, observing AUC and EER values dropping by 3 and 4%, respectively, between the two extreme cases. Balanced accuracy decreases significantly when compression is firstly introduced, with a drop of 5% between the no-compression and 128 kBits/s cases. At the same time, it maintains stable values when the bitrate decreases, falling only by 1% between 128 and 32 kBits/s cases. We can conclude that, overall, the proposed system, thanks to its high-level semantic approach, is able to maintain its effectiveness even

Compression Rate	EER	Balanced Acc	AUC
No Compression	0.054	0.944	0.99
128 kBits/s	0.069	0.898	0.98
64 kBits/s	0.071	0.897	0.98
32 kBits/s	0.098	0.885	0.96

Table 3.7: *ROC AUC, EER and balanced accuracy values of ProsospeakerSSD on compressed versions of $\mathcal{D}_{ASV\ eval}$ for different bitrates.*

in presence of strong signal compression.

Comparison between EmoSSD and ProsospeakerSSD

We now compare the results obtained for the two methods based on high-level features, EmoSSD and ProsospeakerSSD.

The first obvious difference is the set of synthetic algorithms which the two methods are able to detect. While EmoSSD works effectively only on TTS, ProsospeakerSSD reaches very good results also with speech samples created with VC techniques. This is of course a great advantage, since the ability to track down a variety of synthesis techniques is a desirable feature for a SSD system.

On the other side, ProsospeakerSSD requires a longer training phase, being composed of three different DL and ML systems, while EmoSSD's training step, having only one set of features, is shorter.

By looking at Figure 3.12 and Figure 3.9 we can compare the performances of the two methods on each test dataset and synthesis algorithm. On average ProsospeakerSSD reaches higher values of detection rate, both on bonafide and fake subsets. In fact, the mean value of detection rate on all datasets is $\mu_{\text{PRSPKR}} = 0.924$ with $\sigma_{\text{PRSPKR}} = 0.112$, while for EmoSSD $\mu_{\text{EMO}} = 0.887$ with $\sigma_{\text{EMO}} = 0.119$. It is interesting to notice that both systems have lower detection rate values for $\mathcal{D}_{\text{ASV eval A14}}$. This behaviour may be due to the fact that A14 is an hybrid TTS/VC algorithm.

Noise Robustness

In this section we present an additional experiment which aims at analysing the performances of one of the SSD methods in presence of audio degradation. We want to investigate the robustness of the detector and the ef-

fectiveness of training with augmented data. We focused on testing noise robustness only for EmoSSD method, but a similar analysis can be applied to the prosody and speaker cues based SSD.

We create a second version of the training and testing dataset using data augmentation techniques. Our goal is to add to the training data a wide variety of noise, whereas test data is obtained in a controlled scenario to enables results analysis. We do so by adding white noise to the speech tracks considering two different approaches. For the train and validation sets, we perform noise injection according to a double-layer probability distribution. The first layer injects white noise randomly between 30 dB and 15 dB of power SNR with probability $p_1 = 0.8$. The second layer randomly injects white noise between 15 dB and 10 dB of power SNR, with probability $p_2 = 0.3$. For the test set, instead, power SNR is fixed in the range SNR $= [25, 20, 15, 10]$ dB.

In the following we present the results relative to two experiments. In the first one, we simply test the original detector on noisy speech samples. In the second one, we use the augmented dataset for training the detector and we test it again on degraded speech samples. To train this second system we used the same setup presented in Section 3.2.3.

Table 3.8: *Results (detection rate) of the evaluation of the proposed system for different datasets and TTS algorithms using clean and augmented training sets.*

SNR	**Train**	**Real**			**Deepfake**													
[dB]	**Augm.**	$\mathcal{D}_{\text{LJ}}$	$\mathcal{D}_{\text{IEM}}$	$\mathcal{D}_{\text{ASV}}$ eval BF	$\mathcal{D}_{\text{ASV}}$ eval A07	$\mathcal{D}_{\text{ASV}}$ eval A08	$\mathcal{D}_{\text{ASV}}$ eval A09	$\mathcal{D}_{\text{ASV}}$ eval A10	$\mathcal{D}_{\text{ASV}}$ eval A11	$\mathcal{D}_{\text{ASV}}$ eval A12	$\mathcal{D}_{\text{ASV}}$ eval A13	$\mathcal{D}_{\text{ASV}}$ eval A14	$\mathcal{D}_{\text{ASV}}$ eval A15	$\mathcal{D}_{\text{ASV}}$ eval A16	$\mathcal{D}_{\text{CL PO}}$	$\mathcal{D}_{\text{CL AZ}}$	$\mathcal{D}_{\text{CL GS}}$	$\mathcal{D}_{\text{CL GW}}$	$\mathcal{D}_{\text{CL WA}}$
∞		0.941	0.943	0.970	0.948	0.988	1.000	0.900	0.895	0.890	0.831	0.763	0.927	0.898	0.483	0.812	0.993	0.921	0.855
25		0.947	0.944	0.996	0.911	0.883	0.992	0.875	0.861	0.803	0.740	0.710	0.872	0.736	0.421	0.558	0.966	0.851	0.610
20		0.965	0.943	0.999	0.814	0.699	0.917	0.800	0.783	0.539	0.632	0.547	0.687	0.439	0.304	0.264	0.819	0.639	0.331
15		0.982	0.942	0.999	0.565	0.421	0.587	0.576	0.542	0.202	0.446	0.216	0.303	0.129	0.138	0.032	0.361	0.238	0.060
10		0.988	0.934	0.999	0.342	0.224	0.223	0.355	0.334	0.128	0.314	0.084	0.093	0.080	0.093	0.000	0.051	0.038	0.020
∞	✓	0.854	0.828	0.865	0.975	0.994	1.000	0.957	0.973	0.940	0.910	0.877	0.965	0.941	0.603	0.832	0.996	0.966	0.920
25	✓	0.857	0.829	0.894	0.969	0.970	1.000	0.952	0.969	0.922	0.863	0.870	0.956	0.901	0.584	0.768	0.994	0.942	0.803
20	✓	0.861	0.829	0.904	0.947	0.926	0.999	0.939	0.961	0.892	0.824	0.834	0.927	0.837	0.533	0.522	0.978	0.907	0.697
15	✓	0.797	0.823	0.842	0.927	0.884	0.995	0.923	0.946	0.845	0.809	0.783	0.868	0.758	0.497	0.259	0.955	0.845	0.617
10	✓	0.656	0.807	0.800	0.886	0.817	0.984	0.907	0.916	0.843	0.836	0.764	0.829	0.748	0.466	0.268	0.887	0.767	0.676

Table 3.8 shows the detection rate of the proposed binary classifier for the two training configurations.

Detection rate are computed separately for each dataset of real speech tracks and each TTS algorithm used to generate deepfake speech tracks. The top half of the table shows the performance of the system trained on clean data, while the bottom half shows the performance of the system trained with noise-augmented data. In the first row of Table 3.8, the test set has not been augmented with noise, hence SNR $= \infty$. These values correspond to the results already presented in Figure 3.9 and show that, in absence of noise, the proposed method is able to well discriminate between synthetic and real speech.

We can observe in rows 2 to 5 that, as the noise level increases, the performances of the synthetic speech detector degrades more and more.

For lower SNR values, the classifier tends to label all samples as authentic. In fact, the detection rate for bonafide datasets increase at the decreasing of SNR, while detection rate for synthetic data drop dramatically, leading in some extreme cases to a detection rate = 0. This behaviour is, obviously, caused by the strong differences between training and testing speech samples. This experiment highlights the weaknesses of the proposed detector in a real-world scenario and it encourages the use of data augmentation strategy on the training set.

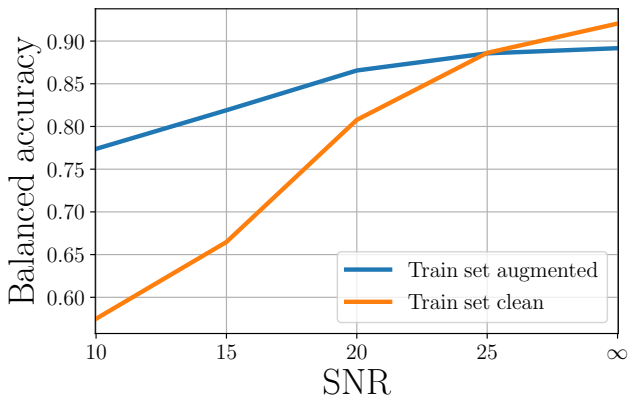


Figure 3.13: *Balanced accuracy values for arbitrary SNR using clean and augmented train sets on complete noise-augmented test set.*

Results relative to this second experiment are presented from row 6 to row 10 in Table 3.8. When the training set is augmented, the presence of noise in the testing set does not significantly affect the detector performance for high values of SNR. As noise level increase, we still observe a drop in detection rate values, which is however much smaller than the one observed in the previous experiment.

To further analyze the effects of training data augmentation, in Figure 3.13 we report the balanced accuracy values for different SNRs, training both on clean and augmented dataset. In detail, we consider as test set a fusion of all the test datasets and we compute accuracy with class-balanced sample weights.

From Table 3.8, we see that the system trained on clean data achieves higher accuracy on clean data. However, the latter outperforms the former in direct proportion to the decrease in SNR, reaching a difference of almost 20% in the noisiest experiment, i.e., for SNR $= 10$ dB.

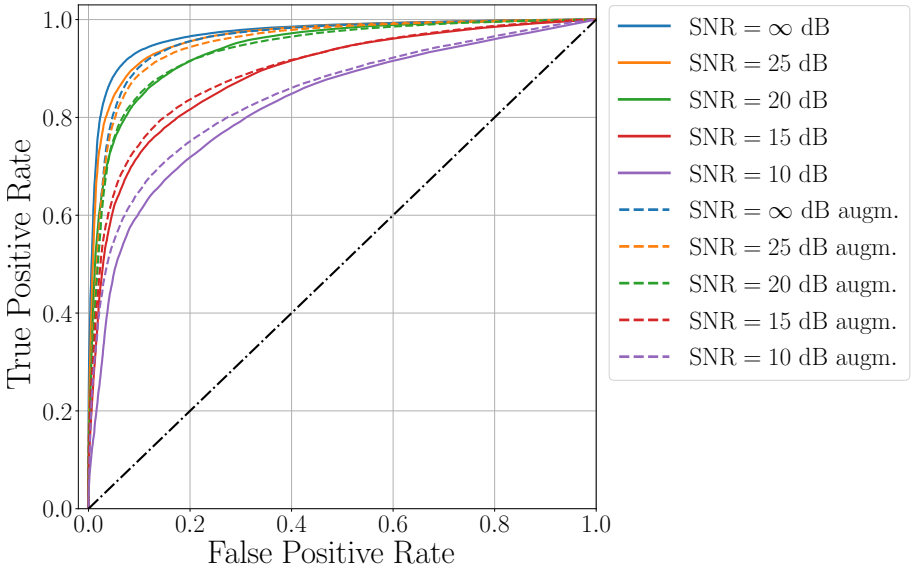


Figure 3.14: *ROC curves using clean and augmented train sets on complete noise-augmented test set.*

Figure 3.14 confirms this trend by showing the ROC curves obtained with the proposed method considering clean (solid) and augmented train sets (dashed). Also in this case, True Positive Rate (TPR) and False Positive

Rate (FPR) are computed simply considering all the samples in the test set. We can observe that, when SNR is high, training on clean data is more advantageous than using the augmented training set. As the test set SNR level decreases, the ratio between true positives and false positives generally lowers, but it drops more in the case of the classifier trained on clean data for the one trained on augmented data.

3.2.4 Conclusions

In this section, three different methods for synthetic speech detection have been presented. The first one is based on low-level features, defined starting from an auto-regressive model of the speech signal. This set of features acts as input to a supervised classifier stage, that predicts if the input is real or it has been synthetically forged. The evaluation proves that this method achieves really good results in closed set configuration, i.e., when all test samples have been generated with synthesis algorithms observed in training.

The second and third method follow a similar perspective, since they both aim at extracting contextual high-level features. In both cases, a transfer learning approach is adopted, hence a neural network is first trained for a different task and then it is used as embedding extractor for SSD task. In the second proposed method, EmoSSD, the focus is on the emotional content of speech, which is rarely present in synthetically generated speech. In the third approach, ProsospeakerSSD, prosodic and identity speaker embeddings are fused to address a larger set of synthesis techniques. Both methods have been tested against a state-of-the-art baseline and on an audio dataset obtained as the combination of different fake and real datasets. The evaluation has proved the validity of semantic approach, being able to correctly classify almost all real and synthetic speech samples. Obviously, the second method is a preferable choice since it tackles a larger variety of synthesis algorithms.

Finally, it is worth highlighting that high level feature based synthetic speech detection methods are based on a shared a priori assumption, i.e., that speech synthesis algorithms fail at recreating some human speech properties. We considered in particular emotional, prosody and speaker cues and we tested on most recent synthesis available algorithms. Nonetheless,

observing the incredible increase in synthetic speech's quality over the last few years, it is logical to imagine that future synthesis methods will be able to overcome these weaknesses and hence deceive our proposed detection methods. To address this issue, we believe the key strategy in the future of authenticity assessment in multimedia forensics is the extraction of semantic information on different levels and from different media. Examples are the extraction of the speech's content using speech-to-text techniques or the joint analysis of audio and video evidences. Moreover, the forensic analyst can create an ensemble combining high level feature based systems, like those proposed in this section, with methods focusing on signal level properties, e.g., low level methods presented above. In this scenario, authenticity assessment can therefore rely on the coherence among all media and all semantic levels and be robust to the advances of forgery methods.

3.3 Synthetic Speech Attribution

In this Section we focus on the task of synthetic speech classification, i.e., predict which synthesis algorithm has been used for the production of synthetic speech. The ability of identifying the synthesis technique allows to identify the origin or the authorship of the falsified audio. In fact, disinformation attacks on social media are often applied on scale and therefore a common synthesis pipeline is shared among multiple deep-fake audio tracks. By detecting the common traces left from each pipeline, the forensic analyst is able to link different audio assets to one single author and therefore to have a better understanding of the disinformation campaign. In the previous section of the chapter we analysed the problem of synthetic speech detection, addressing it as a binary classification problem and presenting two different strategies to solve it.

In this part of the manuscript, as mentioned, we focus on a different problem, namely synthetic speech attribution. We propose a system able to predict not only if the speech track is real or fake, but also, if it is fake, which algorithm has been used to create the track. This problem can be seen as a multiclass classification, where each class label is associated to one synthesis algorithm.

Unfortunately, speech generation is a very popular topic and several new

techniques are presented each year by tech companies and universities. This phenomenon raises questions about the effectiveness of any closed set synthetic speech attribution system. It is necessary to design systems able to deal with both known and unknown generation techniques.

These two tasks are called closed-set and open-set synthetic speech attribution, respectively. We address the two mentioned problems applying a framework similar to the one presented in Section 3.2.2, changing the second part of the pipeline, i.e., the classifier.

3.3.1 Problem Formulation and Method

In this Section we define the open set and closed set problem for synthetic speech attribution. Contextually, we illustrate the proposed methods for both scenarios.

Closed Set

Let us consider a speech signal $x(t)$ sampled at sampling frequency F_s. The speech signal is associated to a label

$$y \in [\mathrm{REAL}, \mathrm{DF}_1, \mathrm{DF}_2, ..., \mathrm{DF}_\mathrm{N}] \tag{3.27}$$

where the label REAL is associated to real, or bonafide, speech samples, while the label DF_i indicates the specific algorithm used for speech generation.

We propose a data-driven classification method that, given the input $s(t)$, produces an estimate $\hat{y}$ of the label y. In Figure 3.15 the pipeline of the system is presented and it is divided in two steps.

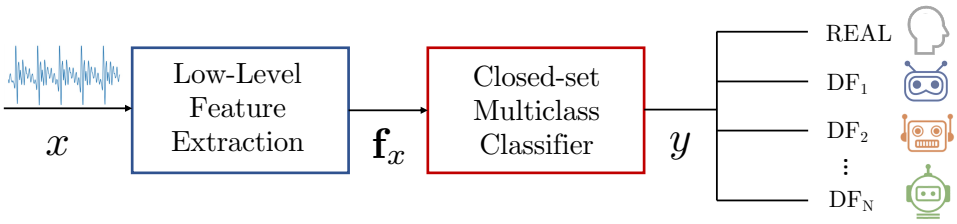


Figure 3.15: *Architecture of the closed-set multiclass pipeline.*

In the first step the speech audio signal is transformed into a feature vector, which aims at capturing the traces produced by the speech gener-

ation algorithm. The feature extraction phase is the one presented in Section 3.2.2, i.e., based on short-term and long-term analysis. This choice is motivated by the assumption that these features, which exploit source-filter speech model, have good discrimination power not only in identifying that the speech signal is synthetic but also in determining which algorithm has been used in the process. Moreover, this approach is suitable when not a large dataset is available for the training step. This scenario is plausible in the synthetic speech attribution task, since the analyst may find difficulties in collecting a large number of samples for each algorithm under analysis. Hence, the feature vector extracted by the low-level feature extraction step corresponds to the one described in (3.16).

$$\mathbf{f}_x = \mathbf{f}_x^{\mathrm{STLT}} \tag{3.28}$$

The second block of the pipeline is the classifier, as illustrated in Fig 3.15. It takes as input the feature representation $\mathbf{f}_x^{\mathrm{STLT}}$ and produces an estimate of the label $\hat{y}$. The mapping between the input features and the output is learnt during a training phase. It is worth noticing that we do not make any assumption about the classification method. In fact, any supervised classification method can be used, i.e., SVM or RFC.

Open Set

Let us consider a speech signal $x(t)$ sampled at sampling frequency F_s. The speech signal is associated to a label

$$y \in [\mathrm{REAL}, \mathrm{DF}_1, \mathrm{DF}_2, ..., \mathrm{DF}_\mathrm{N}, \mathrm{UNKN}] \tag{3.29}$$

where the label REAL corresponds to bonafide speech samples, DF_i indicates a specific synthesis algorithm while UNKN corresponds to synthetic speech samples produced with an unknown algorithm. With respect to closed set scenario, the proposed framework is able to deal with new and unknown synthesis technique and to output the label UNKN whenever a fake audio of uncertain origin is presented as input.

In Figure 3.16 we present the pipeline of the system, which is indeed very similar to the one presented in Figure 3.15.

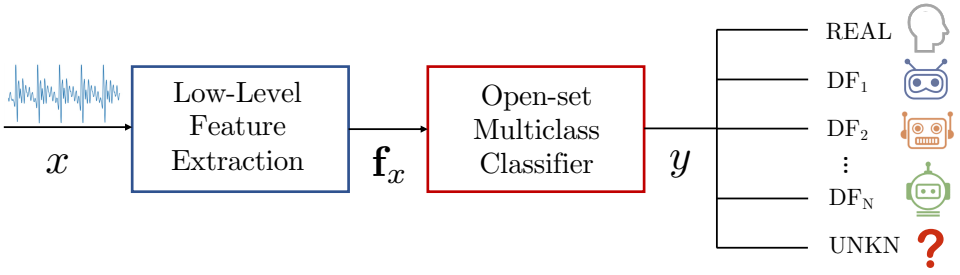


Figure 3.16: *Architecture of the open-set multiclass pipeline.*

In the firs step, a vector of features $\mathbf{f}_x$ is extracted starting from raw audio signal $x(t)$, applying a short-time and long-time analysis described in Section 3.2.2. Hence,

$$\mathbf{f}_x = \mathbf{f}_x^{\mathrm{STLT}}. \tag{3.30}$$

The following classification step is performed through a classic machine-learning supervised technique. The classifier is able to estimate the mapping between the feature representation and the output y. As already mentioned, in this scenario an additional class named UNKN is designed to gather all samples that are produced with synthesis methods new and never analysed by the system. It is worth noting that feature set used in the open and closed set scenario is the same and correspond to the low-level feature set presented in (3.16). As mentioned, this choice is driven by the very nature of the task, i.e., attribution, and by the possible scarcity of large collection of samples for each considered synthesis algorithm.

3.3.2 Experimental Setup

In this Section we report the technical details related to our experiments for both closed-set and open-set scenario.

The technical details relative to the baseline and the training strategy are illustrated in Section 3.2.2 in the part relative to low-level feature based method. The main difference lies in the output configuration of the supervised classifier. While for synthetic speech detection we consider only two labels as possible output, in this case the number of labels corresponds to the number of synthetic speech generation algorithms considered. Once

this change is applied, we repeat a grid-search using the same set of parameters on the same set of supervised classifiers. Therefore, the reader may find these details in the mentioned Section.

On the other hand, the partition of the datasets used for training and testing has been adapted to this scenario, and illustrated in the following.

Dataset

For this experimental phase we used the ASVSpoof2019 dataset, described in Section 3.1.3.

For the closed set scenario, we repeated the experiments twice. In the first case, we used $\mathcal{D}_{\text{ASV tr}}$ as training set and $\mathcal{D}_{\text{ASV dev}}$ as test set. This is possible since the training and development set share the same algorithm set, i.e., from A01 to A06. The second experiment is performed on the evaluation partition dataset, which contains a larger number of synthesis technique, from A07 to A19. In this case, the 80% of $\mathcal{D}_{\text{ASV dev}}$ is used for the training stage while the evaluation stage is carried on the remaining 20%.

In the open set experiment, we train the classifier on $\mathcal{D}_{\text{ASV tr}}$ and we test it on the union of $\mathcal{D}_{\text{ASV dev}}$ and $\mathcal{D}_{\text{ASV eval}}$. In this scenario for training we divide the speech synthesis algorithms in two sets. The first set contains all samples that will correspond to known classes, hence the classifier learns to recognise speech samples belonging to these specific algorithms. The second set of algorithms is gathered into a single class, namely known-unknown class. This known-unknown class has the role to prepare the classifier to successfully deal with speech samples created with algorithms never seen during training and to classify them correctly with the label "unknown". Specifically, we used as known classes the bonafide one plus 4 of the 6 synthetic classes present in $\mathcal{D}_{\text{ASV tr}}$. We select as known-unknown the two remaining synthetic speech methods from $\mathcal{D}_{\text{ASV tr}}$ (i.e., KN-UNKN).

3.3.3 Results

In this section we collect and comment all the results achieved through the performed experimental campaign. Specifically, we split the results depending on the used multiclass classification framework: closed-set and

open-set.

Closed-set results

In this experiment we considered the closed-set multi-class scenario. In practice we consider speech tracks generated by different algorithms as different classes. Therefore the goal is to detect whether the speech is bonafide (i.e., REAL) or synthetic, and to which synthetic class it belongs.

Figure 3.16 shows the confusion matrix obtained using the baseline Bicoherence, the proposed STLT and the fusion Bicoherence + STLT methods training the classifiers on $\mathcal{D}_{\text{ASV tr}}$ and testing on $\mathcal{D}_{\text{ASV dev}}$. This is possible as $\mathcal{D}_{\text{ASV tr}}$ and $\mathcal{D}_{\text{ASV dev}}$ share the same algorithms. For each method, we show the best results achieved through grid-search in terms of balanced accuracy, even though the same trend can be observed using different classifiers and parameters. In this scenario it is possible to notice that the baseline approach performs poorly, but it can be used to enhance the STLT method. The best balanced accuracy achieved by Bicoherence + STLT is 0.93.

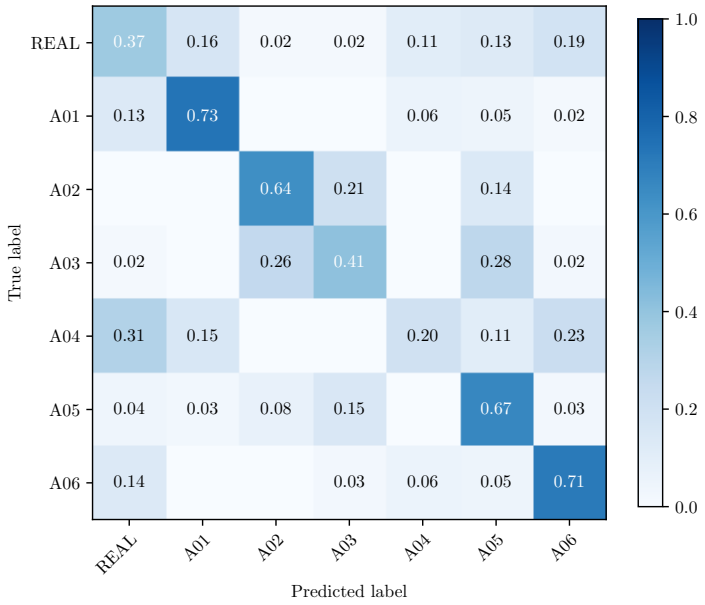
Figure 3.16 shows the same results achieved by training on a portion of $\mathcal{D}_{\text{ASV eval}}$ (i.e., 80%) and testing on the remaining portion of $\mathcal{D}_{\text{ASV eval}}$ (i.e., 20%). This was necessary as only two methods from $\mathcal{D}_{\text{ASV eval}}$ are present in $\mathcal{D}_{\text{ASV tr}}$. Therefore, to be able to classify in closed-set all the other methods, we had to show some speech tracks generated with them to the classifier. Also in this case STLT and the fusion Bicoherence + STLT provides satisfying results. The methods on which the classifiers suffer the most are A10 and A12, which exploit WaveRNN and WaveNet. However, the other NN-based methods perform well.

Open-set results

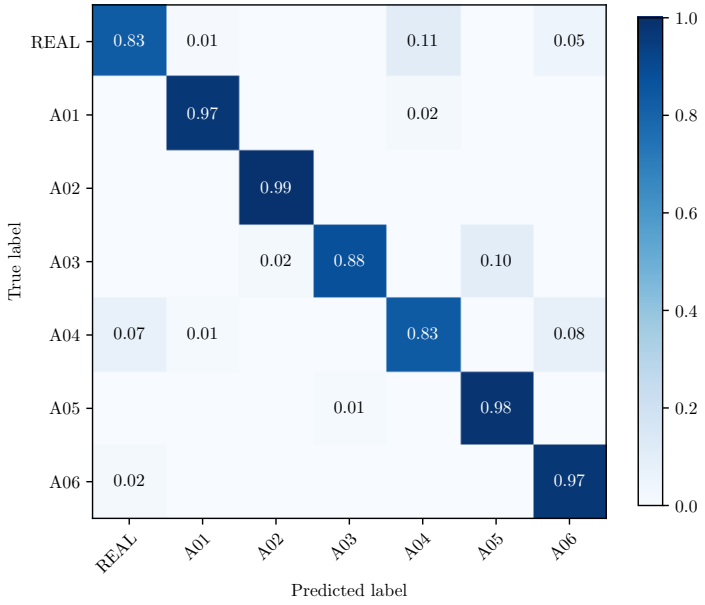
In this experiment we evaluate the open-set performance. The goal is to train the classifier on a limited set of classes (i.e., bonafide and some synthetic speech methods), and be able to classify the known classes as such, and unknown classes as unknown. In particular, as all unknown classes are synthetic speech by definition (i.e., there is only one bonafide class), the important point is to avoid mixing bonafide with fakes.

Figure 3.16 shows the results achieved training on $\mathcal{D}_{\text{ASV tr}}$ and testing

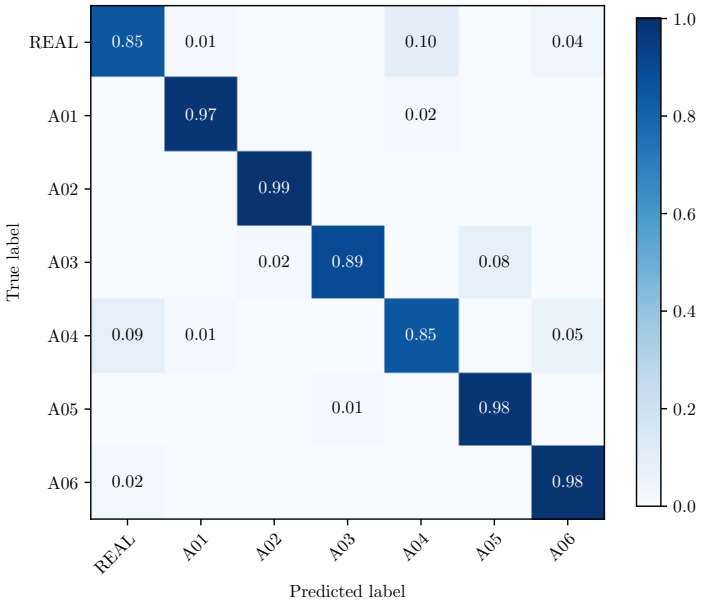
on the union of $\mathcal{D}_{\text{ASV dev}}$ and $\mathcal{D}_{\text{ASV eval}}$. Specifically, we used as known classes the bonafide one plus 4 of the 6 synthetic classes present in $\mathcal{D}_{\text{ASV tr}}$.



(a) *Bicoherence*



(b) *STLT*



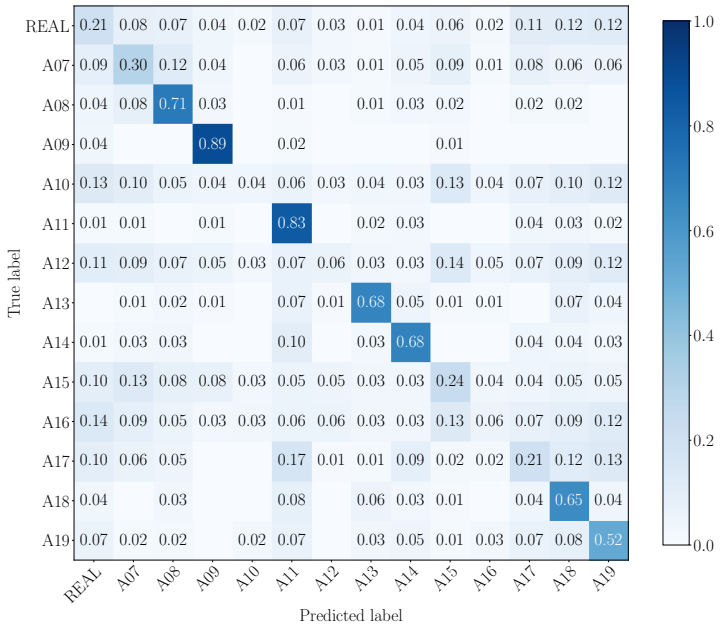
(c) *STLT + Bicoherence*

Figure 3.16: *Confusion matrices showing closed-set results for each used feature vector on dataset* $\mathcal{D}_{ASV\,dev}$

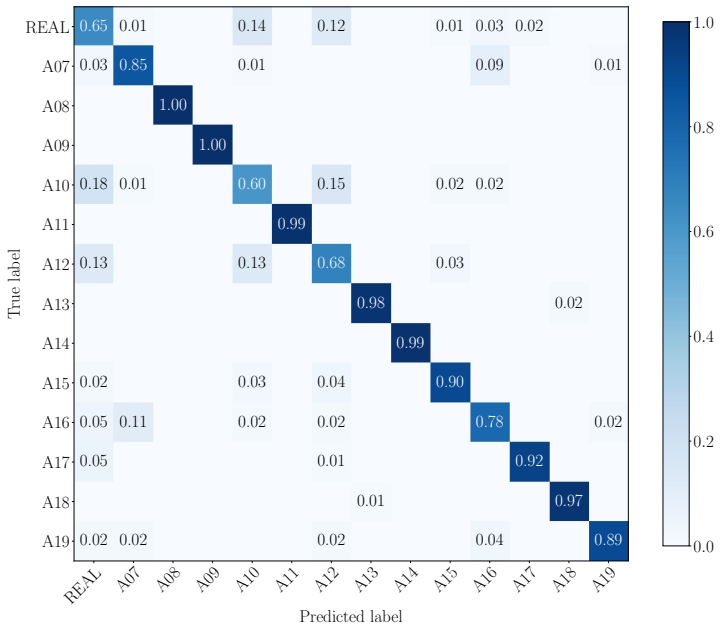
We select as known-unknown the two remaining synthetic speech methods from $\mathcal{D}_{\text{ASV tr}}$ (i.e., KN-UNKN). The classifier can give as output on label out of 6 classes: bonafide (i.e., REAL), one of the 4 known synthetic methods, or unknown (i.e., UNKN). Therefore, in presenting the results, we show the accuracy in predicting each one of the known classes, the accuracy in predicting the REAL class and the accuracy in predicting UNKN class. Moreover, we separate A16 and A19 classes, as they should be recognized as A04 and A06, respectively. All other classes are grouped as unknown (i.e., UNKN), as the classifier cannot distinguish sub-classes among them.

Figure 3.16(a) shows the results achieved selecting the pair (A02, A05) as known-unknown. In this case it is possible to see that all known classes are correctly classified, also considering A16 and A19. Unknown classes are unfortunately detected as bonafide 49% of the times. This means that, if the classifier predicts that the speech is synthetic or unknown, the classifier is most likely correct. However, when it predicts bonafide, there is a large possibility that the speech has been generated through a synthetic method.

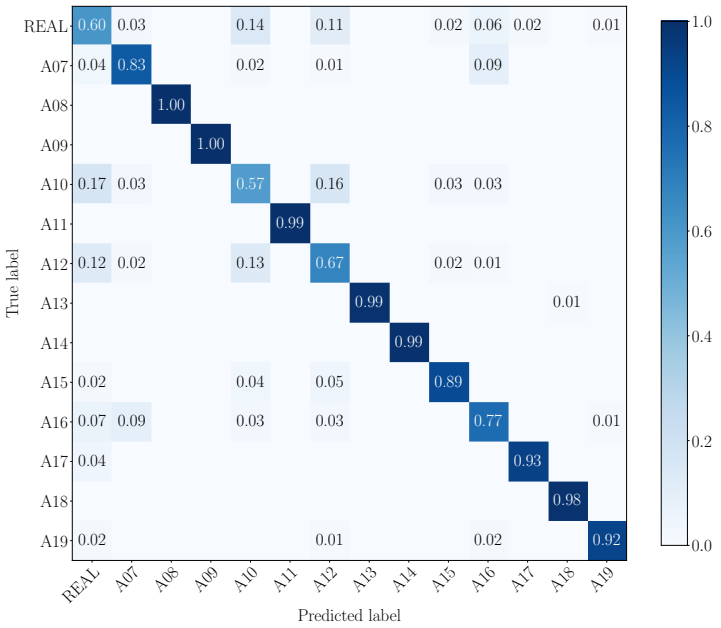
Figure 3.16(b) shows the same results in the case of known-unknown equal



(a) *Bicoherence*



(b) *STLT*



(c) *STLT + Bicoherence*

Figure 3.16: *Confusion matrices showing closed-set results for each used feature vector on dataset* $\mathcal{D}_{ASV\ eval}$

to the pair (A04, A06). In this case, A16 and A19 are correctly classified as unknown (i.e., the class to which A04 and A06 belong), and the same conclusions made before can be done.

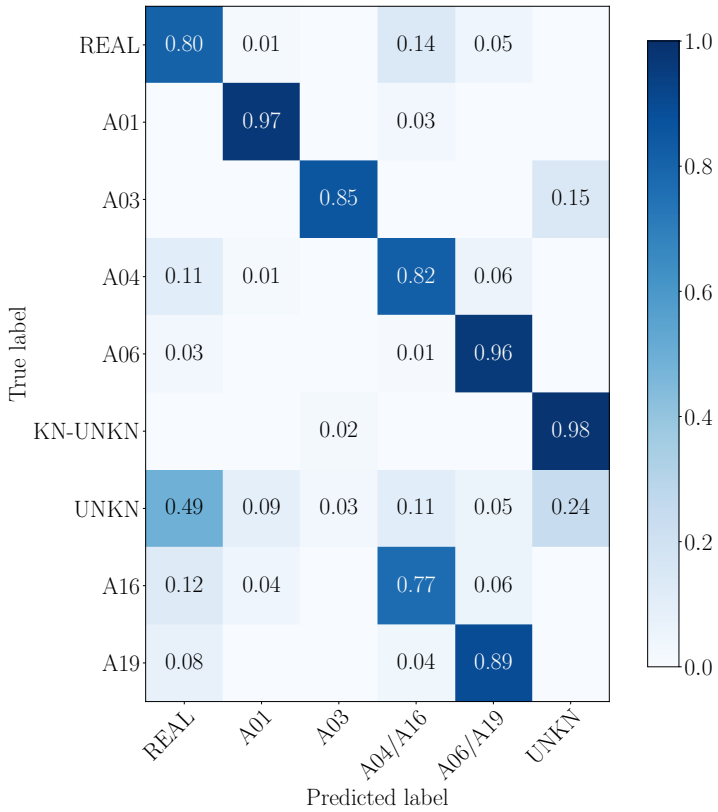
By digging more into the unknown speeches wrongly detected as bonafide, we noticed an interesting fact. Independently from the known-unknown pair selected at training time among the ones available in $\mathcal{D}_{\text{ASV tr}}$, the wrongly classified unknowns are A10, A11, A12 and A15. In fact they are misclassified as bonafide in 89% of the cases. These are methods based on WaveNet, WaveRNN and Griffin-Lim. The first two families of methods produce very natural sounding speech and they are based on end-to-end techniques. On the other side, the last family, based on Griffin-Lim algorithms, is never represented in the known-unknown set. All methods based on vocoders, waveform concatenation, waveform filtering even if post-processed with a GAN are correctly guessed. Therefore, to solve the open-set issue of wrongly classifying this subset of methods it is probably necessary to increase the amount of known-unknowns.

3.3.4 Conclusions

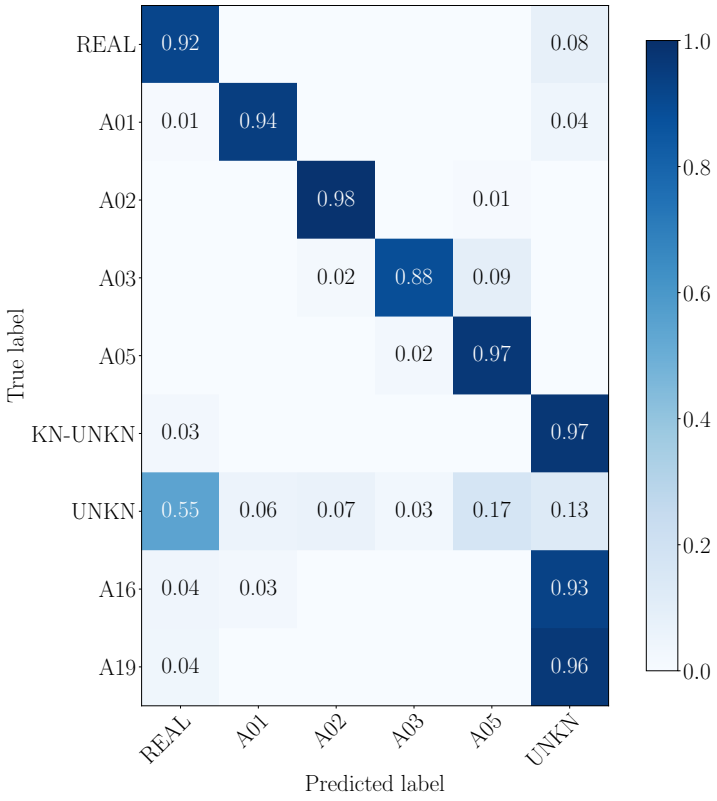
In this section we addressed the problem of synthetic speech attribution in both closed and open set scenarios. The proposed method is based on a classical supervised-learning pipeline: a set of features is extracted from the audio under analysis; a supervised-classifier is trained to solve the classification problem based on the extracted features.

The proposed features have been compared with the recently proposed baseline method [3] exploiting bicoherence analysis on the ASVSpoof 2019 dataset [169]. Results show that the proposed method outperforms the bicoherence-based one in both closed-set and open-set scenarios. Moreover, joint use of the proposed features and the bicoherence-ones provides an accuracy gain in some situations.

Despite the achieved promising results, the open-set scenarios needs fur-



(a) *KN-UNKN = (A02, A05)*



(b) *KN-UNKN = (A04, A06)*

Figure 3.16: *Confusion matrices showing Bicoherence + STLT open-set results on the union of $\mathcal{D}_{ASV\ dev}$ and $\mathcal{D}_{ASV\ eval}$.*

ther investigation. In fact, it is still challenging to accurately detect some families of synthetic speech tracks due to the huge variety of synthetic speech generation methods.

3.4 Final Remarks

In this section we addressed two different problems, synthetic speech detection and synthetic speech attribution. Both tasks have become crucial in the field of audio forensics, in particular for authenticity assessment of speech audio tracks.

Regarding SSD, we propose two different approaches: the first one is based on low-level features, defined through classic signal processing tech-

niques, while the second one is based on high-level features, which aim at capturing high-level semantic properties of the speech using NN architectures. Both methods have been evaluated on a large dataset of real and synthetic speech signals, showing promising results also in noisy scenarios. An analyst may choose between low-level and high-level strategy depending on the availability of data or on the computational resources.

Furthermore, we propose a system for SSA task, expanding the low-level feature framework to multiclass classification scenario. The evaluation stage is performed both in closed and open set configuration, and we observe that our method is able to address successfully the classification task with a limited class set. On the other side, the open set scenario represents a more challenging setup for the proposed method and needs further investigation.

CHAPTER 4

Integrity Verification

In this chapter we focus on integrity verification of audio tracks in forensics analysis. To verify the integrity of an audio file means to establish if the the audio file is completely original, or pristine, or if it has been subjected to a manipulation. Therefore, integrity verification techniques aim at detecting traces left from any operation. It is not difficult to imagine real-world scenarios where this kind of manipulation may be used. For instance, a speech by a person of interest can be modified, substituting some utterances or words and changing the actual content. If such an operation is done maliciously, it may represent a great danger for reliability of media communication and facilitate the spreading of false news. A second application example is integrity assessment for audio forensics analysis. In fact, these manipulation strategies can be used to falsify audio evidences or to steer the path of investigation.

Possible tampering operations on audio files are, for instance, deletion, copy move or splicing. In this section we specifically focus on the problem

of splicing detection and localisation, i.e., we assume the manipulated file is a combination of two or more audio tracks that has been concatenated in a specific point,i.e., the splicing point. We propose two methods that present some differences and some analogies.

The first method analyses the acoustic recording environment and looks for inconsistencies in the detected detected acoustic scenario. In fact, we assume that the splicing operation is a concatenation of two real recordings performed in two different acoustic setups. The reverberation time is used as characteristic descriptor of the environment, and it can be estimated directly from the audio signal by looking at energy curve trend on sub-bands. All the steps of the method are relying on signal analysis and processing techniques, hence no training data nor training stage is required.

In the second method we assume that the spliced audio is a combination of synthetic and real speech fragments. This scenario has been rarely investigated, but we believe that in the near future it may gain more centrality in the audio forensics research community. In fact, an attacker may exploit the recent NN based synthesis technique to target a specific identity, taking advantage of the availability of audio training data online, and to operate on specific utterances or words. To address this task, we rely on a feature representation extracted through an end-to-end spoof detection neural network. Analysed cues are related to the speech signal origin, i.e., if it is real or synthetic.

The two methods address two different types of splicing operation and consequently focus on different traces. Nonetheless, the adopted strategies show some similarities. The discriminating cues are extracted in a local fashion, i.e., working on short time frames with a specific overlap. The sequence of cues are then transformed in a function over time which follows a specific behaviour. When the extracted cues are constant over time, the function has small constant values. When the cues change, hence a splicing occurs, the function exhibits a peak. Therefore such a function is able to spot inconsistencies in the features under analysis. To detect a splicing we simply need to estimate a correct threshold while the location of the splicing corresponds to the location of the peaks. This simple approach allows to obtain satisfying results in both cases and it is flexible enough to take in account both single and multiple splicing case.

In the following section we first give a formal problem formulation of splicing detection and localisation. We then introduce the reader to the principal works present in the state of the art addressing the task. Then, we give details about the first method, based on acoustic cues. Finally, we present the second method, which tackles the partially synthetic speech detection and splicing localisation.

4.1 Problem Formulation

Formally, let us consider a set of audio signals sampled at frequency F_s. These N discrete time signals are defined as

$$\begin{aligned} &x_1(t), \quad t = 0, 1, ..., L_1 - 1, \\ &x_2(t), \quad t = 0, 1, ..., L_2 - 1, \\ &\dots \\ &x_N(t), \quad t = 0, 1, ..., L_N - 1 \end{aligned} \tag{4.1}$$

where L_i is the length of $x_i(t)$ track. A spliced audio track is obtained by concatenating in time the set of signals $x_0(t), x_1(t), \dots x_N(t)$, thus it is defined as

$$x_{\text{spliced}}(t) = [x_1(t) \parallel x_2(t) \dots x_{N-1}(t) \parallel x_N(t)].$$

The resulting length of $x_{\text{spliced}}(t)$ is $L_{\text{spliced}} = \sum_{i=1}^{N} L_i$. In Figure 4.1 we report a schematic representation of the splicing operation when $N = 2$.

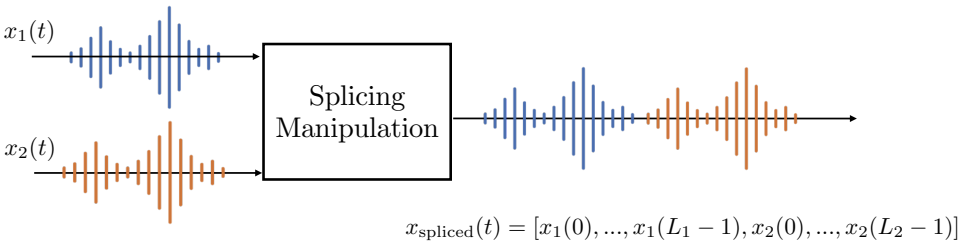


Figure 4.1: *Splicing operation schema for $N = 2$.*

Given a generic audio track, solving the splicing detection problem means understanding whether the audio track is a single recording, or it is a composition of two or more signals as $x_{\text{spliced}}(t)$. If this is the case, solving the splicing localisation problem means estimating the splicing time instants, i.e., the sample indexes $t_1^{\text{spl}}, .., t_{N-1}^{\text{spl}}$ where the sequences $x_1(t), ...x_N(t)$ meet (which will corresponds to $t^{\text{spl}} = L_1$ in the example depicted in Figure 4.1). In this chapter we will propose two methods that address both problems, namely audio splicing detection and audio splicing localisation.

4.2 Related Works

In this section we present some works from the state of the art that address the problem of audio splicing detection and localisation.

Traditionally, forgery detection methods are divided in two categories, active and passive techniques. Active techniques implement the authentication assessment through the verification of watermarks' presence, which certify the ownership, in the audio file. The watermark is directly embedded in the signal and it is designed to be inaudible, robust to noise or perturbations and relatively easy to extract. Recent examples of watermarking techniques may be found in [88], where the authors propose a fully data-driven decomposition of the audio signal and embed the watermark in the decomposed version of the input. Other examples are the watermarking techniques proposed in [12], based on wavelet decomposition, or [90], based on spread spectrum. Unfortunately, in real word and forensics applications an audio signal does not contain any watermark and hence watermarking techniques are not applicable. For this reason, passive, or blind, techniques analyse audio signal characteristic have been proposed to perform integrity authentication. One of the first features used in passive authenticity verification is the ENF, which is the frequency of the electric power system used for the recording devices. It is usually assumed that ENF is subject to oscillations and that its residual can be extracted from the audio recording. Since all devices powered by the same electric grid share the same ENF oscillations, integrity verification is performed looking for inconsistencies in the ENF traces [30, 56, 76]. Recently, in [39], inconsistencies in ENF traces are used to expose a splicing. As ENF traces are subtle and might be

hindered by high noise levels, in [100] the authors propose to use spectral phase analysis to increase noise robustness.

In the state of the art we find some works which address the problem of audio splicing detection and localisation looking at signal properties or acoustic parameters inconsistencies. As an example, in [31] the authors detect splicing by searching for signal discontinuities that are enhanced through high pass filtering. In [119], the authors focus on noise traces instead. The rationale is that different recordings may contain different amount of noise, thus noise level estimates can be used to expose splicing. Another interesting approach is proposed in [32]. Here the authors use a blind channel estimator to detect microphone response footprints. Audio tracks showing more than one microphone footprint are detected as spliced. More recently, the authors of [185] propose to exploit acoustic channel impulse response and ambient noise as environmental signature for an audio recording. If this signature changes in time, audio splicing is detected. On the other side, the research community has been investigating anti-forensics methods, i.e., methods which aim at compromising the effectiveness of forensic audio analysis, and anti-forensics detection. An example is [184], where both an anti-forensic and an anti-forensic detection framework are presented, the latter based on a rich-feature model based classification schema.

The problem of detecting spliced audio assembled with synthetic and real speech samples is a relatively new topic in the field of audio forensics analysis. While synthetic speech detection has been object of extensive investigation in the last few years, to the best of our knowledge there are not methods that directly address this problem or the subsequent splicing localisation task. One first attempt of rising interest around this topic is the work presented in [179], which focuses solely on partially spoof detection, not addressing the localisation problem. The authors test some popular synthetic speech detection architectures to detect only partially synthetic audio, proving that their performances degrades significantly.

4.3 Speech Audio Splicing Detection and Localisation Exploiting Reverberation Cues

In this section we address the problems of splicing detection and localisation making the assumption that spliced audio excerpts may come from different recordings, which can be therefore characterised by different environmental traces. In particular, motivated by [105], we exploit reverberation time as forensic trace. This measures the degree of reverberation characteristic of an audio signal propagating within an environment. Given a suspect audio track, our method estimates the reverberation time across different temporal windows and searches for possible inconsistencies. If reverberation time suddenly changes from an instant to another, the audio track is detected as spliced.

As already introduced in Section 2.1, Reverberation Time (RT) is defined as the time interval in which the sound pressure level is reduced by a specific range expressed in dB. This range is typically set from 0 dB to 60 dB, in which case RT is also called T_{60}. The higher the T_{60}, the longer the reverberation in the analysed room.

T_{60}can be analytically computed from a RIR. However, when a signal recording is available, estimating the complete RIR is a challenging task. Fortunately, it is possible to estimate the RT directly from an audio recording with some approximations [33]. These methods work particularly well on signals that exhibit small pauses from time to time. This condition is typically fulfilled by speech signals, as no matter how fast a person speaks, some pauses in between words are customarily present. As shall be clear from the next section, we exploit this property in our work.

4.3.1 Proposed method

The proposed method for speech audio splicing detection and localisation verifies the integrity of a suspect signal by analysing the acoustic properties of the reverberant room in which the recording has been performed. If the reverberation behavior of the environment shows a drastic change within the recording, splicing attack is detected. With respect to the problem formulation presented in Section 4.3.1, we consider the concatenation of $N = 2$ tracks, i.e., at most one splicing point is detected. As future

work, we plan to apply the proposed approach to detect and localise multiple splicing points.

To address the problem, we follow the pipeline depicted in Fig. 4.2. First, we turn the signal into a time-frequency representation. Then, we estimate the reverberation time on sliding windows. Finally we search for inconsistencies among estimated reverberation times along the recording. In the following, we provide details about each proposed step.

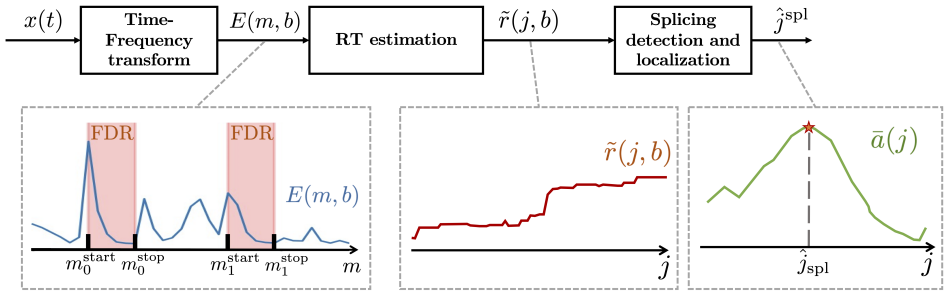


Figure 4.2: *Pipeline of the proposed acoustic-based splicing detection and localisation method*

Time Frequency Transform

The goal of this step is to turn the input signal into a representation that highlights regions useful for RT estimation. Given a recorded signal $x(t)$ sampled with sampling frequency F_s, we first divide it in J frames $x_j(t),\ j = 0, 1, ..., J-1$, using a rectangular window $w_R(t)$ of length L_{w_R} with overlap of L_{h_R} samples. The frame length L_{w_R} determines the temporal resolution for RT estimation. Each frame is transformed into a time-frequency representation through STFT, thus obtaining

$$X_j(m,k) = \sum_{n=0}^{L_w-1} x_j(n - mL_h)w(n)e^{-i\frac{2\pi}{K}nk}, \tag{4.2}$$

where $k = 0, 1, ..., K-1$ is the frequency bin index, $m = 0, 1, ..., M-1$ it the time window index, $n = 0, 1, ..., L_w - 1$ is the time sample index within the frame, $w(t)$ is a window of length L_w and L_h is the hop length. For the sake of notational simplicity, hereinafter we drop the frame index j

whenever not strictly necessary, keeping in mind that the following operations are applied per-frame.

As not all the spectral bands are relevant for RT estimation, we adopt an octave band representation of a particular portion of the spectrum. Specifically, we chose B significant octave bands described by their lower (i.e., $f_b^{\min},\ b = 0, 1, ..., B-1$) and upper (i.e., $f_b^{\max},\ b = 0, 1, ..., B-1$) frequency limits. Moreover, as phase information is not of interest in our scenario, we compute the energy envelope curve for each band. These two operations lead to

$$E(m, b) = \sum_{k=\lfloor K f_b^{\min}/F_s \rfloor}^{\lfloor K f_b^{\max}/F_s \rfloor} |S(m, k)|^2. \tag{4.3}$$

An example of $E(m, b)$ for one frame and band is shown in Fig. 4.2.

Reverberation Time Estimation

The goal of this step is to estimate a RT for each frame and for each octave-band independently. The adopted algorithm is divided in three steps.

In the first step, we identify and isolate Free Decay Regions (FDRs) in each rectangular frame, indexed previously with j,. These are defined as the portions of the signal where the sound stimulus has already finished and only the reverberation effect is present. These regions can be detected by looking for a persistent energy decrease in time, following the approach introduced in [165]. In a nutshell, the algorithm looks for $E(m, k)$ portions that are monotonically decreasing for at least $\bar{M}$ samples. In each $j - th$ rectangular window multiple free-decay regions can be present and detected.

We therefore obtain a set of I FDRs for each frame j and band b. Each FDR is described by its start time index $m_i^{\text{start}},\ i = 0, 1, ..., I-1$ and stop time index $m_i^{\text{stop}},\ i = 0, 1, ..., I-1$. Two FDRs are shown in the example of Fig. 4.2 superimposed to the related $E(m, b)$.

In the second step, we apply a modified version of Schroeder's algorithm [139] to each detected FDR to estimate the RT. To this purpose, we compute the energy decay curve, which is the normalised cumulative sum of the energy envelope $E(m, b)$ in dB, defined as

$$c_i(m,b) = 10\log_{10}\left(\frac{\sum_{\mu=m}^{m_i^{\text{stop}}} E(\mu,b)}{\sum_{\mu=m_i^{\text{start}}}^{m_i^{\text{stop}}} E(\mu,b)}\right), \tag{4.4}$$

with $m = m_i^{\text{start}}, ..., m_i^{\text{stop}}$, where i corresponds to the FDR index and b is the frequency sub-band index. For each band and FDR, we fit a line to $c_i(m,b)$ in the temporal dimension m using a least-square fitting procedure. The slope $d_i(b)$ of the fitted line can be used to estimate the RT value as

$$r_i(b) = \frac{-60/d_i(b)}{F_{\text{s}}/L_h}, \tag{4.5}$$

where F_s is the original sampling frequency and L_h is the hop size used for computing the STFT of equation (4.2). To obtain a single RT estimate per band, we average the estimated RT $r_i(b)$ using the fitting mean square error $e_i(b)$ as weight, thus obtaining

$$\bar{r}(b) = \frac{\sum_{i=0}^{I-1} e_i(b) r_i(b)}{\sum_{i=0}^{I-1} e_i(b)}. \tag{4.6}$$

As the process is repeated for each frame j, we end up with a RT estimate per frame and band $\bar{r}(j,b)$.

Finally, as RT estimates can be noisy due to the approximation process on short windows, we apply a cleaning operation. To reduce RT fluctuations over time, we apply a 1D median filter of size R to $\bar{r}(j,b)$, thus obtaining

$$\tilde{r}(j,b) = \text{median}\{\bar{r}(\gamma,b), \gamma \in [j - \lfloor R/2 \rfloor, ..., j + \lfloor R/2 \rfloor]\}.$$

An example of $\tilde{r}(j,b)$ for one band is shown in Fig. 4.2, where it is possible to see an increase in the estimated RT approximately from the middle of the signal.

Splicing Detection and Localisation

The goal of this step is to analyze RT estimates over time and detect and localise an inconsistency, if present.

If audio splicing occurs at time index j_{spl}, we expect that RT changes after j_{spl}. Therefore, $\tilde{r}(j,b)$ values for $j < j_{\text{spl}}$ should be strongly different

from $\tilde{r}(j,b)$ values for $j \geq j_{\text{spl}}$ within each band. To check whether this happens, we compare RT estimates before and after each possible j value. If a j providing noticeable RT differences exists, we detect and localise the splicing.

More specifically, for each band, we compute the Absolute Average Difference (AAD) between $\tilde{r}(j,b)$ samples to the left and to the right of each index j. Formally, for the b-th band we compute

$$a(j,b) = \left| \frac{1}{j} \sum_{\lambda=0}^{j-1} \tilde{r}(\lambda,b) - \frac{1}{(J-j)} \sum_{\lambda=j}^{J-1} \tilde{r}(\lambda,b) \right|, \tag{4.7}$$

for $j = Q, ..., J-Q-1$, being Q the minimum amount of samples that grants significant statistics before and after the candidate splicing time instant. To aggregate results over each frequency band, we make use of a weighted average. Formally, we compute the full-band AAD as

$$\bar{a}(j) = \frac{1}{B} \frac{\sum_{b=0}^{B-1} a(j,b) A(b)}{\sum_{b=0}^{B-1} A(b)}, \tag{4.8}$$

where $A(b)$ is the signal energy in the b-th band. An example of $\bar{a}(j)$ is shown in Fig. 4.2.

At this point, the full-band AAD $\bar{a}(j)$ should exhibit a pronounced peak in correspondence of the splicing time index, if splicing did occur (as shown in Fig. 4.2). We therefore search for peaks that have a minimum prominence (10% in our experiments), which measures how much a peak emerges from the neighboring baseline of the signal. If peaks exist, we select the highest one. The position j of this peak is considered the candidate splicing point $\hat{j}^{\text{spl}}$. The height of the peak $\bar{a}(\hat{j}^{\text{spl}})$ is used as splicing detection confidence value. In other words, we detect splicing if $\bar{a}(\hat{j}^{\text{spl}}) > T$, where T is a threshold that can be tuned by observing a small training set of data. It is worth reminding that this strategy is successful since we are addressing the problem of single splicing detection and localisation. In presence of multiple splicing points, the procedure should be slightly modified to consider multiple peaks and multiple peak positions.

4.3.2 Experimental results

In this section we first present the experimental setup designed for the evaluation step, including the dataset created for the task. Then, we present the metrics and the results for the proposed method compared to some baselines.

Dataset

For the evaluation step we have created a dataset which includes both pristine and spliced speech signals affected by reverberations. As already mentioned in Section 4.3.1, a reverberant audio signal can be obtained as the convolution between a dry source signal acquired in an anechoic environment, and a RIR for a specific room and source-receiver position.

As source signals we used part of the ACE dataset [38], which includes 65 utterances from both male and female speakers acquired in an anechoic room with variable length between 15 s and 90 s.

For the RIRs, we decided to include both simulated ones and RIRs acquired in real environments. To create synthetic RIRs we used a Python toolbox called Pyroomacoustics [131], which exploits the Image Source Model algorithm [4] for RIR simulation. We considered 7 shoe box rooms with volumes going from 54 m^3 to 700 m^3 and

$$\mathrm{T}_{60}^{\mathrm{PRA}} \in \{0.31, 0.40, 0.52, 0.62, 0.72, 0.82, 0.93\} \text{ s}. \tag{4.9}$$

Moreover, for each room two different source-receiver configurations have been considered.

This approach allows to quickly create a large set of simulated environments but lacks in describing the diffuse components, due to late reverberation and room irregularities. For this reason, we decided to take into account also real RIRs included in the ACE dataset. These signals are relative to 7 rooms, with volumes varying approximately from 47 m^3 to 360 m^3 and average reverberation time

$$\mathrm{T}_{60}^{\mathrm{ACE}} \in \{0.34, 0.37, 0.39, 0.44, 0.64, 0.65, 1.25\}\ s. \tag{4.10}$$

Also in this case, two different microphone and source positions have been considered.

By performing convolution between the considered RIRs and the dry speech signals, we obtained a set of reverberant speech signals, which have been further processed by adding an additive white noise for 3 different SNR levels, namely $\mathrm{SNR} \in \{10\ \mathrm{dB}, 20\ \mathrm{dB}, 30\ \mathrm{dB}\}$.

For the creation of tampered examples, a subset of the resulting speech signals have been concatenated in random position, reproducing the slicing operation. This entire procedure led to a total of approximately $20\,000$ audio recordings, equally divided in spliced and not spliced instances. Signals convoluted with real and simulated RIRs are always kept apart to allow a separate analysis on the two datasets.

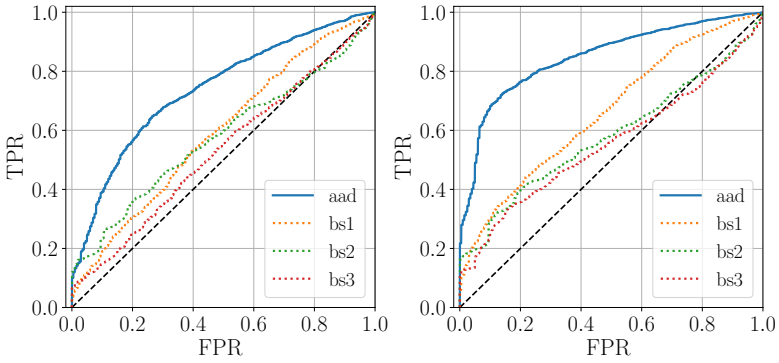
Setup

All the values for the parameters used in our algorithm are presented in Table 4.1.

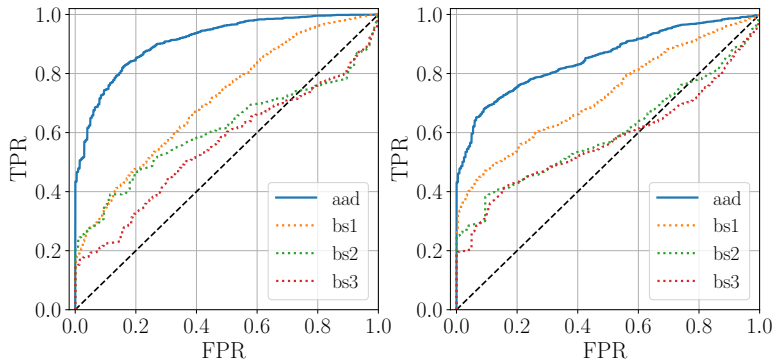
Table 4.1: *Parameters of the evaluation setup for the acoustic cues based splicing detection method.*

Parameter	Value	Parameter	Value
F_s	16 kHz	B	6
L_{w_R}	32000 (2 s)	$f_b^{\min}$	$[88.4, 176.8, 353.5, 707.1, 1414.2, 2828.2]$ Hz
L_{h_R}	16000 (0.5 s)	$f_b^{\max}$	$[176.8, 353.5, 707.1, 1414.2, 2828.2, 5656.8]$ Hz
L_w	800 (0.05 s)	$\bar{M}$	13 (~ 0.5 s)
L_h	600 (0.0375 s)	R	7 (~ 0.25 s)
K	1024	Q	133 (~ 5 s)

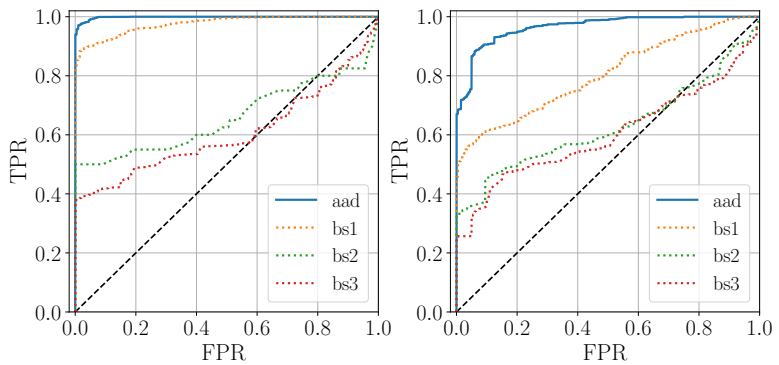
The performance of our method for the detection task is compared to three different baseline methods. They all share the RT estimation step proposed in our method, but they use different indicators to detect whether RT remains constant or changes in time. The first one (*bs1*) makes use of the standard deviation of the estimated RT. The second one (*bs2*) makes use of the difference between the maximum and minimum RT estimates. The third one (*bs3*) makes use of the maximum magnitude of the RT gradient in time. Whenever one of these indicators exceeds a threshold, splicing is detected.



(a) $\Delta T_{60} \in [0, 0.25)$ *ACE* **(b)** $\Delta T_{60} \in [0, 0.25)$ *Simulated*



(c) $\Delta T_{60} \in [0.25, 0.5)$ *ACE* **(d)** $\Delta T_{60} \in [0.25, 0.5)$ *Simulated*



(e) $\Delta T_{60} \in [0.5, 2)$ *ACE* **(f)** $\Delta T_{60} \in [0.5, 2)$ *Simulated*

Figure 4.3: *Results for the acoustic-based splicing detection method in terms of ROC curves obtained with different* ΔT_{60} *values compared to all baseline methods. Figures (a), (c) and (e) are relative to the ACE dataset. Figures (b), (d) and (f) are relative the simulated dataset.*

Detection results

For the evaluation of the splicing detection task, we adopted ROC curves, which show TPR and FPR pairs for the different threshold values T.

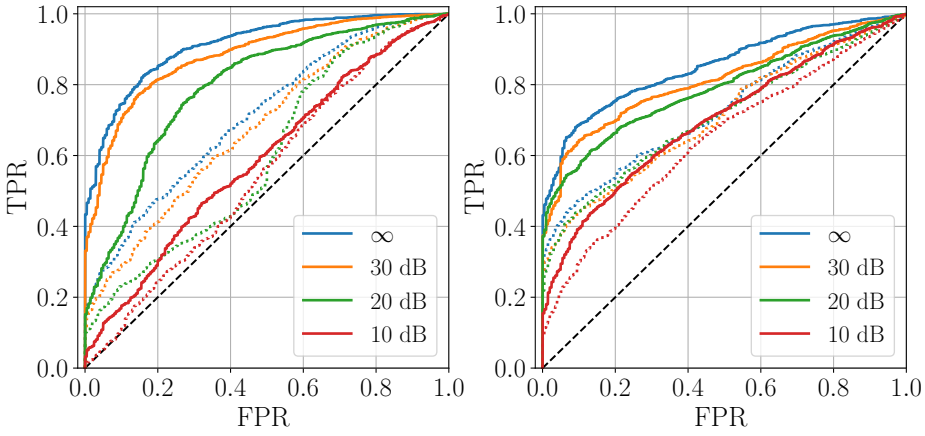
We first present ROC curves in a noiseless scenario for different $\Delta\mathrm{T}_{60}$, i.e., the absolute value of the difference between reverberation time before and after the splicing point. The smaller the $\Delta\mathrm{T}_{60}$, the closer the RTs before and after the splicing point. Therefore, a small $\Delta\mathrm{T}_{60}$ depicts a more challenging setup. Fig. 4.3 reports results for the two different datasets against the baseline methods. We can observe that the higher is $\Delta\mathrm{T}_{60}$, the better is the performance of the proposed method as expected. The proposed method always outperforms all baselines, confirming that the use of a deeper statistical analysis of reverberation times through AAD enables better performance especially for low $\Delta\mathrm{T}_{60}$ values. Finally, notice that the achieved performance are better on ACE dataset for high $\Delta\mathrm{T}_{60}$, whereas they look better on the simulated dataset for smaller $\Delta\mathrm{T}_{60}$. This highlights the impact that diffusive events that are present in ACE but not in the simulated data impact on RT estimation.

To evaluate the impact of additive noise, we also report ROC curves for different SNR values in Fig. 4.4 . In this case, we set $\Delta\mathrm{T}_{60}$ to the interval $[0.25, 0.5]$, and only show the best baseline (i.e., *bs1*). Notice that, when the SNR decreases, all methods lose effectiveness in detecting spliced recordings as expected. Nonetheless, for SNR=30 dB detection is still adequate, in particular for the ACE dataset.

From the above analysis, it is possible to select an appropriate threshold value T according to the desired ratio between TPR and FPR.

Localisation results

Regarding the splicing localisation task, a preliminary consideration is necessary. The proposed method relies on RT, which can only be estimated within FDRs. Therefore, we can only tell whether a splicing occurs in-between two different FDRs, but we cannot estimate the precise time instant. As a consequence, the splicing point localisation is affected by an intrinsic error, determined by the distance between two successive FDRs. We therefore evaluate splicing localisation by providing the correct locali-



(a) *ACE dataset* **(b)** *Simulated dataset*

Figure 4.4: *Results for the acoustic-based splicing detection method in terms of ROC curves for different SNR values compared to baseline* bs1 *(dashed)*

sation rate defined as the fraction of times we predict the splicing point up to an error of 5 s with respect to the real splicing.

Tables 4.2 and 4.3 show localisation rates obtained for the two dataset and for different values of ΔT_{60} and SNR. Best results are obtained for noiseless recordings and high values of ΔT_{60} as expected. In particular, we get 86% of correct localisation on the ACE dataset. As already observed for the detection task, the algorithm tested on the ACE dataset gives better results with respect to the simulated one. This is due to the fact that simulated RIRs are an approximation of a real-world scenario. Nonetheless, with real RIRs we achieve better results.

When the noise component increases or the change in RT values is less accentuated, localisation performance degrades. It is interesting to observe that on the ACE dataset the method seems to suffer more from small values of ΔT_{60} than from lower SNR values. We can assume that, when the difference before and after the splicing in RT is noticeable enough, the performance is still positive, despite the low SNR value.

4.3.3 Conclusions

In this section, we faced the problem of speech audio splicing detection and localization using acoustic environment cues. The goal is to understand

Table 4.2: *Results for the acoustic-based splicing localisation method in terms of localization rates for ACE dataset.*

ΔT_{60} \ SNR	10 dB	20 dB	30 dB	∞ dB
[0, 0.25)	0.029	0.030	0.096	0.202
[0.25, 0.5)	0.065	0.231	0.408	0.535
[0.5, 2)	0.606	0.70	0.836	0.861

Table 4.3: *Results for the acoustic-based splicing localisation method in terms of localization rates for simulated dataset.*

ΔT_{60} \ SNR	10 dB	20 dB	30 dB	∞ dB
[0, 0.25)	0.085	0.182	0.285	0.425
[0.25, 0.5)	0.196	0.417	0.574	0.680
[0.5, 2)	0.257	0.492	0.626	0.744

whether a speech signal is original or it has been manipulated through splicing. To solve this problem, we proposed a method that exploits inconsistencies in estimated reverberation time. Specifically, we estimate the amount of reverberation in time from an audio signal, and we verify whether reverberation time suddenly changes. The proposed method has been validated on real and simulated room impulse responses applied to male and female speakers with different amount of additive noise.

The proposed method is tailored to speech signals as it requires multiple free decay regions to be present in the recording. Future work will be devoted to more robust reverberation time estimation methods that can be applied also to other kinds of signals. Moreover, an iterative procedure to detect and localize more than one splicing point will be devised.

4.4 Partially Synthetic Speech Identification and Splicing Localisation through Metric Learning

In this section we present a novel method for detecting partially synthetic speech signals and localise the splicing point. As mentioned in Section 4.2, several techniques have been proposed to detect and localise splicing based on environmental cues but almost no investigation has been performed on

the task we address, i.e., identification and localisation of splicing implemented as a concatenation of bonafide and spoof audio tracks. Nonetheless, we believe this problem may play a crucial role in audio forensics analysis and research and methods addressing it are clearly in demand.

The method we propose combines two different strategies. On one side, a state-of-the-art NN architecture produces an embedding representation of the input in a local fashion. On the other side, a signal processing technique, originally proposed for music segmentation task [46], computes a novelty function which allows to identify splicing points by means of a peak finding algorithm. This combination is successful thanks to the use of a metric learning approach. In fact, the embedding extractor network is trained combining a traditional classification loss with a triplet loss [140, 154], which controls the configuration of the embedding space. In fact, this design choice allows to define a feature space where the distance between embedding vectors is meaningful, i.e., close points corresponds to input signals belonging to the same class. The novelty function provides sufficient information to predict whether the audio track has been manipulated or not and to localise the splicing point.

4.4.1 Proposed Method

In this section we present our proposed method for splicing detection and localisation of partially synthetic speech signal. As already mentioned, we aim at detecting if a speech signal has been manipulated substituting fragments of bonafide speech with synthetic ones. If this splicing operation happened, the splicing point in time is localised.

In Figure 4.5 we depict a schematic of the proposed method. The input speech signal $x(t)$ is first segmented in windows and then processed by the embedding extractor block, which outputs a feature vector $\mathbf{f}_j$ for each time frame j. The architecture used for embedding extraction is Rawnet 2, trained with a metric learning approach. The sequence of features is then transformed in a novelty function $\Delta(j)$, designed to have high values in correspondence of the splicing point and low values otherwise. The final step analyses the novelty function and detects whether the original audio has been spliced or not, and possibly estimate the splicing point. In the following we detail every block of the pipeline.

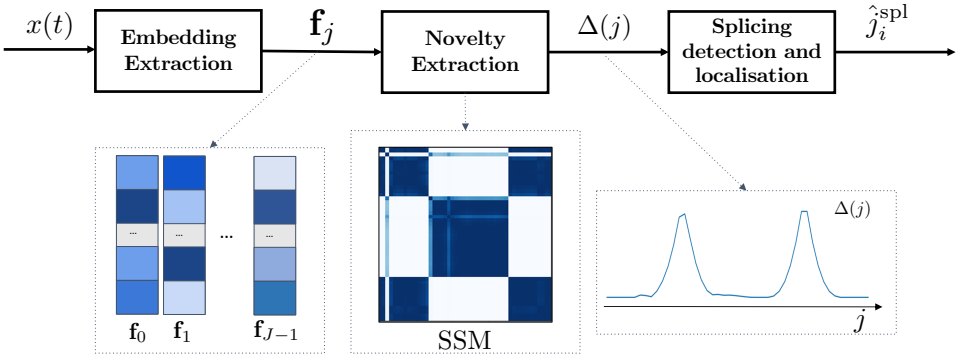


Figure 4.5: *Pipeline of the proposed splicing detection and localisation method for partially synthetic speech.*

Embedding extractor

The objective of this step is to extract an embedding representation able to characterise the speech generation process and detect whether the speech is authentic or it has been synthetically generated. Since, by definition of the splicing problem, this feature varies over time, we extract the embeddings representation locally. Starting from the input signal $x(n)$, we divide it into J frames $x_j(n)$ with $j = 0, 1, ..., J-1$, using a rectangular window of length L_w and overlap of L_h samples. Each frame $x_j(n)$ is then projected in an embedding space using a pre-trained network, obtaining a vector $\mathbf{f}_j$, of dimension L. As embedding extractor network, we propose Rawnet 2, already presented in 3.1.2 and used as a baseline in Section 3.2.3. Rawnet 2 is an end-to-end network for synthetic speech detection, originally proposed in [156]. Since this architecture takes as input a raw waveform, the input signal $x_j(n)$ is directly fed into the network, hence no pre-processing or preliminary transformation is needed. At the end of this step exactly J embeddings of dimension L are extracted.

In this work we rely on the fact that this embedding representation is able to describe the synthesis strategy of the analysed frames. In fact, frames belonging to the same class, synthetic or real, will be mapped in the same portion of the embedding space, while samples of different classes will be distant. To achieve this, we adopt a metric learning approach in the training stage of this network to further enforce this discriminative property. In

particular, we apply a two head expansion to the original Rawnet 2 architecture, which allows to combine triplet loss and categorical cross-entropy loss. This approach allows to achieve two goals. On one side, we optimise the network to solve the binary classification problem, i.e., synthetic speech detection. On the other side, triplet loss enforces proximity in the embedding space of samples belonging to the same class (synthetic or real) while pulling apart samples of different classes. This re-configuration of the embedding space facilitates the task of splicing detection, allowing to use a technique based on distance measures between the embedding vectors extracted from consecutive time windows of the original audio. More details about the implementation of the metric learning approach may be found in the evaluation setup below.

It is worth noting that our method is not strictly dependent on the architecture of the back-end embedding extraction,i.e., Rawnet 2. In fact, other deep neural networks trained on the task of synthetic speech detection may be used.

Triplet loss

We briefly introduce the reader to triplet loss, originally proposed in [140] for face recognition. The triplet loss is a loss function defined on triplets of the input samples. Each triplet i is composed of three inputs, namely the anchor x_a^i, a positive input x_p^i, which belongs to the same class of the anchor, and a negative input x_n^i, which belongs to a different class w.r.t. the anchor. Each triplet is transformed in an embedding representation, usually through convolutional or feed-forward layers, obtaining $\mathbf{f}_a^i$, $\mathbf{f}_p^i$ and $\mathbf{f}_n^i$. The triplet loss aims at minimising the Euclidean distance between the anchor and the positive embedding representations and, at the same time, maximising the Euclidean distance between the anchor and negative embedding representations. A schematic representation of this idea is reported in Figure 4.6.

Formally, this condition can be expressed as

$$||\mathbf{f}_a^i - \mathbf{f}_p^i||^2 + \alpha < ||\mathbf{f}_a^i - \mathbf{f}_n^i||^2 \quad \forall\, i \in \mathcal{T}, \tag{4.11}$$

where α is a parameter called margin, that corresponds to the minimal distance between positive and negative pairs, and $\mathcal{T}$ is the set of all valid

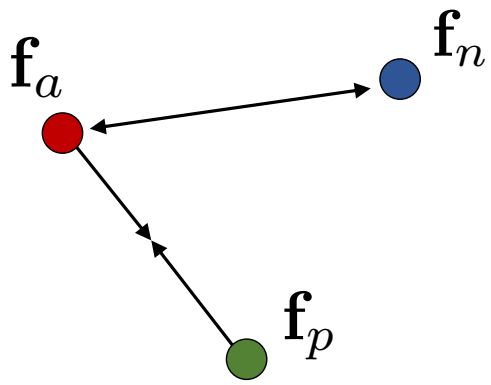


Figure 4.6: *Triplet loss strategy*

triplets of a training dataset with cardinality I. This corresponds to minimise the triplet loss, defined as

$$\mathcal{L}_{\text{tri}} = \sum_{i}^{I} \max(||\mathbf{f}_a^i - \mathbf{f}_p^i||^2 - ||\mathbf{f}_a^i - \mathbf{f}_n^i||^2 + \alpha, 0). \tag{4.12}$$

In practice, the selection of all possible triplets present in a dataset is unfeasible, hence triplet selection is usually performed online, i.e., on large batches of the training dataset. Moreover, different triplets mining strategies are used to ensure convergence of the training. For more details about the mining strategies we refer the reader to the original publication [140].

Novelty function extraction

Once the embedding representation $\mathbf{f}_j$ is extracted for each time frame j, we propose a second step to extract a novelty function $\Delta(j)$. Such a function should exhibit high values in correspondence of splicing point, more specifically when the speech signal switches from being authentic to be synthetically generated and the contrary. The procedure for the novelty function extraction follow the methodology proposed in [46] and originally proposed for automatic audio segmentation.

Given the J embeddings $\mathbf{f}_0, \mathbf{f}_1, ..., \mathbf{f}_{J-1}$ extracted from the analysed audio, we compute the distance matrix of these embeddings. The distance matrix is a square $J \times J$ symmetric matrix D, such that each element $D(j_1, j_2)$

is computed as

$$D(j_1, j_2) = ||\mathbf{f}_{j_1}||^2 - 2\mathbf{f}_{j_1} \cdot \mathbf{f}_{j_2} + ||\mathbf{f}_{j_2}||^2 \tag{4.13}$$

with $j_1 = [0, 1, ..., J-1]$ and $j_2 = [0, 1, ..., J-1]$ and where the operator $||\cdot||$ indicates the L^2 norm and $\cdot$ corresponds to the dot product. The matrix D is then further processed, to obtain a Self Similarity Matrix (SSM) S such that each element $S(j_1, j_2)$ is defined as

$$S(j_1, j_2) = exp\left(\frac{-\beta D(j_1, j_2)}{\sigma_D}\right) \tag{4.14}$$

where β is a parameter and σ_D is the standard deviation of the matrix D. The reader may find an example of SSM in Figure 4.5.

Then, the novelty function $\Delta(j)$ is extracted from SSM $S(j_1, j_2)$ for each frame index j with the following procedure. We correlate a checkerboard-like kernel $K(n_1, n_2)$ of size $N \times N$ along the main diagonal of the S matrix as

$$\Delta(j) = \sum_{n_1=-N}^{N} \sum_{n_2=-N}^{N} K(n_1, n_2) S(j + n_1, j + n_2). \tag{4.15}$$

with $j = [0, 1, ..., J-1]$.

In particular, we employed a two-dimensional Gaussian kernel K_{Gauss}, defined as

$$K_{Gauss}(n_1, n_2) = \phi(n_1, n_2) K_{box}(n_1, n_2) \tag{4.16}$$

where the K_{box} is a box kernel of dimension $N \times N$. The dimension of the box (and Gaussian) kernel N is assumed to be odd, and, by indexing the matrix with $\nu_1, \nu_2 \in [\lfloor -N/2 \rfloor : \lfloor N/2 \rfloor]$, can be defined as

$$K_{box}(\nu_1, \nu_2) = \text{sgn}(\nu_1)\text{sgn}(\nu_2), \tag{4.17}$$

where sgn is the sign function.

The function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ is a radially-symmetric Gaussian function computed as

$$\phi(n_1, n_2) = exp(-\epsilon^2 (n_1^2 + n_2^2)). \tag{4.18}$$

The parameter $\epsilon > 0$ is used to control the degree of tapering toward 0 at the edges of the kernel. The scaling applied with the Gaussian function

allows to smooth the edges of a simple checkerboard-like kernel, avoiding edge effects.

The final $\Delta(j)$ describes the behaviour of the distance measure between subsequent embeddings vectors. When the distance is small, i.e., the embeddings are relative to speech fragments belonging to the same class (real or synthetic), the novelty function has small values. On the other side, when a splicing occurs and hence the class changes (from real to synthetic or vice versa) the Gaussian kernel highlights the distance has increased and a peak is observed in the novelty function.

Splicing detection and localisation

As a final step, we estimated the presence and the location of the splicing point by analysing the novelty function $\Delta(j)$. In particular, we apply a peak-finding algorithms, pre-filtering the results constraining a minimum value for the prominence of each peak (10%). Please note that the prominence of each peak measures how much a peak extends outward from the lowest value.

The positions of the selected peaks correspond to the predicted splicing positions $\hat{j}_1^{\text{spl}}, \hat{j}_2^{\text{spl}}, ..., \hat{j}_{N-1}^{\text{spl}}$.

For splicing detection, the input signal is classified as spliced if at least one peak of the novelty function is above a selected threshold. Formally

$$\exists\, i \mid \Delta(\hat{j}_i^{\text{spl}}) > T, \tag{4.19}$$

where the threshold T is a parameter of the algorithm.

4.4.2 Experimental Results

In the following we give the details about the experimental setup used to validate the proposed method and we present the results we obtained. We first describe the two dataset built for training and testing the system. We then report the specifics about embedding extractor training and novelty computation. Finally, we comment the results obtained for both detection and localisation task.

Dataset

To validate the proposed approach, we define and build two different datasets of spliced and pristine audio tracks.

We start from the audio tracks of the ASVSpoof 2019 dataset, originally proposed in [160] and illustrated in Section 3.1.3. This dataset includes a collection of real and synthetic speech samples, generated with different algorithms, and divided in three partitions: training, development and evaluation set. Moreover, the dataset includes metadatas which specify the speaker identity. Genuine and synthetic tracks may share the speaker, i.e., the synthetic ones have been created targeting a specific identity used in the bonafide recordings.

The first dataset, $\mathcal{D}_{\text{S1}}$, includes both pristine and spliced audio files. The spliced audio tracks are created concatenating one real and one fake speech track in random position and order. The selected bonafide and spoof samples are associated to the same speaker, i.e., the perceived speaker does not change before and after the splicing point. The pristine audio tracks are created concatenating two bonafide or two spoof speech signals, again associated to the same speaker identity. Moreover, if the two concatenated tracks are synthetic, we consider samples generated with the same synthesis algorithm. The reader may notice that fully synthetic speech signals are considered pristine signals. This is motivated by the design of our method, which aims at detecting only transitions between synthetic and real speech and vice versa. For both spliced and pristine tracks, possible silences at the start and at the end of the original audio tracks are trimmed before the concatenation operation. Depending on the ASVSpoof 2019 partition used, we define $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$, created starting from the development and the evaluation partition respectively. We anticipate that the training partition of ASVSpoof2019 is used to train the embedding extractor and it is hence excluded from the evaluation dataset. Therefore, we present the results on $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$ separately, since the first one has been created using the same synthesis algorithms present in the training partition, while the second one contains samples obtained with different algorithms.

The second dataset created for the evaluation is $\mathcal{D}_{\text{S2}}$, created with the purpose of testing the proposed method if two splicing points are present.

Therefore, spliced audio tracks are obtained concatenating three audio files, alternating bonafide and spoof samples. On the other side, pristine audio tracks are created concatenating either three bonafide or three synthetic speech samples. Again, we guarantee that only one speaker identity is present in the final speech track and that all synthetic portions are generated with the same algorithm. Two subsets $\mathcal{D}_{\text{S2 dev}}$ and $\mathcal{D}_{\text{S2 eval}}$ of $\mathcal{D}_{\text{S2}}$ are defined, based on the original ASVSpoof 2019 partition used. In Table 4.4 we specify additional details about the two datasets.

		Pristine	Spliced	Tot	Num Speaker	Num Synth Alg
$\mathcal{D}_{\text{S1}}$	$\mathcal{D}_{\text{S1 dev}}$	9877	10052	19929	10	6
	$\mathcal{D}_{\text{S1 eval}}$	19955	19991	39946	48	13
$\mathcal{D}_{\text{S2}}$	$\mathcal{D}_{\text{S2 dev}}$	10023	9977	20000	10	6
	$\mathcal{D}_{\text{S2 eval}}$	20086	19962	40048	48	13

Table 4.4: *Breakdown of $\mathcal{D}_{S1}$ and $\mathcal{D}_{S2}$ dataset, showing development and evaluation splits composition per number of samples, speakers, and synthesis methods.*

Setup and baseline

In this section we explain the training strategy used for the embedding extractor block and the parameters used for the novelty extraction and splicing detection and localisation operations.

For the embedding extraction process, we divide the audio in J frames of length $L_w = 1$ s and hop size $L_h = 0.125$ s. Then each frame is used as input to a Rawnet 2 network, which has been modified and optimised to address the splicing detection and localisation task. We indicate this new version of Rawnet 2 as 2HeadRawnet 2.

The first difference is the length of the input fed to the network, which is originally equal to 4 seconds while in our case the input is one frame of length $L_w = 1$ s. This modification is necessary since we need a finer time resolution to address the task of splicing detection.

The second difference is the network front-end design. While the back-end is the original one, for the final part of the network we adopt the two head expansion strategy originally proposed in [154]. The network is modified to have two heads, or outputs, that takes as input a "feature map". In

general, the feature map corresponds to the output of the back-end part of the network and in Rawnet 2 corresponds to the output of the GRU layer. The first head is the classification head, which applies in sequence a pooling, a flatten and a fully connected layer such that the final number of neurons is equal to the number of classes (in our case 2). The second head is the embedding head, which process the feature map by flattening it and applying a fully connected layer of final dimension equal to the desired embedding dimensionality, in our case $L = 512$. In Figure 4.7 we report a schematic of the applied 2HeadRawnet 2 architecture.

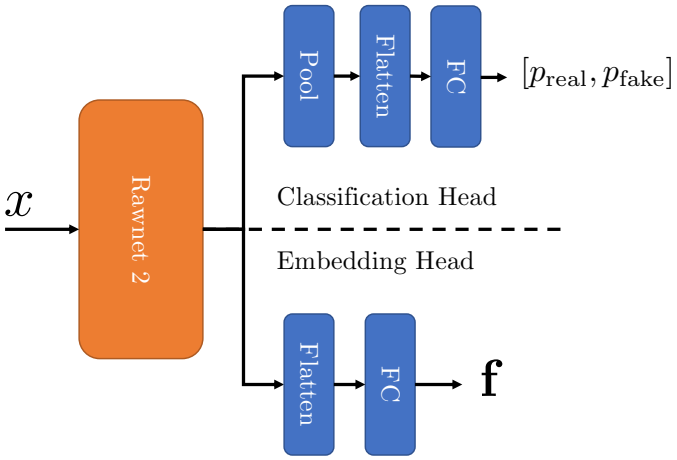


Figure 4.7: *2HeadRawnet 2 for partially synthetic speech splicing detection and localisation.*

During the training stage, each head is trained with a different loss. In our case, the classification head is trained with binary cross entropy loss $\mathcal{L}_{\text{BCE}}$, which optimise the weights of the network to solve the binary classification problem. The embedding head is trained with triplet loss $\mathcal{L}_{\text{tri}}$ [140], which on the other side promotes a better embedding distribution in the feature space. In particular, to select the triplets we used a batch-hard online mining strategy [140], adding two constraints in the triplet definition. First, we select anchor, negative and positive of each triplet such that all three are associated to the same speaker identity. This choice helps the network to learn to discriminate samples taking into account only the speech generation process and not the speaker identity. The second constraint is that when the anchor is a synthetic speech, the positive must have been

generated with the same algorithm. Doing so, the network is encouraged to cluster together synthetic samples generated with the same algorithm, achieving higher generalisation ability.

The two losses are combined together to obtain the final loss

$$\mathcal{L} = \mathcal{L}_{\text{BCE}} + \lambda \mathcal{L}_{\text{tri}}. \tag{4.20}$$

We trained the 2HeadRawnet 2 on the training partition of ASVSpoof2019 dataset, using Adam optimizer with decaying learning rate. All the details relative to the training stage are reported in Table 4.5.

Parameter	**Value**	**Parameter**	**Value**
Batch Size	256	γ_{LR}	0.85
Num Epochs	40	Margin Triplet Loss	0.5
LR init	0.001	λ	0.5

Table 4.5: *Training parameters for embedding extractor of partially synthetic spoof detection and localisation method.*

After training 2HeadRawnet 2, we evaluate the network training by testing it on development and evaluation partitions of ASVSpoof 2019 dataset ($\mathcal{D}_{\text{ASV dev}}$ and $\mathcal{D}_{\text{ASV eval}}$). We report the results in Table 4.6. With respect to the original Rawnet 2, these two metric values highlight a small drop in the performances. Nonetheless, we need to take in account the fact that we are considering shorter portions of the raw audio signal (only 1 second w.r.t. 4 seconds in the original version). Moreover, the training network here is optimised to map the audio input into an embedding representation where same class inputs are close and different class inputs are distant, and not optimised to simply predict the correct label. The proposed method can be applied with any back-end feature map extractor, hence, despite the lower binary accuracy, we stick to the choice of 2HeadRawnet 2 for the embedding extraction step.

Regarding the novelty function extraction block, we use $\beta = 2$ for SSM matrix compression. The dimension of the Gaussian kernel is $N = 5$ and the parameter $\epsilon = \sqrt{L/2}$.

As baseline we use the same architecture Rawnet 2 without two head expansion and metric learning loss. The network is trained following the

Dataset	EER	Balanced Acc
$\mathcal{D}_{\text{ASV dev}}$	0.159	0.880
$\mathcal{D}_{\text{ASV eval}}$	0.285	0.767

Table 4.6: *Result of 2HeadRawnet 2 on $\mathcal{D}_{ASV\ dev}$ and $\mathcal{D}_{ASV\ eval}$*

original implementation and then an intermediate layer, i.e., output of the GRU layer, is used as the embedding representation. The novelty extraction step and the peak finding remain unchanged. The choice of this baseline is motivated by the fact that we believe that the crucial novelty of the proposed method is not the network architecture per-se, but the adopted metric learning approach given the task at hand, i.e., splicing detection and localisation.

Detection Results

In the following section we present the results relative to the detection task. To evaluate the detection results, we adopted ROC curves. In particular, given a candidate splicing point $\hat{j}^{\text{spl}}$, correspondent to a peak in the novelty function, we use the value $\Delta(\hat{j}^{\text{spl}})$ as confidence value or soft score for the binary classification problem, i.e., spliced or not spliced. If more than one splicing is present, we simply select the highest peak. On the other side, if no peaks are present we use the global maximum of the novelty function.

We first present the results obtained on $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$ for both the proposed method and the baseline. In this case, only one splicing point is present in the audio track. In Figure 4.8 we present the ROC curve of the proposed method (solid line) against the baseline (dashed line) on $\mathcal{D}_{\text{S1}}$ partitions, $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$. We can first observe that the proposed method strongly outperforms the proposed baseline for both $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$. This experiment proves that the choice of metric learning for the embedding extractor is crucial to obtain meaningful results for the splicing detection task and successful. Simply adopting a classification oriented learning for the embedding extractor does not allow to obtain any significant result. Moreover, detection on the single-splicing dataset created with the development partition of ASVSpoof 2019, i.e., $\mathcal{D}_{\text{S1 dev}}$, is more accurate that the one

on created with evaluation partition, i.e., $\mathcal{D}_{\text{S1 eval}}$. This behaviour is linked to the results of 2HeadRawnet 2 presented in Table 4.6. The network obtains lower binary classification accuracy on the eval partition, which contains a larger number of synthesis algorithms never seen during the training stage. Nonetheless, the splicing detection accuracy on $\mathcal{D}_{\text{S1 dev}}$ partition are very good, reaching a value of AUC $= 0.97$.

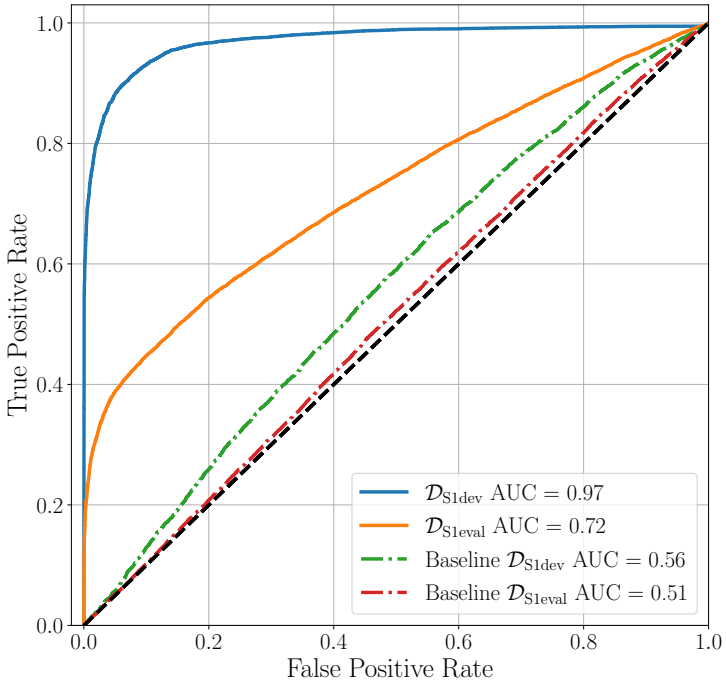


Figure 4.8: *ROC curves and correspondent AUC values for the proposed splicing detection method of partially synthetic speech on $\mathcal{D}_{S1}$. Dashed line curve corresponds to the baseline, solid line curve corresponds to the proposed method.*

In Figure 4.9, we present ROC curves on $\mathcal{D}_{\text{S2}}$, meaning that in this case two splicing points are present in the audio file. The presence of multiple splicing points, and hence multiple peaks in the novelty function, improves the accuracy of the proposed algorithm. We can notice that on $\mathcal{D}_{\text{S2 dev}}$ we obtain almost perfect splicing detection accuracy, and that performances on $\mathcal{D}_{\text{S2 eval}}$ are enhanced w.r.t. Figure 4.8.

The overall results validate the proposed pipeline, and confirms that the choice of metric learning allows to learn an embedding space on which the

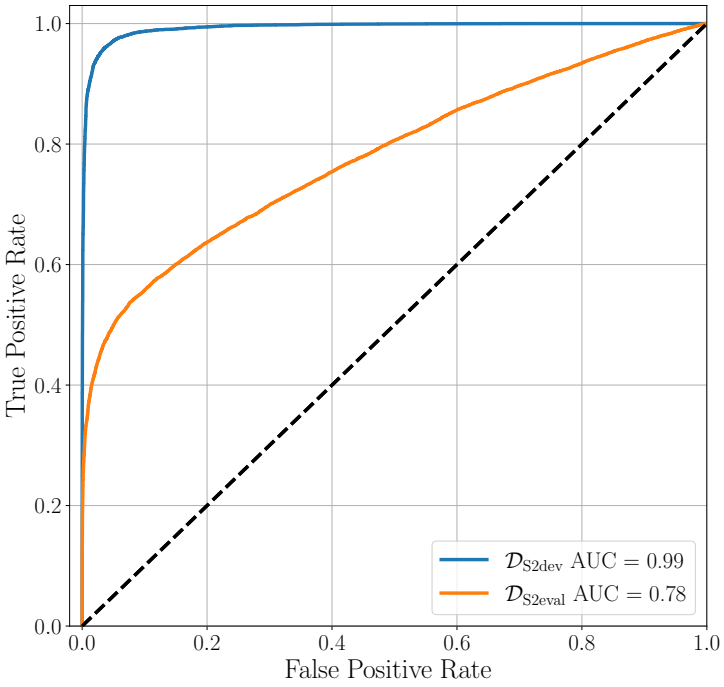


Figure 4.9: *ROC curves and correspondent AUC values for the proposed splicing detection method of partially synthetic speech on $\mathcal{D}_{S2}$.*

proposed distance-based novelty extraction method performs efficiently.

Localisation Results

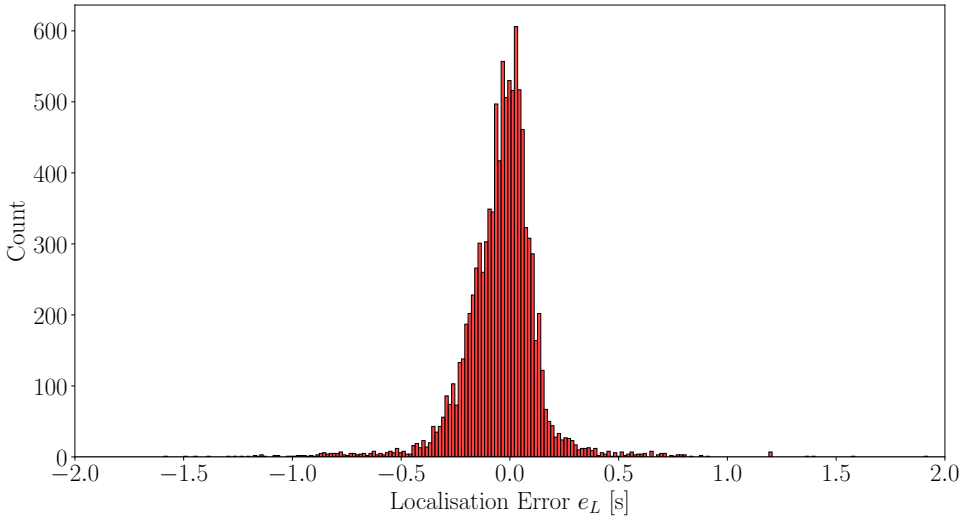
We now focus on the results obtained for the localisation task. To decouple the experiment from the detection task, we analyse only audio files when a splicing is actually present. Given a peak in position $\hat{j}^{\text{spl}}$ of the novelty function $\Delta(j)$, we compute the candidate splicing position in seconds as

$$\hat{t}^{\text{spl}} = \frac{\hat{j}^{\text{spl}} L_h}{F_s} \tag{4.21}$$

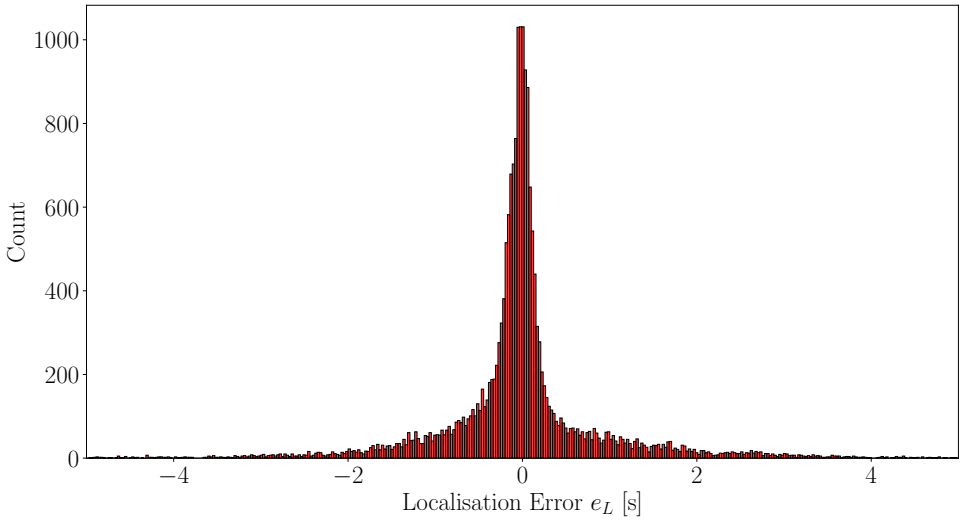
where L_h corresponds to the hop size used for windowing the audio signal and F_s is the sampling frequency. If more two splicing points are present in the track, we order the peaks by their prominence value and we select the first two as candidate splicing points.

The localisation error is simply defined as

$$e_L = t^{\text{spl}} - \hat{t}^{\text{spl}} \tag{4.22}$$



(a) $\mathcal{D}_{S1\ dev}$



(b) $\mathcal{D}_{S1\ eval}$

Figure 4.10: *Histogram of localisation error in one splicing case computed on $\mathcal{D}_{S1\ dev}$ and $\mathcal{D}_{S1\ eval}$ for the proposed splicing localisation method of partially synthetic speech.*

In Figure 4.10 we present the distribution of localisation error e_L in the single-splicing scenario, computed separately for $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$. As already mentioned, tracks from this dataset have been created using only one concatenation operation, i.e., one single splicing point is present. In Figure 4.10a we can observe that localisation error is on average pretty

small, ranging roughly in $[-0.5, 0.5]$ s around the true splicing point t^{spl}. The majority of splicing localisation errors are very close to 0.

On the other side, analogously to what we observed in the splicing detection results, the localisation error on the evaluation dataset $\mathcal{D}_{\text{S1 eval}}$ presented in Figure 4.10b is in a wider range of values, suggesting therefore worse localisation accuracy w.r.t. the previous case.

To better study the performances of the proposed localisation method, we perform a second experiment. We define a tolerance window of length W_T centered in the true splicing point t^{spl}. We then consider correct a localisation prediction if $\hat{t}^{\text{spl}}$ falls inside the tolerance window, i.e., if

$$\hat{t}^{\text{spl}} \in \left[t^{\text{spl}} - \lfloor W_T/2 \rfloor, ..., t^{\text{spl}}, ..., t^{\text{spl}} + \lfloor W_T/2 \rfloor\right]. \tag{4.23}$$

The localisation accuracy is finally defined as the ratio between the correct localisation estimates over the total number of considered spliced tracks.

In Figure 4.11 we report the values of localisation accuracy obtained varying the tolerance window length for the two subset $\mathcal{D}_{\text{S1 dev}}$ and $\mathcal{D}_{\text{S1 eval}}$. We can observe that on $\mathcal{D}_{\text{S1 dev}}$ setting a tolerance window of length of only 0.5 s allows to obtain 0.9 localisation accuracy. As already observed in the other experiments, the accuracy on the development subset is lower and to reach a 0.9 value of the localisation accuracy the tolerance window must be longer (circa 3 s).

We repeated the same experiments for tracks with two splicing points, i.e., three different files have been concatenated. In this case we compute the average of the localisation errors obtained on each track on each splicing point. Therefore, the average localisation error is

$$\begin{aligned} e_{L1} &= \hat{t}_1^{\text{spl}} - t_1^{\text{spl}} \\ e_{L2} &= \hat{t}_2^{\text{spl}} - t_2^{\text{spl}} \\ e_{L\text{avg}} &= (e_{L1} + e_{L2})/2. \end{aligned} \tag{4.24}$$

To define the mapping between each candidate splicing point and the actual target splicing point we simply match each candidate to the closest target. In Figure 4.12 we present the distribution of average localisation error for the two dataset $\mathcal{D}_{\text{S2 dev}}$ and $\mathcal{D}_{\text{S2 eval}}$. We can observe that in general we achieve lower localisation precision w.r.t. the one-splicing case. This behaviour is expected, since, in presence of multiple transitions, the novelty

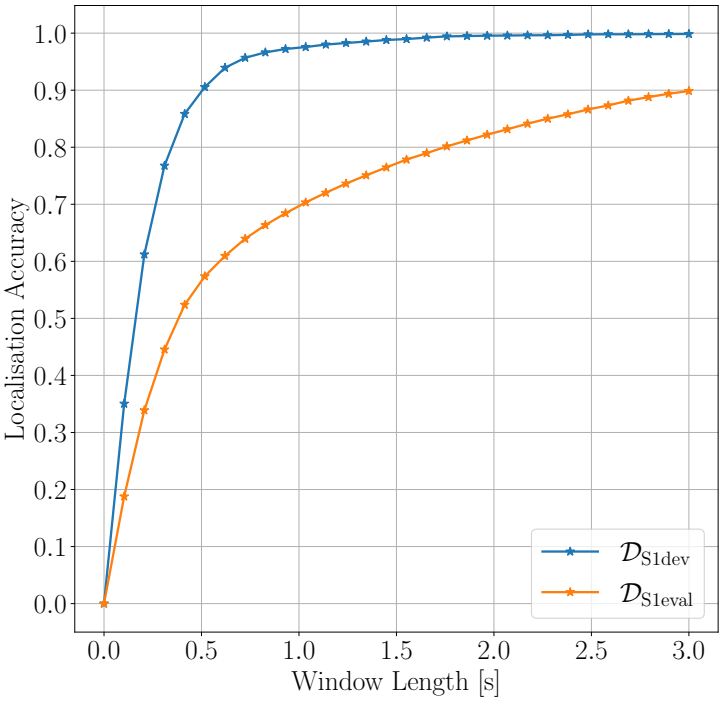
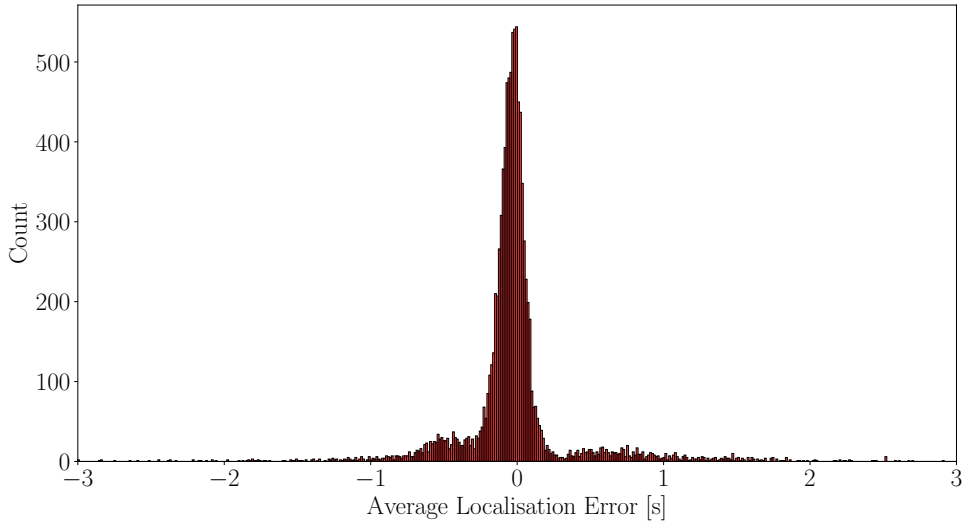


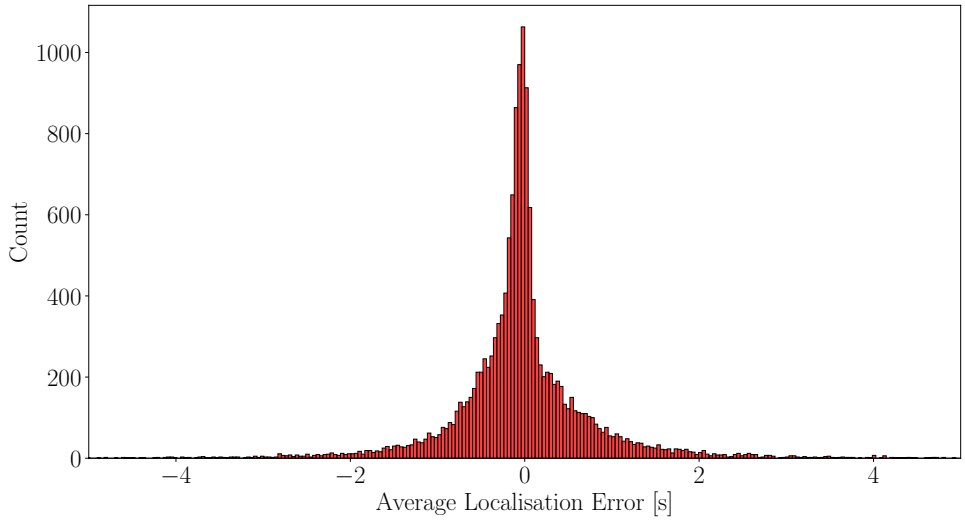
Figure 4.11: *Localisation accuracy for different tolerance window length in the one splicing case on $\mathcal{D}_{S1\ dev}$ and $\mathcal{D}_{S1\ eval}$ for the proposed splicing localisation method of partially synthetic speech.*

function extracted may present more spurious peaks that would affect the localisation effectiveness. Nonetheless, we obtain promising results, especially on $\mathcal{D}_{\text{S2 dev}}$. It is interesting to notice that the histogram in Figure 4.12a shows that part of the samples has an average localisation error around 0.5 s and -0.5 s, hence creating in the histogram plot two peaks around those values. These samples correspond to tracks where only one splicing point out of the two present is correctly localised. The peak finding algorithm associate a probably spurious peak of the novelty function to a second splicing point, obtaining therefore a higher localisation error.

Finally, in Figure 4.13, we repeat the second experiment, counting how many times the estimated splicing points fall inside a window centered around the actual splicing point and computing localisation accuracy for various lengths of the tolerance window. Again, values of the localisation accuracy are smaller in the two-splicing scenario if compared to the val-



(a) $\mathcal{D}_{S2\ dev}$



(b) $\mathcal{D}_{S2\ eval}$

Figure 4.12: *Histogram of average localisation error on each splicing pair in two splicing case computed for $\mathcal{D}_{S2\ dev}$ and $\mathcal{D}_{S2\ eval}$ for the proposed splicing localisation method of partially synthetic speech.*

ues obtained in the single-splicing scenario. Moreover, we confirm that the proposed method performs better on the development dataset rather than for the evaluation dataset. Nonetheless, we observe that setting a tolerance window of length 1.5 s we are able to correctly localise the 90% of the

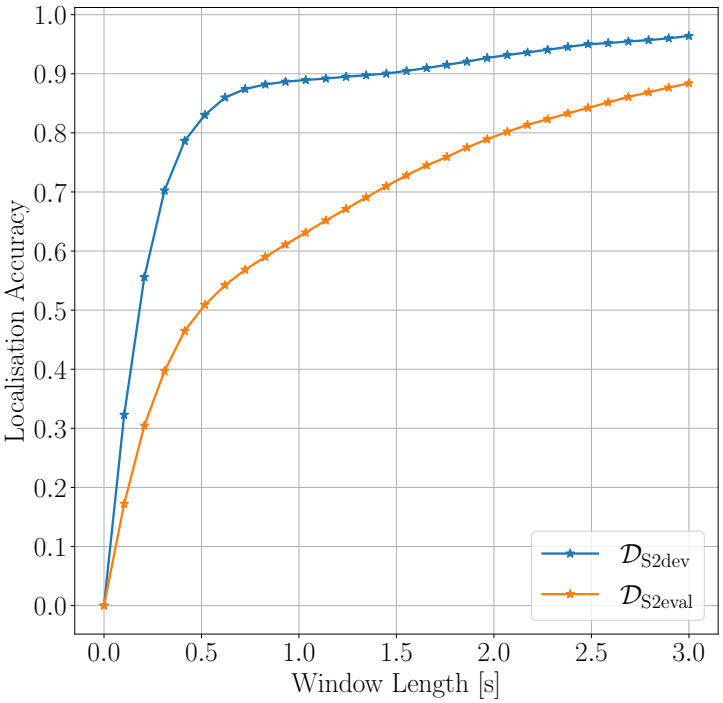


Figure 4.13: *Localisation accuracy computed with different tolerance window lengths in the two-splicing scenario on $\mathcal{D}_{S2\ dev}$ and $\mathcal{D}_{S2\ eval}$ for the proposed splicing localisation method of partially synthetic speech..*

splicing in the $\mathcal{D}_{\text{S2 dev}}$.

4.4.3 Conclusions

In this section we presented a method to detect and localise splicing manipulation with synthetic and real speech fragments. The pipeline is composed of a first embedding extractor block, based on a modification of Rawnet 2 network architecture and training. To address the specific task, we decide to use a metric learning approach, which facilitate the use of distance-based segmentation algorithms for the subsequent steps. In fact, a novelty function is then computed starting from the distance of embeddings extracted for consecutive time frames. When the splicing occurs, i.e., there is a transition between real and synthetic speech, the novelty function has a peak, which can be easily retrieved using a peak finding algorithm. The location of the peak corresponds to the estimated location of the splicing point,

while detection is carried out using a threshold on the novelty function. We test the method on two different datasets, containing one and two splicing respectively. We analyse the performances of detection and localisation separately and both show good results, especially when the set of algorithms used for generating the synthetic fragments is the same used during training of the embedding extractor block. We believe that by selecting a different back-end network for the embedding extractor step, i.e., alternative to Rawnet 2, we can obtain even better results. Therefore, we plan to test different architectures in future works.

4.5 Final Remarks

In this chapter we presented two methods for splicing detection and localisation. The first one estimates the reverberation time of the acoustic environment on sub-bands and over time, exploiting free-decay regions of the recording. From this estimate over time, we define a function, using absolute average difference, that highlights inconsistencies of the reverberation time and therefore detect splicing operation. We test the method on a simulated dataset of pristine and spliced tracks. For the simulation we use both real and simulated RIRs which corresponds to 14 different rooms, then convolved with real speech signals. The results are analysed varying the difference of reverberation time between the two combined recordings and the noise levels. For higher SNR values and for more abrupt changes of reverberation time we obtain almost perfect results in both detection and localisation. The second presented method address a different type of splicing. We assume the spliced audio is a combination of real and synthetic speech, the latter targeting the same speaker present in the real fragment. We extract an embedding representation on short time frames using an end-to-end NN trained with triplet loss. We then transform the embedding sequence in a novelty function, by analysing distance between consequent embeddings. Peak finding is applied to retrieve splicing position and splicing detection is performed through a simple thresholding operation. A dataset is created for the experimental setup and to test the proposed method, including both single and double-spliced tracks. The experiments shows very good detection and localisation performances, especially when the algorithms used

for synthetic speech fragments creation are familiar to the embedding extractor network. The two methods address the same problem formulated in two different contexts. Nonetheless, the two proposal share the same time-dependent approach, that allows not only to detect the splicing, but also to localise it, a problem rarely addressed in the literature.

CHAPTER 5

Conclusions and Future Works

In this thesis we presented three different research perspectives for audio forensics analysis and we proposed solutions that combine classic signal processing techniques with recent ML and DL methodologies. We mainly focused on authenticity assessment and integrity verification tasks, starting from single-channel audio signals, where the main source is usually speech.

In Chapter 2 we addressed the problem of acoustic condition assessment, i.e., the blind estimation of indicators that describe the recording location from an acoustically standpoint. We defined and estimated high-level descriptors, combining both signal processing and neural-network based feature representation and a supervised regression stage, that learns the mapping between the inputs and the parameter values. This framework has allowed to reach higher modelling flexibility and to obtain good prediction performances. In particular, in Section 2.2 we proposed an acoustic parameter similarity measure, that takes into account both reverberation behaviour and noise level of the recording environment. Given a reference

signal, the system analyses the acoustic difference w.r.t. a signal under analysis on a specific parameter, and describes it as a similarity metric. We presented two different strategies, one using a classic distance measure and one using a metric learning strategy. Both methods use a popular pre-trained CNN as feature extractor. Specifically, the second approach has the advantage to learn a compact embedding space where distance measure is meaningful, thanks to the joined training of the feature extractor and parameter similarity estimation. The performances of the regression model has been studied for different SNR value and measured in terms of correlation between the predicted and actual acoustic parameter similarity. We demonstrated that the proposed strategy is successful and we believe that the use of advanced neural network techniques in the future will greatly benefit the audio forensic analysis. Moreover, as future works we plan to consider additional environmental parameters, like the room volume or room geometry, which would contribute to further define each room fingerprint. On the other side, a second possible future work would be to define a single similarity measure that unifies all the acoustic characteristics (reverberation, noise, geometry, ...) in a single value. This approach would probably result in a less-interpretable metric but at the same time would allow to take into account multiple acoustic factors in parallel. Moreover, we used CNN architecture to extract the embedding, but the adoption of attention layers may aid to focus on specific portions of the input where the environment effects are more present (e.g., correspondent to late reverberation). In Section 2.3, we present a reliability measure of automatic transcriptions performed through TTS systems. This method may assist the analysis of large audio corpora acquired through wiretapping or help during the acquisition system setup stage. In this case we select a rich set of features coming from classic audio machine learning literature, in combination with a regression system. To recreate a realistic acoustic scenario, both training and testing set are composed of speech signals augmented with different type of ambient noises ad different levels. The reliability measure is first defined as the correctness, in such challenging noise scenarios, of the automatic transcription with respect to the true transcription. Then, at inference time, it is estimated solely from the audio input. The observed performances are studied globally and for different time granularity, prov-

ing the validity of the designed method, and achieving good performances especially when more time frames predictions are combined. Moreover, we performed an additional experiment using uncontrolled test data, i.e., audio content of real-world talk shows with frequent overlapping voices. This additional experiment has highlighted the ability of the system to deal also with setups very different to the training ones. To further increase the robustness of the method, it is evident the necessity of expanding the set of possible interference signals on one side. On the other side, the inclusion of additional TTS engines would open the possibility to study the effects of noise and interfering signals on different automatic transcription strategies.

In Chapter 3 we addressed a second approach for authenticity verification, namely synthetic speech detection and attribution. This research theme is not lacking of challenges, given the number and the variety of methods for creating synthetic speech. For the synthetic speech detection problem, the investigation proceeded on different analysis levels, starting from the use of low-level features, based on a linear predictive coding analysis of speech, up to the use of high-level features, namely features able to capture semantic key aspects of the speech signal. Low-level based synthetic speech detection, presented in Section 3.2.2, are based on the extraction of a set of features derived by a short-term and long-term analysis of the speech signal. The speech signal is modelled as an auto-regressive process, and the error of such approximation is able to discriminate synthetic speech from bonafide speech. The features are then simply used as input to a supervised binary classifier. This approach is first compared and then combined with a second one, based on bicoherence features. We tested the method on ASVSpoof 2019, a large dataset containing several different synthesis algorithms, and we studied the results obtained using ROC curves and AUC metric. We noticed that, when the set of synthesis algorithm of the test set match the one used in training, we are able to achieve almost perfect classification accuracy. On the other side, when there is a mismatch between the training and testing set the system loses its discriminative potential, even though it is still able to reach acceptable performances. This issue needs to be addressed in future works, looking for a stronger set of features, able to guarantee an higher adaptability in an open set scenario. Nonetheless, this method is suitable whenever no large training data or high

computational capacity is available. The second set of methods are high-level base synthetic speech detection systems, presented in Section 3.2.3. In this case, the feature representation is designed to be able to capture information at an higher semantic level, and to achieve this a transfer-learning approach is adopted. The embedding vector is obtained through complex NN deep architectures, pre-trained on a different task and then used as input to a basic and simple classification algorithm, since our main goal is to understand the discriminative potential of such feature sets rather than optimising the front-end. We presented two methods based on high-level features, EmoSSD and PropospeakerSSD. In the first method, EmoSSD, the embedding represents the emotional quality and intensity expressed by the analysed speech. In the second method, ProsospeakerSSD, two different architectures distillate prosody style on one side and information regarding the speaker identity on the other side, then fused to obtain the final embedding vector. The main difference between the two methods is evident in the evaluation stage, in which the performances are analysed using both ROC curves and binary classification metrics. In fact, even achieving good performances on unknown synthesis algorithms and outperforming the baseline, EmoSSD is not able to deal with speech samples generated through VC methods, i.e., methods that start with real-speech samples and adapt them to match a specific identity. On the other side, ProsospeakerSSD, through the combination of multiple high-level information, is able to discriminate successfully both VC and TTS generated speech samples. These preliminary results show that the fusion of multiple semantic descriptors is a successful recipe, even if it obviously requires huge datasets and the training of multiple components. To test the robustness of the two methods, both training and test dataset have been augmented, considering not only ASVSpoof 2019 dataset but also additional real and synthetic speech corpora, like LibriSpeech or Cloud2019. Moreover, we conducted a study on the robustness of EmoSSD method in presence of additive white noise, which has led to the conclusion that the augmentation of training data is necessary to ensure the robustness of such methods. A possible evolution of both methods is the fusion and joint training of back-end and front-end blocks. In fact, after a separate pre-training, the embedding extractor can be trained jointly with additional classification layers directly on the syn-

thetic speech detection task. This solution would allow to fine-tune the entire network and to probably further increase the overall performances. Nonetheless, we believe that the use of high-level descriptors represents a new exciting perspective for audio forensics analysis. The second task addressed is synthetic speech attribution, i.e., identify which synthesis algorithm has been used to forge the audio input, presented in Section 3.3. This is a less common task, but its solution allows to reconstruct the history of the analysed falsified audio and have a better insight on its origin. The proposed method exploits low-level features and a multiclass supervised classifier. To better match the real-world conditions, two different scenarios are tested, closed-set and open-set. In the first case training and testing share the set of algorithms, while in the open-set case the test set contains speech produced using new synthesis strategies, which should be correctly classified as unknown by the attribution system. The evaluation stage for the closed-set case has revealed satisfying classification accuracy on the majority of the considered classes. The second scenario is of course more challenging, especially when unknown samples corresponds to speech created with end-to-end synthesis methods. Synthetic speech attribution task is far from being solved and need in the future to be further investigated. Possible developments may include the learning of specific synthesis algorithms signature by looking not only at low-level characteristics, but also to high level properties, like the synthesised prosody or speaker.

In Chapter 4, the problem of audio splicing and localisation is faced with two different perspectives. In the first one, presented in Section 4.3 it is assumed that the splicing operation is performed concatenating two recordings acquired in different rooms. Therefore, the splicing is detected and localised by looking at inconsistencies in the reverberation time estimates. In this case a signal-processing algorithms are combined to first extract an estimate of RT over time and on bands. The sequence of estimates are then combined to obtain a function able to highlight changes of the recording environment, finally used for both detection and localisation of the splicing. To assess the method's performances, a dataset of spliced is created using both simulated and real RIRs. The proposed method works correctly on detection and localisation task, especially when the changes in RT are more pronounced. One of the most interesting results is the fact that the best

results are obtained when the speech is convolved with real RIR, proving the robustness of the method and its applicability in real-world scenarios. Moreover, this behaviour shows the limitations of simulation algorithms, which in this specific case is not able to fully model the late reverberations components of the room response. The second part of the chapter is devoted to the presentation of a splicing detection and localisation method for partially synthetic speech samples. In this case, an embedding representation is first extracted locally, using a network originally proposed for the synthetic speech detection task. Then, an algorithm convert the sequence of embeddings in a novelty function, by analysing the distance between consecutive embeddings vectors. The use of metric learning for the training of the embedding network guarantees an embedding space where distance between points is meaningful, and therefore the success of the subsequent algorithm. In this case, we evaluated the method in two setups, first using spliced tracks with a single splicing point, and secondly using double-spliced signals. To prove that the use of triplet loss is key to the method's success, we use as baseline the same embedding extractor network trained using a simple classification loss. Results on the detection first confirm the need of metric learning loss. Moreover, detection is successful in most cases with both single and double splicing, especially when the synthesis algorithms seen during the training match the ones used for spliced audio creation. For the localisation task, better results are obtained in the single splicing case and, again, with known synthesis algorithms. Globally, the evaluation stage confirms the validity of the approach and encourages the use of novel metric learning networks for the task. This choice represents the core novelty of the work, which anyway has been rarely addressed in the literature. A possible evolution of the method may include a joint training of the embedding extractor and the novelty function extraction step, including layers that model the time behaviour of the embeddings, like RNN or attention-layers. This approach would allow to optimise the network also for maximising the detection and localisation accuracy. In general, possible future works include the use of most advanced techniques for splicing detection, for instance minimising phase discontinuities, to further challenge our proposed splicing and detection methods.

Overall, the adoption of data-driven methods for audio forensics anal-

ysis has proved successful in many ways, but at the same time has highlighted some common issues and possible future developments:

- **Limited Training Data** Data-driven methods require large amounts of data to train and to test the systems. For acoustic related tasks, often RIR simulation is exploited to create synthetic responses for several different rooms, but this approach often fails in recreating all the components of the reverberant environment. In fact, as observed in the experiments of our reverberation cues based splicing detection method, the use of real data provides a more reliable performance assessment. Unfortunately, the collection of these kind of audio corpora is time expensive and requires the concrete availability of several recording environment. Regarding synthetic speech, in the last few years this theme has received more and more attention, hence synthetic speech corpora have been released especially for automatic speaker verification challenges, like the mentioned ASVSpoof 2019 dataset. This last dataset includes several different synthesis algorithms but presents an unbalanced ratio between bonafide and spoof samples. For this reason when we implemented the high-level based synthetic speech detection methods we combined different speech datasets containing both real and fake speech samples to reach a reasonable amount of data. Moreover, we proved that the success of synthetic speech attribution methods often relies on the availability of a large amount of samples for each class. Fortunately, the generation of synthetic speech corpora requires less effort compared to the acquisition of real RIRs. Hence, as future works we plan the creation of a new synthetic speech dataset, created through the implementation of recent synthesis algorithms.

- **Generalisation** A common problem shared by most successful forensic solutions is that they are not easily adaptable to new and unseen forgery and manipulation techniques. Often, detection methods based on the analysis of low-level or signal-based traces fail in achieving good generalisation and flexibility. The proposal of high-level semantic approach aims at overcoming this problem. The extraction of contextual information allows to detect anomalies traces left from the forgery of the audio asset, even if the synthesis pipeline is un-

known to the data-driven system. The preliminary inquiry and results reported in this thesis have revealed the success of this approach, which nonetheless requires further investigation in future works.

- **Robustness** In the scenarios addressed in this thesis, often the analysed audio input has been acquired in a not controlled environment or has gone through multiple unpredictable processing, like the ones typically applied by sharing a media on social media platforms. The presence of noise, interfering sources, compression or filtering modifies deeply the characteristics of the audio asset, often affecting the performances of forensic detectors. Nonetheless, a robust authentication or integrity verification method must be resistant to all possible acoustic conditions or processing. In this thesis we often analysed the effect of noise in the functioning of the proposed method. Generally, data-driven methods greatly benefit from the use of augmented data during the training stage, as we proved in the synthetic speech detection case. In fact, this strategy may degrade partially the results but guarantees higher robustness of the detection method. A possible future development may foresee the use of more complex augmentation operations, like multiple compression, re-acquisition, the inclusion of different type of noises or reverberation effects.

- **Anti-forensic Scenario** Anti-forensic techniques are methods that aim at obstructing the audio forensics analysis. They are generally designed to attack specific detectors or to remove the traces left by manipulation and falsification. Moreover, the recent advances of NN has represented not only a great opportunity for audio forensic analysis, but it has also facilitated the development of anti-forensic attacks. For instance, attacker may exploit Generative Adversarial Networks to generate audio assets specifically tailored to exploit the weaknesses of a given detection system. Nonetheless, little or nothing research has been devoted to study audio forensic analysis in adversarial setup. The development of attack-resilient detector is hence paramount and it will surely represent part of the future works in the audio forensic research community.

Bibliography

[1] S. Agarwal, H. Farid, T. El-Gaaly, and S.-N. Lim. Detecting deep-fake videos from appearance and behavior. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2020.

[2] Y. Agiomyrgiannakis. Vocaine the vocoder and applications in speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[3] E. A. AlBadawy, S. Lyu, and H. Farid. Detecting AI-synthesized speech using bispectral analysis. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[4] J. B. Allen and D. A. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America (JASA)*, 65:943–950, 1979.

[5] S. Ö. Arık, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, A. Ng, J. Raiman, et al. Deep voice: Real-time neural text-to-speech. In *International Conference on Machine Learning (ICML)*, 2017.

[6] L. M. Arslan. Speaker transformation algorithm using segmental codebooks (STASC). *Speech Communication*, 28(3):211–226, 1999.

[7] L. Attorresi, D. Salvi, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Combining speaker identification and prosody analysis for synthetic speech detection. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (Submitted)*, 2022.

[8] R. E. Baker and A. R. Bradlow. Variability in word duration as a function of probability, speech style, and prosody. *Language and Speech*, 52(4):391–413, 2009.

[9] BBC News. Deepfake app causes fraud and privacy fears in China, 2019.

[10] D. Berthelot, T. Schumm, and L. Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv:1703.10717*, 2017.

[11] A. Bertrand. Applications and trends in wireless acoustic sensor networks: A signal processing perspective. In *IEEE Symposium on Communications and Vehicular technology (SCVT)*, 2011.

[12] V. Bhat, I. Sengupta, and A. Das. An adaptive audio watermarking based on the singular value decomposition in the wavelet domain. *Digital Signal Processing*, 20(6):1547–1558, 2010.

[13] T. Bianchi and A. Piva. Image forgery localization via block-grained analysis of JPEG artifacts. *IEEE Transactions on Information Forensics and Security (TIFS)*, 7(3):1003–1017, 2012.

[14] T. Bianchi, A. D. Rosa, M. Fontani, G. Rocciolo, and A. Piva. Detection and localization of double compression in mp3 audio tracks. *EURASIP Journal on Information Security*, 2014(1):1–14, 2014.

[15] A. Black and N. Campbell. Optimising selection of units from speech databases for concatenative synthesis. In *EUROSPEECH*, 1995.

[16] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro. Video Face Manipulation Detection Through Ensemble of CNNs. In *International Conference on Pattern Recognition (ICPR)*, 2020.

[17] C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Automatic reliability estimation for speech audio surveillance recordings. In *IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2019.

[18] C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Synthetic speech detection through short-term and long-term prediction traces. *EURASIP Journal on Information Security*, 2021:1–14, 2021.

[19] C. Borrelli, A. Canclini, F. Antonacci, A. Sarti, and S. Tubaro. A denoising methodology for higher order ambisonics recordings. In *IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2018.

[20] M. Brundage, S. Avin, J. Clark, H. Toner, P. Eckersley, B. Garfinkel, A. Dafoe, P. Scharre, T. Zeitzoff, B. Filar, et al. The malicious use of artificial intelligence: forecasting, prevention, and mitigation. *arXiv:1802.07228*, 2018.

[21] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan. IEMOCAP: Interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42:335–359, 2008.

[22] D. Capoferri, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. Speech audio splicing detection and localization exploiting reverberation cues. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2020.

[23] F. Castelli, D. Salvi, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. A metric learning approach to synthetic speech splicing detection and localisation. In *European Signal Processing Conference (EUSIPCO) (Submitted)*, 2022.

[24] M. Chen, X. He, J. Yang, and H. Zhang. 3-D convolutional recurrent neural networks with attention model for speech emotion recognition. *IEEE Signal Processing Letters*, 25:1440–1444, 2018.

[25] T. Chen, A. Kumar, P. Nagarsheth, G. Sivaraman, and E. Khoury. Generalization of audio deepfake detection. In *Odyssey Speaker and Language Recognition Workshop*, 2020.

[26] S. Cherubin, C. Borrelli, M. Zanoni, M. Buccoli, A. Sarti, and S. Tubaro. Three-dimensional mapping of high-level music features for music browsing. In *IEEE International Workshop on Multilayer Music Representation and Processing (MMRP)*, 2019.

[27] M. Cinelli, G. D. F. Morales, A. Galeazzi, W. Quattrociocchi, and M. Starnini. The echo chamber effect on social media. *Proceedings of the National Academy of Sciences*, 118(9), 2021.

[28] G. Colombetti. From affect programs to dynamical discrete emotions. *Philosophical Psychology*, 22:407–425, 2009.

[29] E. Conti, D. Salvi, C. Borrelli, B. Hosler, P. Bestagini, F. Antonacci, A. Sarti, M. C. Stamm, and S. Tubaro. Deepfake speech detection through emotion recognition: A semantic approach. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (Accepted)*, 2022.

[30] A. J. Cooper. The electric network frequency (enf) as an aid to authenticating forensic digital audio recordings–an automated approach. In *Audio Engineering Society Conference*, 2008.

[31] A. J. Cooper. Detecting butt-spliced edits in forensic digital audio recordings. In *AES International Conference: Audio Forensics: Practices and Challenges*, 2010.

[32] L. Cuccovillo, S. Mann, M. Tagliasacchi, and P. Aichroth. Audio tampering detection via microphone classification. In *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2013.

[33] T. de M. Prego, A. A. de Lima, R. Zambrano-Lopez, and S. L. Netto. Blind estimators for reverberation time and direct-to-reverberant energy ratio using subband speech decomposition. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2015.

[34] S. Desai, E. V. Raghavendra, B. Yegnanarayana, A. W. Black, and K. Prahallad. Voice conversion using artificial neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

[35] B. Desplanques, J. Thienpondt, and K. Demuynck. Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2020.

[36] H. Dinkel, N. Chen, Y. Qian, and K. Yu. End-to-end spoofing detection with raw waveform CLDNNS. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[37] H. Dudley. Remaking speech. *The Journal of the Acoustical Society of America (JASA)*, 11:169–177, 1939.

[38] J. Eaton, N. D. Gaubitch, A. H. Moore, and P. A. Naylor. The ace challenge - corpus description and performance evaluation. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2015.

[39] P. A. A. Esquef, J. A. ApolinÃ¡rio, and L. W. P. Biscainho. Improved edit detection in speech via enf patterns. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2015.

[40] F. Eyben, S. Buchholz, N. Braunschweiler, J. Latorre, V. Wan, M. J. Gales, and K. Knill. Unsupervised clustering of emotion and voice styles for expressive tts. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[41] T. H. Falk, V. Parsa, J. F. Santos, K. Arehart, O. Hazrati, R. Huber, J. M. Kates, and S. Scollie. Objective quality and intelligibility prediction for users of assistive listening devices: Advantages and limitations of existing tools. *IEEE Signal Processing Magazine (SPM)*, 32(2):114–124, 2015.

[42] T. H. Falk, C. Zheng, and W.-Y. Chan. A non-intrusive quality and intelligibility measure of reverberant and dereverberated speech. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 18(7):1766–1774, 2010.

[43] G. Fant. The source filter concept in voice production. *Speech Transmission Laboratory. Quarterly Progress and Status Reports*, 1:21–37, 1981.

[44] J. Fernandes, F. Teixeira, V. Guedes, A. Junior, and J. P. Teixeira. Harmonic to noise ratio measurement-selection of window and length. *Procedia Computer Science*, 138:280–285, 2018.

[45] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory. Using deep bidirectional recurrent neural networks for prosodic-target prediction in a unit-selection text-to-speech system. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2015.

[46] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2000.

[47] J. Franke. A Levinson-Durbin recursion for autoregressive-moving average processes. *Biometrika*, 72:573–581, 1985.

[48] N. R. French and J. C. Steinberg. Factors governing the intelligibility of speech sounds. *The Journal of the Acoustical Society of America (JASA)*, 19(1):90–119, 1947.

[49] T. Fukumori, M. Morise, and T. Nishiura. Performance estimation of reverberant speech recognition based on reverberant criteria rsr-dn with acoustic parameters. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2010.

[50] H. Gamper and I. J. Tashev. Blind reverberation time estimation using a convolutional neural network. In *IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2018.

[51] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren. Darpa timit acoustic phonetic continuous speech corpus cdrom, 1993.

[52] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[53] A. F. Genovese, H. Gamper, V. Pulkki, N. Raghuvanshi, and I. J. Tashev. Blind room volume estimation from single-channel noisy speech. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[54] R. L. Goldsworthy and J. E. Greenberg. Analysis of speech-based speech transmission index methods with implications for nonlinear operations. *The Journal of the Acoustical Society of America (JASA)*, 116(6):3679–3689, 2004.

[55] D. Griffin and Jae Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing (TASLP)*, 32:236–243, 1984.

[56] C. Grigoras. Applications of enf analysis in forensic authentication of digital audio and video recordings. *Journal of the Audio Engineering Society*, 57(9):643–661, 2009.

[57] A. Guarino. Digital forensics as a big data challenge. In *ISSE Securing Electronic Business Processes*, pages 197–203. Springer, 2013.

[58] D. Güera, S. Baireddy, P. Bestagini, S. Tubaro, and E. J. Delp. We need no pixels: Video manipulation detection using stream descriptors. *arXiv:1906.08743*, 2019.

[59] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[60] M. Harper. The automatic speech recogition in reverberant environments (aspire) challenge. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[61] R. G. Hautamäki and T. Kinnunen. Why did the x-vector system miss a target speaker? impact of acoustic mismatch upon target score on voxceleb data. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2020.

[62] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. Cnn architectures for large-scale audio classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[63] B. Hosler, D. Salvi, A. Murray, F. Antonacci, P. Bestagini, S. Tubaro, and M. C. Stamm. Do deepfakes feel emotions? A semantic approach to detecting deepfakes via emotional inconsistencies. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021.

[64] E. Howcroft. How faking videos became easy and why that's so scary. *Bloomberg: New York, NY, USA*, 2018.

[65] C. Hsu, H. Hwang, Y. Wu, Y. Tsao, and H. Wang. Voice conversion from non-parallel corpora using variational auto-encoder. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016.

[66] G. Hua, H. Liao, Q. Wang, H. Zhang, and D. Ye. Detection of electric network frequency in audio recordings–from theory to practical detectors. *IEEE Transactions on Information Forensics and Security (TIFS)*, 16:236–248, 2020.

[67] G. Hua, H. Liao, H. Zhang, D. Ye, and J. Ma. Robust enf estimation based on harmonic enhancement and maximum weight clique. *IEEE Transactions on Information Forensics and Security (TIFS)*, 16:3874–3887, 2021.

[68] A. Huang. Similarity measures for text document clustering. In *New Zealand Computer Science Research Student Conference (NZCSRSC)*, 2008.

[69] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky. PocketSphinx: A free, real-time continuous speech recognition system for hand-held devices. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2006.

[70] A. J. Hunt and A. W. Black. Unit selection in a concatenative speech synthesis system using a large speech database. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 1996.

[71] R. T. Irene, C. Borrelli, M. Zanoni, M. Buccoli, and A. Sarti. Automatic playlist generation using convolutional neural networks and recurrent neural networks. In *European Signal Processing Conference (EUSIPCO)*, 2019.

[72] A. Janicki. Spoofing countermeasure based on analysis of linear prediction error. In *The International Speech Communication Association (ISCA)*, 2015.

[73] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, z. Chen, P. Nguyen, R. Pang, I. Lopez Moreno, and Y. Wu. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[74] Z. Jin, J. Cao, Y. Zhang, J. Zhou, and Q. Tian. Novel visual and statistical image features for microblogs news verification. *IEEE Transactions on Multimedia*, 19(3):598–608, 2016.

[75] S. Jørgensen, J. Cubick, and T. Dau. Speech intelligibility evaluation for mobile phones. *Acta Acustica United with Acustica*, 101(5):1016–1025, 2015.

[76] M. Kajstura, A. Trawinska, and J. Hebenstreit. Application of the electrical network frequency (enf) criterion: A case of a digital recording. *Forensic Science International*, 155(2-3):165–171, 2005.

[77] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu. Efficient neural audio synthesis. In *International Conference on Machine Learning (ICML)*, 2018.

[78] M. R. Kamble, H. B. Sailor, H. A. Patil, and H. Li. Advances in anti-spoofing: from the perspective of ASVspoof challenges. *APSIPA Transactions on Signal and Information Processing*, 2020.

[79] T. Kaneko and H. Kameoka. Cyclegan-vc: Non-parallel voice conversion using cycle-consistent adversarial networks. In *European Signal Processing Conference (EUSIPCO)*, 2018.

[80] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo. Cyclegan-vc2: Improved cyclegan-based non-parallel voice conversion. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[81] M. Karbasi, A. H. Abdelaziz, and D. Kolossa. Twin-hmm-based non-intrusive speech intelligibility prediction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[82] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[83] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019.

[84] J. M. Kates and K. H. Arehart. Coherence and the speech intelligibility index. *The Journal of the Acoustical Society of America (JASA)*, 117(4):2224–2237, 2005.

[85] H. Kawahara. Straight, exploitation of the other aspect of vocoder: Perceptually isomorphic decomposition of speech sounds. *Acoustical Science and Technology*, 27(6):349–353, 2006.

[86] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne. Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds. *Speech Communication*, 27:187–207, 1999.

[87] Keith Ito and Linda Johnson. The LJSpeech dataset, 2017.

[88] K. Khaldi and A.-O. Boudraa. Audio watermarking via emd. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 21:675–680, 2012.

[89] T. Kinnunen, J. Lorenzo-Trueba, J. Yamagishi, T. Toda, D. Saito, F. Villavicencio, and Z. Ling. A spoofing benchmark for the 2018 voice conversion challenge: leveraging from spoofing countermeasures for speech artifact assessment. In *The Speaker and Language Recognition Workshop*, 2018.

[90] D. Kirovski and H. Malvar. Robust spread-spectrum audio watermarking. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2001.

[91] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[92] K. Kobayashi, T. Toda, and S. Nakamura. Intra-gender statistical singing voice conversion with direct waveform modification using log-spectral differential. *Speech Communication*, 99:211–220, 2018.

[93] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015.

[94] P. Korshunov and S. Marcel. Speaker inconsistency detection in tampered video. In *European Signal Processing Conference (EUSIPCO)*, 2018.

[95] R. Korycki. Authenticity examination of compressed audio recordings using detection of multiple compression and encoders identification. *Forensic Science International*, 238:33–46, 2014.

[96] N. Krishnamurthy and J. H. Hansen. Babble noise: modeling, analysis, and applications. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 17(7):1394–1407, 2009.

[97] K. D. Kryter. Methods for the calculation and use of the articulation index. *The Journal of the Acoustical Society of America (JASA)*, 34(11):1689–1697, 1962.

[98] N. G. La Vigne, S. S. Lowry, J. A. Markman, and A. M. Dwyer. Evaluating the use of public surveillance cameras for crime control and prevention. Urban Institute (research report),

2011. https://www.urban.org/research/publication/evaluating-use-public-surveillance-cameras-crime-control-and-prevention.

[99] A. Lieto, D. Moro, F. Devoti, C. Parera, V. Lipari, P. Bestagini, and S. Tubaro. "Hello? Who Am I Talking to?" A Shallow CNN Approach for Human vs. Bot Speech Classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[100] X. Lin and X. Kang. Exposing speech tampering via spectral phase analysis. *Digital Signal Processing*, 60:63–74, 2017.

[101] H. Liu, J. Shi, J. Huang, Q. Zhou, S. Wei, B. Li, and X. Yuan. Single-mode wild area surveillance sensor with ultra-low power design based on microphone array. *IEEE Access*, 7:78976–78990, 2019.

[102] D. Looney and N. D. Gaubitch. Joint estimation of acoustic parameters from single-microphone speech observations. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[103] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv:1508.04025*, 2015.

[104] R. C. Maher. Audio forensic examination. *IEEE Signal Processing Magazine*, 26(2):84–94, 2009.

[105] H. Malik. Acoustic environment identification and its applications to audio forensics. *IEEE Transactions on Information Forensics and Security (TIFS)*, 8:1827–1837, 2013.

[106] H. Malik and H. Zhao. Recording environment identification using acoustic reverberation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1833–1836, 2012.

[107] T. Masuko, K. Tokuda, T. Kobayashi, and S. Imai. Speech synthesis using HMMs with dynamic features. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1996.

[108] D. Matrouf, J. Bonastre, and C. Fredouille. Effect of speech transformation on impostor acceptance. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2006.

[109] B. Mc Fee, C. Raffel, D. Liang, D. Ellis, M. Mc Vicar, E. Battenberg, and O. Nieto. LibROSA: Audio and Music Signal Analysis in Python. In *Python in Science Conference (PySci)*, 2015.

[110] S. Milani, P. F. Piazza, P. Bestagini, and S. Tubaro. Audio tampering detection using multimodal features. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[111] H. Ming, D.-Y. Huang, L. Xie, J. Wu, M. Dong, and H. Li. Deep bidirectional LSTM modeling of timbre and prosody for emotional voice conversion. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2016.

[112] A. H. Moore, M. Brookes, and P. A. Naylor. Roomprints for forensic audio applications. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2013.

[113] M. Morise, F. Yokomori, and K. Ozawa. WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, 99:1877–1884, 2016.

[114] E. Moulines and F. Charpentier. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9:453–467, 1990.

[115] L. Muda, B. KM, and I. Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *Journal of Computing*, 2(3):138–143, 2010.

[116] P. A. Naylor and N. D. Gaubitch. *Speech dereverberation*. Springer Science & Business Media, 2010.

[117] P. A. Naylor and N. D. Gaubitch. Acoustic signal processing in noise: It's not getting any quieter. In *IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2012.

[118] E. Onur, C. Ersoy, H. Deliç, and L. Akarun. Surveillance wireless sensor networks: Deployment quality analysis. *IEEE Network*, 21(6):48–53, 2007.

[119] X. Pan, X. Zhang, and S. Lyu. Detecting splicing in digital audios using local noise level estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[120] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an ASR corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[121] S. P. Panda and A. K. Nayak. A waveform concatenation technique for text-to-speech synthesis. *International Journal of Speech Technology*, 20:959–976, 2017.

[122] M. Papa, C. Borrelli, P. Bestagini, F. Antonacci, A. Sarti, and S. Tubaro. A data-driven approach for acoustic parameter similarity estimation of speech recording. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (Accepted)*, 2022.

[123] P. P. Parada, D. Sharma, and P. A. Naylor. Non-intrusive estimation of the level of reverberation in speech. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[124] P. P. Parada, D. Sharma, T. van Waterschoot, and P. A. Naylor. Evaluating the non-intrusive room acoustics algorithm with the ace challenge. In *ACE Challenge Workshop*, 2015.

[125] N. I. Park, J. W. Lee, K.-S. Shim, J. S. Byun, and O.-Y. Jeon. A method of forensic authentication of audio recordings generated using the voice memos application in the iphone. *Forensic Science International*, 320:110702, 2021.

[126] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)*, 12:2825–2830, 2011.

[127] K. Peng, W. Ping, Z. Song, and K. Zhao. Non-autoregressive neural text-to-speech. In *International Conference on Machine Learning (ICML)*, 2020.

[128] G. Picardi, C. Borrelli, A. Sarti, G. Chimienti, and M. Calisti. A minimal metric for the characterization of acoustic noise emitted by underwater vehicles. *Sensors*, 20(22):6644, 2020.

[129] W. Ping, K. Peng, A. Gibiansky, S. Ö. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller. Deep voice 3: 2000-speaker neural text-to-speech. In *International Conference on Machine Learning (ICML)*, 2018.

[130] S. Qi, Z. Huang, Y. Li, and S. Shi. Audio recording device identification based on deep learning. In *IEEE International Conference on Signal and Image Processing (ICSIP)*, pages 426–431, 2016.

[131] R. Scheibler, E. Bezzam, and I. Dokmanic. Pyroomacoustics: a python package for audio room simulation and array processing algorithms. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[132] M. Ravanelli and Y. Bengio. Speaker recognition from raw waveform with sincnet. In *IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018.

[133] M. K. Reddy and K. S. Rao. Robust pitch extraction method for the HMM-based speech synthesis system. *IEEE Signal Processing Letters*, 24:1133–1137, 2017.

[134] K. S. Rhebergen and N. J. Versfeld. A speech intelligibility index-based approach to predict the speech reception threshold for sentences in fluctuating noise for normal-hearing listeners. *The Journal of the Acoustical Society of America (JASA)*, 117(4):2181–2192, 2005.

[135] D. P. N. Rodríguez, J. A. Apolinário, and L. W. P. Biscainho. Audio authenticity: Detecting ENF discontinuity with high precision phase analysis. *IEEE Transactions on Information Forensics and Security (TIFS)*, 5(3):534–543, 2010.

[136] M. Sahidullah, T. Kinnunen, and C. Hanilçi. A comparison of features for synthetic speech detection. In *The International Speech Communication Association (ISCA)*, 2015.

[137] M. Scharkow, F. Mangold, S. Stier, and J. Breuer. How social network sites and other online intermediaries increase exposure to news. *Proceedings of the National Academy of Sciences*, 117(6):2761–2763, 2020.

[138] M. Schröder, M. Charfuelan, S. Pammi, and I. Steiner. Open source voice creation toolkit for the MARY TTS platform. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2011.

[139] M. R. Schroeder. New Method of Measuring Reverberation Time. *The Journal of the Acoustical Society of America (JASA)*, 37:409–412, 1965.

[140] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[141] A. Sehr, E. A. Habets, R. Maas, and W. Kellermann. Towards a better understanding of the effect of reverberation on speech recognition performance. In *IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2010.

[142] D. Sharma, P. A. Naylor, and M. Brookes. Non-intrusive speech intelligibility assessment. In *European Signal Processing Conference (EUSIPCO)*, 2013.

[143] D. Sharma, Y. Wang, P. A. Naylor, and M. Brookes. A data-driven non-intrusive measure of speech quality and intelligibility. *Speech Communication*, 80:84–94, 2016.

[144] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, et al. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[145] B. Sisman, J. Yamagishi, S. King, and H. Li. An overview of voice conversion and its challenges: From statistical modeling to deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 29:132–157, 2020.

[146] R. Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. Weiss, R. Clark, and R. A. Saurous. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. In *International Conference on Machine Learning (ICML)*, 2018.

[147] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust DNN embeddings for speaker recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[148] A. Spanias, T. Painter, and V. Atti. *Linear Prediction in Narrowband and Wideband Coding*, pages 91–112. Wiley, 2007.

[149] H. J. M. Steeneken and T. Houtgast. A physical method for measuring speech-transmission quality. *The Journal of the Acoustical Society of America (JASA)*, 67(1):318–326, 1980.

[150] Y. Stylianou, O. Cappé, and E. Moulines. Continuous probabilistic transform for voice conversion. *IEEE Transactions on Speech and Audio Processing*, 6(2):131–142, 1998.

[151] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[152] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen. An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 19(7):2125–2136, 2011.

[153] H. Tachibana, K. Uenoyama, and S. Aihara. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[154] A. Taha, Y.-T. Chen, T. Misu, A. Shrivastava, and L. Davis. Boosting standard classification architectures through a ranking regularizer. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020.

[155] S. Tahir and W. Iqbal. Big data an evolving concern for forensic investigators. In *International Conference on Anti-Cybercrime (ICACC)*, 2015.

[156] H. Tak, J. Patino, M. Todisco, A. Nautsch, N. Evans, and A. Larcher. End-to-end anti-spoofing with RawNet2. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[157] K. Tanaka, H. Kameoka, T. Kaneko, and N. Hojo. WaveCycleGAN2: Time-domain neural post-filter for speech waveform generation. *arXiv:1904.02892*, 2019.

[158] K. Tanaka, H. Kameoka, T. Kaneko, and N. Hojo. Atts2s-vc: Sequence-to-sequence voice conversion with attention and context preservation mechanisms. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[159] M. Todisco, H. Delgado, and N. Evans. Constant Q cepstral coefficients: a spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45:516–535, 2017.

[160] M. Todisco, X. Wang, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. Kinnunen, and K. A. Lee. ASVspoof 2019: Future horizons in spoofed and fake audio detection. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2019.

[161] K. Tokuda, H. Zen, and A. W. Black. An HMM-based speech synthesis system applied to english. In *IEEE Speech Synthesis Workshop*, 2002.

[162] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti. Scream and gunshot detection and localization for audio-surveillance systems. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[163] J.-M. Valin and J. Skoglund. Lpcnet: Improving neural speech synthesis through linear prediction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[164] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. In *ISCA Workshop on Speech Synthesis Workshop*, page 125, 2016.

[165] J. Vieira. Automatic estimation of reverberation time. In *Audio Engineering Society Convention*, 2004.

[166] W. Wang, S. Xu, B. Xu, et al. First step towards end-to-end parametric tts synthesis: Generating spectral parameters with neural attention. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2016.

[167] X. Wang, J. Lorenzo-Trueba, S. Takaki, L. Juvela, and J. Yamagishi. A comparison of recent waveform generation and acoustic modeling methods for neural-network-based speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[168] X. Wang, S. Takaki, and J. Yamagishi. Neural source-filter-based waveform model for statistical parametric speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[169] X. Wang, J. Yamagishi, M. Todisco, H. Delgado, A. Nautsch, N. Evans, M. Sahidullah, V. Vestman, T. Kinnunen, K. A. Lee, L. Juvela, P. Alku, Y.-H. Peng, H.-T. Hwang, Y. Tsao, H.-M. Wang, S. L. Maguer, M. Becker, F. Henderson, R. Clark, Y. Zhang, Q. Wang, Y. Jia, K. Onuma, K. Mushika, T. Kaneda, Y. Jiang, L.-J. Liu, Y.-C. Wu, W.-C. Huang, T. Toda, K. Tanaka, H. Kameoka, I. Steiner, D. Matrouf, J.-F. Bonastre, A. Govender, S. Ronanki, J.-X. Zhang, and Z.-H. Ling. ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech. *Computer Speech & Language*, 64:101–114, 2020.

[170] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *arXiv:1703.10135*, 2017.

[171] Y. Wang, D. Stanton, Y. Zhang, R.-S. Ryan, E. Battenberg, J. Shor, Y. Xiao, Y. Jia, F. Ren, and R. A. Saurous. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *International Conference on Machine Learning (ICML)*, 2018.

[172] J. Weigold, T. Brosnihan, J. Bergeron, and X. Zhang. A MEMS condenser microphone for consumer applications. In *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, 2006.

[173] Z. Wu, O. Watts, and S. King. Merlin: An open source neural network speech synthesis system. In *Speech Synthesis Workshop (SSW)*, 2016.

[174] X. Xiao, X. Tian, S. Du, H. Xu, E. S. Chng, and H. Li. Spoofing speech detection using high dimensional magnitude and phase features: the NTU approach for ASVspoof 2015 challenge. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[175] F. Xiong, S. Goetze, B. Kollmeier, and B. T. Meyer. Joint estimation of reverberation time and early-to-late reverberation ratio from single-channel speech signals. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 27(2):255–267, 2018.

[176] J. Yamagishi, C. Veaux, K. MacDonald, et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92), 2019.

[177] J. Yamagishi, X. Wang, M. Todisco, M. Sahidullah, J. Patino, A. Nautsch, X. Liu, K. A. Lee, T. Kinnunen, N. Evans, et al. ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection. *arXiv:2109.00537*, 2021.

[178] W. Yao, J. Zhao, M. J. Till, S. You, Y. Liu, Y. Cui, and Y. Liu. Source location identification of distribution-level electric network frequency signals at multiple geographic scales. *IEEE Access*, 5:11166–11175, 2017.

[179] J. Yi, Y. Bai, J. Tao, H. Ma, Z. Tian, C. Wang, T. Wang, and R. Fu. Half-Truth: A Partially Fake Audio Detection Dataset. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2021.

[180] M. Zakariah, M. K. Khan, and H. Malik. Digital multimedia audio forensics: past, present and future. *Multimedia Tools and Applications*, 77(1):1009–1040, 2018.

[181] H. Zen, Y. Agiomyrgiannakis, N. Egberts, F. Henderson, and P. Szczepaniak. Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices. In *Conference of the International Speech Communication Association (INTERSPEECH)*, 2016.

[182] H. Zen and H. Sak. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[183] C. Zhang, C. Yu, and J. H. Hansen. An investigation of deep-learning frameworks for speaker verification antispoofing. *IEEE Journal of Selected Topics in Signal Processing*, 11:684–694, 2017.

[184] H. Zhao, Y. Chen, R. Wang, and H. Malik. Anti-forensics of environmental-signature-based audio splicing detection and its countermeasure via rich-features classification. *IEEE Transactions on Information Forensics and Security (TIFS)*, 11(7):1603–1617, 2016.

[185] H. Zhao, Y. Chen, R. Wang, and H. Malik. Audio splicing detection and localization using environmental signature. *Multimedia Tools and Applications*, 76(12):13897–13927, 2017.

[186] X. Zhou and R. Zafarani. Fake news: A survey of research, detection methods, and opportunities. *arXiv:1812.00315*, 2, 2018.

[187] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017.