EURASIP Journal on Information Security

RESEARCH Open Access

Synthetic speech detection through short-term and long-term prediction traces

Clara Borrelli*, Paolo Bestagini, Fabio Antonacci, Augusto Sarti and Stefano Tubaro

Abstract
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. As these methods generate convincing fake human voices, they can be used in a malicious way to negatively impact on today's society (e.g., people impersonation, fake news spreading, opinion formation). For this reason, the ability of detecting whether a speech recording is synthetic or pristine is becoming an urgent necessity. In this work, we develop a synthetic speech detector. This takes as input an audio recording, extracts a series of hand-crafted features motivated by the speech-processing literature, and classify them in either closed-set or open-set. 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 in the forensics literature.

Keywords: Forensics, Audio, Speech, Deepfake

1 Introduction

The possibility of manipulating digital multimedia objects is within everyone's reach. Since a few years ago, this was possible thanks to several user-friendly software suites enabling audio, image, and video editing. Nowadays, media manipulation has become even easier thanks to the use of mobile apps that perform automatic operations such as face-swaps, lip-syncing, and audio auto-tune. Moreover, the huge technological advances determined by deep learning has delivered a series of artificial intelligence (AI)-driven tools that make manipulations extremely realistic and convincing.

All of these tools are surely a great asset in a digital artist's arsenal. However, if used maliciously to generate fake media, they can have a strong and negative social impact. A recent example of synthetically manipulated media that raised a lot of concern is that of deepfakes [1, 2]. Indeed, deepfake AI-driven technology enables replacing one person's identity with someone else in a video [3]. This has been used to disseminate fake news through politician impersonation as well as for revenge porn distribution.

If malicious use of deepfakes is a threat per se, deepfake deception power increases even more when paired with synthetic speech generation techniques. Indeed, synthetic generation of both a video and an audio track opens the doors to new kinds of frauds, security breaches, and convincing fake news spreading methods. However, despite multiple forensic detectors have been proposed for video deepfake analysis [4–7], only a few techniques have been tailored to AI-generated speech analysis [8, 9]. For this reason, in this paper we focus on synthetic audio speech detection.

The problem of synthetic speech detection is particularly challenging due to the wide variety of available methods for fake speech generation. Indeed, synthetic speech can be obtained by simple cut-and-paste techniques performing waveform concatenation [10], in some cases available as open source toolkit. Alternatively, it can be obtained by vocoders exploiting the source-filter model of speech signal [11]. More recently, even multiple

*Correspondence: clara.borrelli@polimi.it
Dipartimento di Elettronica, Informazione e Biongegneria - Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133 Milano, IT, Italy

convolutional neural networks (CNNs)-based methods for synthetic audio generation have been proposed [12]. These produce extremely realistic results that are hard to disambiguate from real speech also from human listeners.

The more general problem of synthetic speech generation detection has been faced through the years within the audio anti-spoofing research community. In this context, multiple algorithms based on either hand-crafted or data-driven features analysis have been proposed [13, 14]. However, since CNN-based methods for synthetic audio generation have been proposed in the last few years, many of the older detectors are bound to fail.

In this paper, we propose a method for synthetic speech audio detection. Given a speech audio track, the goal consists in detecting whether the speech is synthetic (i.e., it has been generated through some algorithms) or bona fide (i.e., it belongs to a real human speaker). In particular, we consider both closed-set and open-set scenarios. In the closed-set scenario, the proposed method detects whether the speech is bona fide or synthetic. In the case of synthetic speech, it also detect which algorithm has been used to generate the speech. In the open-set scenario, the proposed method is also able to highlight whether a fake speech has been generated through an algorithm that has never been seen before.

In order to capture traces from different kinds of synthetically generated speech tracks, we combine a series of features inspired by the speech processing literature. In particular, we propose a set of features based on the idea of modeling speech as an auto-regressive process. Differently, from other state-of-the-art methods [15], we consider multiple different auto-regressive orders at once to define this feature set. Moreover, we explore the effect of combining the proposed features with the bicoherence-based features proposed in [9] to understand whether they complement each other.

In order to validate the proposed method on multiple kinds of synthetically generated speech signals, we performed an extensive set of analyses on the publicly available ASVspoof 2019 dataset [16, 17]. This dataset contains synthetic speech tracks generated through 17 different speech synthesis techniques, ranging from the older (e.g., waveform concatenation, vocoders, etc) to novel ones based on CNNs approaches. The latter are particularly challenging to detect even by human listeners as they produce realistic speech excerpts. The results show that the proposed method proves more accurate than the approach recently proposed in [9]. In some cases, the combination of all the features is also beneficial.

The rest of the paper is organized as follows. First, we introduce some background on synthetic speech generation techniques, also reviewing some state of the art in terms of fake audio detection. We then proceed to illustrate each step of the proposed method, from the feature extraction process to the classification stage. After that, we describe the breakdown of our experimental campaign and report the achieved results. Finally, we conclude the paper highlighting the open questions for future research.

2 Background

In this section, we provide the reader with some background on state-of-the-art algorithms for synthetic speech generation and synthetic speech detection. These pieces of information are useful to better understand the challenges that lie behind the synthetic speech detection problem.

2.1 Fake speech generation

Synthetic speech generation is a problem that has been studied for many years and addressed with several approaches. For this reason, in the literature a large number of techniques that achieve good results are present and there is not a single unique way of generating a synthetic speech track.

In the past, text-to-speech (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 [18–20]. Additional post-processing steps allow to increase smoothness in transition between diphones, simulate human prosody, and retain a good degree of naturalness [21]. The main drawback of concatenative synthesis is the difficulty of modifying the voice timbral characteristics, e.g., to change speaker or embed emotional content in the voice.

To increase the variety of voice qualities or speaking styles, some methods, called HMM-based speech synthesis system (HTS) have been proposed. These operate with contextual hidden Markov models (HMMs) trained on large datasets of acoustic features extracted from diphones and triphones [22–24].

Another family of approaches, known as parametric TTS synthesis algorithms, aims at expanding the variety of generated voices. These methods take inspiration from the concept of vocoder, firstly proposed in 1939 [25]. In this case, starting from a set of speech parameters (e.g., fundamental frequency, spectral envelope and excitation signal), a speech signal is generated, typically as an auto-regressive process. However, parametric TTS synthesis produce results that sound less natural than concatenative one. Nonetheless, in the last years, more sophisticated and high-quality vocoders have been proposed [11, 26, 27]. The simplicity of the approach allows to obtain good results at a reduced computational cost, suitable for real-time scenarios.

The advent of neural networks (NNs) has broken new ground for the generation of realistic and flexible synthesized voices. In particular, modeling audio sample by sample has always been considered really challenging, since speech signal usually counts hundreds of samples for second and retains important structures at different time scales. But in the last few years, CNN and recurrent neural networks (RNN) have enabled to build completely auto-regressive models, hence to synthesize directly raw audio waveforms [12, 28, 29]. 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.

In the proposed method, we exploit a property common to these methods, i.e., they all operate in the time domain and hence inevitably create signals with memory. This feature, in our opinion, is crucial in the discrimination between fake (also called spoof or synthetic) and real (also called bona fide) speech signals.

2.2 Fake speech detection

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 explains 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. 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, it was proved that methods which choose effective and spoof-aware features outperform more complex classifiers. Moreover, long-term features should be preferred with respect to short-time features [30]. Examples are the constant-Q cepstral coefficients (CQCC) [31], based on a perceptually inspired time-frequency analysis, magnitude-based features like log magnitude spectrum or phase-based features like group delay [32]. 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 synthetic speech detection, presenting features like linear-frequency cepstral coefficients (LFCC) or mel-frequency cepstral coefficients (MFCC) [13]. In [15], 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.

More recent methods explore deep learning approaches, inspired by the success of these strategies in speech synthesis as well as other classification tasks. NNs have been proposed both for feature learning and classification steps. For example, in [8], a time frequency representation of the speech signal is presented at the input of a shallow CNN architecture. A similar framework is tested in [14]. In this case, the CNN is used solely for the feature learning step, whereas a RNN able to capture long-term 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). Also, end-to-end strategies have been proposed for spoofing detection [33]. These avoid any pre- or post-processing of the data and fuse the classification and feature learning step in a unique sleek process.

One of the most recently proposed method to detect audio deepfakes is [9], which we consider as our baseline. Given the signal $s(n)$ under analysis, the authors split it into W windows $s_w(n)$. By defining the Fourier transform of $s_w(n)$ as $S_w(\omega)$ and the complex conjugate operator as *, they compute the bicoherence as

$$B(\omega_1, \omega_2) = \frac{\sum_{w=0}^{W-1} S_w(\omega_1) S_w(\omega_2) S_w^*(\omega_1 + \omega_2)}{\sqrt{\sum_{w=0}^{W-1} |S_w(\omega_1) S_w(\omega_2)|^2 \sum_{w=0}^{W-1} |S_w^*(\omega_1 + \omega_2)|^2}}. \quad (1)$$

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 bona fide.

3 Synthetic speech detection method

In this paper, we face the problem of synthetic speech detection. This means to detect whether a speech audio track actually represents a real speech or a synthetic one. We face this problem at three different granularity levels: binary classification, closed-set classification, and open-set classification. To do so, 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 modeled as processes with memory. It is therefore possible to extract salient information by studying the relationship between past and current audio samples. Notice that, differently from other state-of-the-art methods exploiting linear prediction analysis with a single prediction order [15], we propose to use multiple orders at once.

In the binary scenario, the proposed method simply tells whether the audio recording under analysis is a real speech or a synthetically generated one. In the closed-set scenario, the proposed method is also able to recognize which synthetic speech generation algorithm has been used within a set of known algorithms. In the open-set scenario, the proposed method is able to detect whether the analyzed speech has been produced with a known or an unknown algorithm.

For each investigated scenario, the proposed pipeline is shown in Fig. 1: we extract some descriptors from the audio track under analysis; we feed the descriptors to a classifier trained to solve the binary, closed-set, or open-set problem. In the following, we illustrate the data model behind the proposed features; we provide all the details about features computation and describe the used classification methods.

3.1 Data 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 [34, 35]. Formally, the speech signal can be modeled as

$$s(n) = \sum_{i=1}^{L} a_i s(n-i) + e(n), \tag{2}$$

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

$$\hat{s}(n) = \sum_{i=1}^{L} a_i s(n-i), \tag{3}$$

where the filter coefficients a_i, $i = 1, \ldots, L$ are also called short-term prediction coefficients. By combining (2) and (3), it is possible to notice that the short-term prediction residual $s(n) - \hat{s}(n)$ is exactly $e(n)$ if the model and predictor filter coefficients a_i are coincident.

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

$$e(n) = \beta_k e(n-k) + q(n), \tag{4}$$

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

$$\hat{e}(n) = \beta_k e(n-k). \tag{5}$$

By combining (4) and (5), it is possible to notice that the long-term prediction residual $e(n) - \hat{e}(n)$ is exactly $q(n)$ 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, $i = 1, \ldots, L$ and the residual $e(n)$, which on its turn can be parameterized by β_k and the noisy residual $q(n)$. 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 signal through operations in the temporal domain (e.g., temporal convolutions, recursion, etc.). It is therefore reasonable to expect that

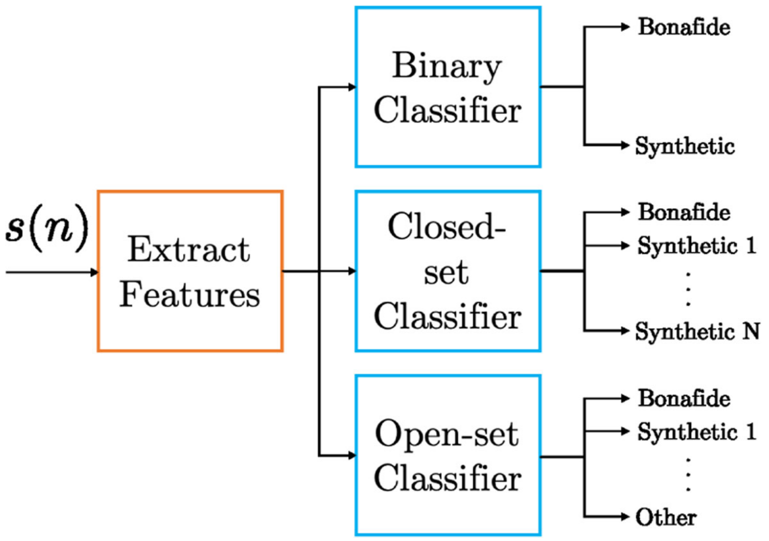

Fig. 1 Pipeline of the proposed method. A feature vector is extracted and fed to different classifiers depending on the problem to be solved

features within this model parameters domain capture salient information about the speech under analysis [15].

3.2 Features

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 $s(n)$ of length N, the feature extraction is divided in two steps, as shown in Fig. 2.

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

$$J_{\mathrm{ST}}(a_i) = \mathbf{E}\left[e^2(n)\right] = \mathbf{E}\left[\left(s(n) - \sum_{i=1}^{L} a_i s(n-i)\right)^2\right], \tag{6}$$

where $\mathbf{E}$ is the expected value operator. By imposing $\partial J_{\mathrm{ST}}/\partial a_i = 0$ for $i = 1, 2, \dots, L$, we obtain a set of well-known equations at the base of linear predictive coding [35], i.e.,

$$r(m) - \sum_{i=1}^{L} a_i r(m-i) = 0,\ m = 1, 2, \dots, L, \tag{7}$$

where $r(m)$ is the autocorrelation of the signal $s(n)$. By expressing (7) in matrix form, we obtain

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_L \end{bmatrix} = \begin{bmatrix} r(0) & r(-1) & \dots & r(1-L) \\ r(1) & r(0) & \dots & r(2-L) \\ \vdots & \vdots & \dots & \vdots \\ r(L) & r(L-1) & \dots & r(0) \end{bmatrix}^{-1} \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(L) \end{bmatrix} \tag{8}$$

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 [36]. Once the set of prediction coefficients are estimated, the short-term prediction error $e(n)$ is obtained as

$$e(n) = s(n) - \sum_{i=1}^{L} a_i s(n-i). \tag{9}$$

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 highest possible pitch of the human voice. The parameter k is obtained minimizing the energy of the long-term prediction error $q(n)$. This is done by minimizing the cost function

$$J_{LT}(k) = \mathbf{E}\left[q^2(n)\right] = \mathbf{E}\left[(e(n) - \beta_k e(n-k))^2\right], \tag{10}$$

where β_k is approximated as $\beta_k = r(k)/r(0)$ [35]. As for the short-time step, the long-term prediction error $q(n)$ can be obtained as

$$q(n) = e(n) - \beta_k e(n-k). \tag{11}$$

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

The features employed in the proposed method are directly derived from $e(n)$ and $q(n)$. 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

$$E_{\mathrm{ST}} = \frac{1}{N}\sum_{i=0}^{N-1} e(i)^2, \qquad E_{\mathrm{LT}} = \frac{1}{N}\sum_{i=0}^{N-1} q(i)^2,$$
$$G_{\mathrm{ST}} = \frac{\frac{1}{N}\sum_{i=0}^{N-1} s(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2}, \qquad G_{\mathrm{LT}} = \frac{\frac{1}{N}\sum_{i=0}^{N-1} e(i)^2}{\frac{1}{N}\sum_{i=0}^{N-1} q(i)^2}. \tag{12}$$

Rather the 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 (12) for each window w define the vectors

$$\begin{aligned} \mathbf{E}_{\mathrm{ST}} &= \left[E_{\mathrm{ST}}^0, E_{\mathrm{ST}}^1, \dots, E_{\mathrm{ST}}^{W-1}\right], \\ \mathbf{E}_{\mathrm{LT}} &= \left[E_{\mathrm{LT}}^0, E_{\mathrm{LT}}^1, \dots, E_{\mathrm{LT}}^{W-1}\right], \\ \mathbf{G}_{\mathrm{ST}} &= \left[G_{\mathrm{ST}}^0, G_{\mathrm{ST}}^1, \dots, G_{\mathrm{ST}}^{W-1}\right], \\ \mathbf{G}_{\mathrm{LT}} &= \left[G_{\mathrm{LT}}^0, G_{\mathrm{LT}}^1, \dots, G_{\mathrm{LT}}^{W-1}\right], \end{aligned} \tag{13}$$

where W is total number of windows. In the proposed method, we used a boxcar window of length equal to 0.025ms.

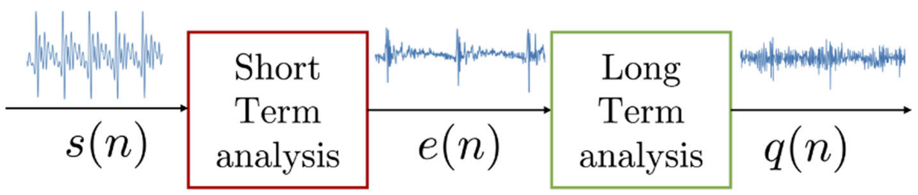

Fig. 2 STLT feature extraction. In this figure, $s(n)$ is the speech signal, $e(n)$ is the source excitation signal, and $q(n)$ is the wide-band noise signal

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

$$\begin{aligned}\mathbf{f} =[& \mu_{\mathbf{E}_{\mathrm{ST}}}, \sigma_{\mathbf{E}_{\mathrm{ST}}}, \max\left(\mathbf{E}_{\mathrm{ST}}\right), \min(\mathbf{E}_{\mathrm{ST}}),\\ & \mu_{\mathbf{E}_{\mathrm{LT}}}, \sigma_{\mathbf{E}_{\mathrm{LT}}}, \max(\mathbf{E}_{\mathrm{LT}}), \min(\mathbf{E}_{\mathrm{LT}}),\\ & \mu_{\mathbf{G}_{\mathrm{ST}}}, \sigma_{\mathbf{G}_{\mathrm{ST}}}, \max(\mathbf{G}_{\mathrm{ST}}), \min(\mathbf{G}_{\mathrm{ST}}),\\ & \mu_{\mathbf{G}_{\mathrm{LT}}}, \sigma_{\mathbf{G}_{\mathrm{LT}}}, \max(\mathbf{G}_{\mathrm{LT}}), \min(\mathbf{G}_{\mathrm{LT}})] .\end{aligned} \tag{14}$$

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 $L \in L_{\min}, \ldots, L_{\max}$. The resulting $\mathbf{f}_l$ feature vectors, where l is the considered order, are concatenated to obtain the final feature vector

$$\mathbf{f}_{\mathrm{STLT}} = \left[\mathbf{f}_{L_{\min}}, \mathbf{f}_{L_{\min+1}}, \ldots, \mathbf{f}_{L_{\max}}\right]. \tag{15}$$

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.

3.3 Classification

During the classification step, a supervised classifier is used to associate a label to the feature vector $\mathbf{f}_{\mathrm{STLT}}$. The classification training step depends on the scenario we face (i.e., binary, closed-set or open-set classification). It is worth noticing that no assumptions are made on the classification method. Indeed, any supervised classifier, like support vector machine (SVM) or random forest, can be used in all the scenarios.

3.3.1 Binary

In the binary case, the supervised algorithm is trained on a dataset where the possible labels are 0, correspondent to real bona fide speech, or 1, correspondent to synthesized speech. In this scenario, we basically train a classifier to distinguish between bona fide or synthetic speech, regardless of the used synthetic speech generation method.

3.3.2 Closed-set

In the closed-set case, a supervised algorithm is trained in a multiclass fashion, where the $N+1$ labels can have value in $[0, 1, 2, \ldots, N]$. In this case, the label 0 is assigned to bona fide speech signals, whereas the labels ranging from 1 to N are assigned to synthetic speech samples generated with N different algorithms. In this case, we basically train a classifier to recognize whether a speech track is bona fide or synthetic. In case it is synthetic, we also detect which method has been used among a set of known ones.

3.3.3 Open-set

The third configuration addresses an open-set scenario. In this case, the possible labels are $[0, 1, 2 \ldots, N, N+1]$, where the label 0 is assigned to bona fide samples, labels from 1 to N are assigned to speech samples generated with N known algorithms, while the label $N+1$ corresponds to synthetic speech signals obtained with unknown algorithms. In other word, in this case, the classifier can tell whether the speech under analysis is bona fide, is fake and generated with a known method, or belong to a class of unknown speech generation methods.

4 Experimental setup

In this section, we report all the technical details related to our experiments. We first provide the description of the used dataset. Then, we report some implementation details behind the used classifiers. Finally, we describe the used training methodology.

4.1 Dataset

In all our experiments, we used the ASVspoof 2019 dataset described in [16, 17]. 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 synthetic speech detection problem considered in our work, defined as logical access dataset in [16].

This dataset is derived from the VCTK base corpus [37] that includes bona fide 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 $\mathcal{D}_{\mathrm{tr}}$, the validation set $\mathcal{D}_{\mathrm{dev}}$, and the evaluation set $\mathcal{D}_{\mathrm{eval}}$. 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 $\mathcal{D}_{\mathrm{tr}}$ contains bona fide 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 [17]). The development set $\mathcal{D}_{\mathrm{dev}}$ contains bona fide speech from 10 (4 male, 6 female) subjects and synthetic speech generated with the same 6 methods used in $\mathcal{D}_{\mathrm{tr}}$ (i.e., from A01 to A06). The evaluation set $\mathcal{D}_{\mathrm{eval}}$ contains bona fide 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, $\mathcal{D}_{\mathrm{eval}}$ only shares 2 synthetic speech generation methods with $\mathcal{D}_{\mathrm{tr}}$ and $\mathcal{D}_{\mathrm{dev}}$, whereas 11 methods are completely new. The complete breakdown of the dataset is reported in Table 1.

Table 1 Breakdown of the used dataset showing the training, development and evaluation splits composition per number of samples, speakers, and synthesis methods

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

The column "Category" roughly indicates the approach used for waveform generation by the synthetic speech generation algorithm, where NN = neural network, VC = vocoder, and WC = waveform concatenation

The synthetic speech generation algorithms considered in this dataset 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 [17]:

A01 is a NN-based TTS system that uses a powerful neural waveform generator called WaveNet [12]. The WaveNet vocoder follows the recipe reported in [38].

A02 is a NN-based TTS system similar to A01 except that the WORLD vocoder [11] 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 [39].

A04 A waveform concatenation TTS system based on the MaryTTS platform [10].

A05 is a NN-based voice conversion (VC) system that uses a variational auto-encoder (VAE) [40] and WORLD vocoder for waveform generation.

A06 is a transfer-function-based VC system [41]. 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 [42], 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 [43], which is faster than WaveNet.

A09 is a NN-based TTS system [44] that uses Vocaine vocoder [27] to generate waveforms.

A10 is an end-to-end NN-based TTS system [45] that applies transfer learning from speaker verification to a neural TTS system called Tacotron 2 [28]. The synthesis is performed through WaveRNN neural vocoder [29].

A11 is a neural TTS system that is the same as A10 except that it uses the Griffin-Lim algorithm [46] 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 [47].

A14 is another combined VC and TTS system that uses the STRAIGHT vocoder [26] for waveform reconstruction.

A15 is another 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 [47].

A18 is a non-parallel VC system [48] that uses a vocoder to generates speech from MFCC.

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.

4.2 Classifiers

The proposed features can be used with any supervised classifier. In our experimental campaign, we focused on simple and classical classifiers in order to study the amount of information captured by the proposed features. Specifically we used a random forest, a linear SVM and a radial basis function (RBF) SVM.

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

- Random forest: 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 have been used. In particular, we used 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 [49] Python library.

5 Results

In this section, we collect and comment all the results achieved through the performed experimental campaign. We first report an analysis that justify the use of multiple prediction orders in the feature extraction procedure. Then, we report the results depending on the used classification framework: binary, closed-set, and open-set. Finally, we conclude the section with a preliminary experiment on encoded audio tracks.

5.1 Impact of prediction order

As mentioned in the Section 2, other methods proposed in the literature make use of the source-filter model to extract characteristic features [15]. 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 reports the best accuracy obtained on $\mathcal{D}_{\text{eval}}$ and $\mathcal{D}_{\text{dev}}$ 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 $\mathcal{D}_{\text{eval}}$.

5.2 Binary results

In this experiment, we consider the binary classification problem. Given an audio recording, our goal is to detect whether it is pristine or synthetic, independently from the used speech generation algorithm.

For this test, we used $\mathcal{D}_{\text{tr}}$ as training set. As features, we compared the baseline bicoherence-based ones [9] (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, we have three different Bicoherence results, and three different STLT + Bicoherence results.

Table 2 shows the results achieved considering the best classifier and preprocessing combination for each feature

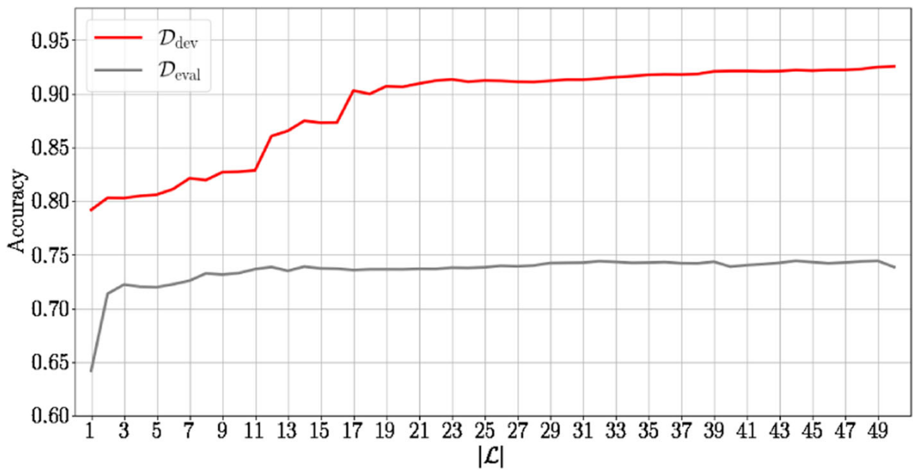

Fig. 3 Accuracy achieved in the binary scenario on $\mathcal{D}_{\text{dev}}$ and $\mathcal{D}_{\text{eval}}$ for different cardinalities of $\mathcal{L}$

Table 2 Bona fide vs. synthetic accuracy on dataset $\mathcal{D}_{\mathrm{dev}}$ for each synthetic speech algorithm

	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**

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 Bicoherences 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.

Table 3 shows the same results breakdown when the trained classifiers are tested on the $\mathcal{D}_{\mathrm{eval}}$ 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 methods, 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 Random Forest. 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. These considerations are valid also on closed-set and open-set results.

As an additional remark on the binary setup, it is worth noting that we also tested the purely data-driven method proposed in [8]. However, due to the heterogeneous nature of the used datasets, and the limited amount of available data when considering balanced classes, we

Table 3 Bona fide vs. synthetic accuracy on dataset $\mathcal{D}_{\mathrm{eval}}$ for each synthetic speech algorithm

	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

could not achieve an accuracy higher than 0.72 on $\mathcal{D}_{\text{dev}}$ and 0.71 on $\mathcal{D}_{\text{eval}}$. As a matter of fact, it is well known that proper CNN training relies on the availability of a huge amount of training data, which is not often available in forensic scenarios.

5.3 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 bona fide (i.e., BF) or synthetic, and to which synthetic class it belongs.

Figure 4 shows the confusion matrix obtained using the baseline Bicoherence, the proposed STLT, and the fusion Bicoherence + STLT methods training the classifiers on $\mathcal{D}_{\text{tr}}$ and testing on $\mathcal{D}_{\text{dev}}$. This is possible as $\mathcal{D}_{\text{tr}}$ and $\mathcal{D}_{\text{dev}}$ 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 5 show the same results achieved by training on a portion of $\mathcal{D}_{\text{eval}}$ (i.e., 80%) and testing on the remaining portion of $\mathcal{D}_{\text{eval}}$ (i.e., 20%). This was necessary as only two methods from $\mathcal{D}_{\text{eval}}$ are present in $\mathcal{D}_{\text{tr}}$. 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 provide satisfying results. The methods on which the classifiers suffer the most are A10 and A12, which exploit WaveRNN and WaveNet. Additionally, also A16 based on waveform concatenation seems to be more difficult to detect than other categories of fake speech.

The reason behind this behavior can be explained as it follows. Both WaveRNN (A10) and WaveNet (A12) are end-to-end methods. This means they are completely data-driven; thus, the produced audio tracks reasonably conform less with the assumed source-filter model. Additionally, they are among the methods that provide the most realistic listening results. For what concerns A16, the problem is different. Fake speech tracks generated through waveform concatenation are roughly portions of bona fide speech atoms spliced together with some processing. For this reason, distinguishing them completely from real bona fide may prove more challenging.

5.4 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., bona fide and some synthetic speech methods), and

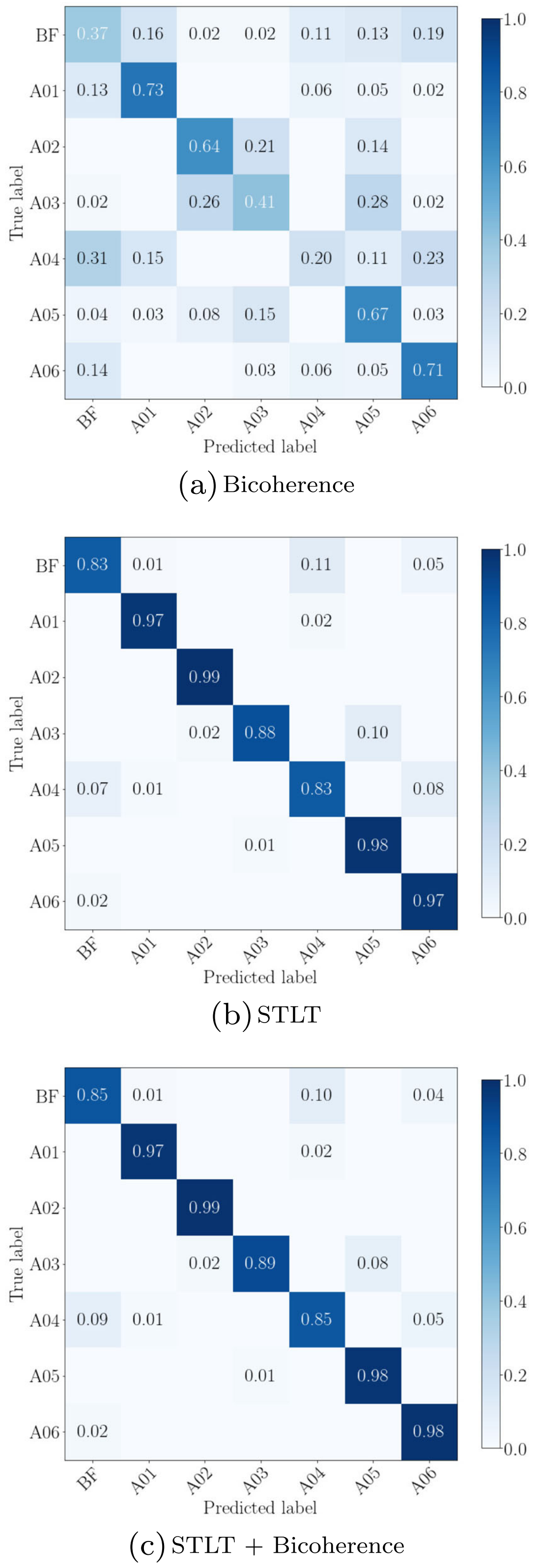

Fig. 4 Confusion matrices showing closed-set results for each used feature vector on dataset $\mathcal{D}_{\text{dev}}$

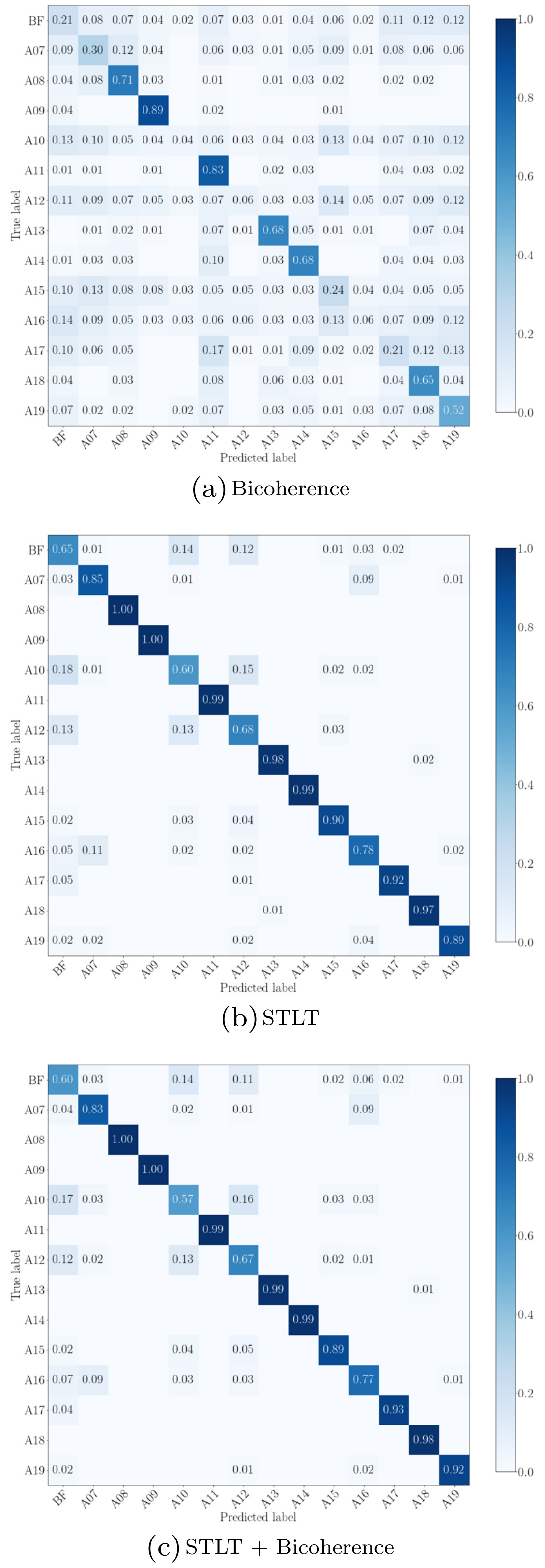

Fig. 5 Confusion matrices showing closed-set results for each used feature vector on dataset $\mathcal{D}_{\text{eval}}$

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 bona fide class), the important point is to avoid mixing bona fide with fakes.

Figure 6 shows the results achieved training on $\mathcal{D}_{\text{tr}}$ and testing on the union of $\mathcal{D}_{\text{dev}}$ and $\mathcal{D}_{\text{eval}}$. Specifically, we used as known classes the bona fide one plus 4 of the 6 synthetic classes present in $\mathcal{D}_{\text{tr}}$. We select as known-unknown the two remaining synthetic speech methods from $\mathcal{D}_{\text{tr}}$ (i.e., KN-UNKN). The classifier can classify the excerpt under analysis into 6 classes: bona fide (i.e., BF), one of the 4 known synthetic methods, or unknown (i.e., UNKN). In evaluating the results, we keep the known classes separated, as they should be recognized correctly. 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 6a 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 bona fide 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 bona fide, there is a chance that the speech has been generated through a synthetic method. Figure 6b shows the same results in the case of known-unknown equal 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 bona fide, we noticed an interesting fact. Independently from the known-unknown pair selected at training time among the ones available in $\mathcal{D}_{\text{tr}}$, the wrongly classified unknowns are A10, A11, A12, and A15. In fact, they are misclassified as bona fide in the 89% of the cases. These are methods based on WaveNet, WaveRNN, and Griffin-Lim. The first two families of methods produce very likely speech. The last family is never represented in the known-unknown set. All methods based on vocoders, waveform concatenation, and 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.

5.5 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 customary compressed using lossy

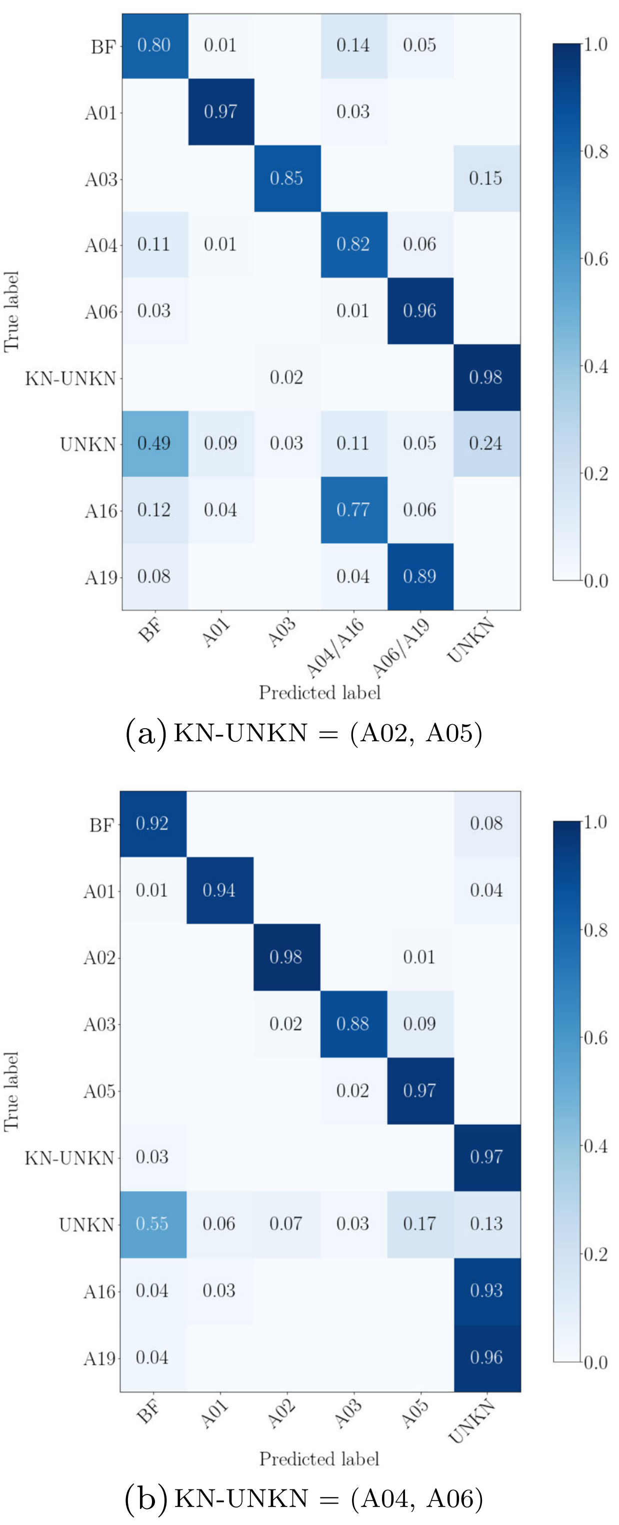

Fig. 6 Confusion matrices showing Bicoherence + STLT open-set results on the union of $\mathcal{D}_{\text{dev}}$ and $\mathcal{D}_{\text{eval}}$

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 $\mathcal{D}_{\text{dev}}$ 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 operation 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.

6 Conclusions

In this paper, we proposed a method to detect AI-generated synthetic speech audio tracks. 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 are motivated by the broad use of source-filter model for the analysis and synthesis of speech signals. As a matter of fact, we propose to extract different statistics obtained by filtering the signal under analysis with short-term and long-term predictors, considering different prediction orders.

The proposed features have been compared with the recently proposed baseline method [9] exploiting bicoherence analysis on the ASVspoof 2019 dataset [17]. The results show that the proposed method outperforms the bicoherence-based one in the binary, closed-set, and open-set scenarios. Moreover, the joint use of the proposed features and the bicoherence-ones provides an accuracy gain in some situations.

Despite the achieved promising results, several scenarios need further investigation. For instance, it is still challenging to accurately detect some families of synthetic speech tracks in the open-set scenario due to the huge variety of synthetic speech generation methods. Moreover, we only considered the logical access synthetic speech detection problem, i.e., we analyze a clean recording of each speech. It is therefore part of our future studies to consider what happens if speech tracks get corrupted by noise, coding, or transmission errors. This scenario is particularly important if we consider that synthetic speech recordings may be shared through social platforms or used live during phone calls.

Acknowledgements
This work was supported by the PREMIER project, funded by the Italian Ministry of Education, University, and Research within the PRIN 2017 program. Hardware support was generously provided by the NVIDIA Corporation.

Author's contributions
All authors contributed in the design of the proposed method and writing the manuscript. All authors read and approved the final manuscript.

Funding
This work was supported by the PREMIER project, funded by the Italian Ministry of Education, University, and Research within the PRIN 2017 program. Hardware support was generously provided by the NVIDIA Corporation.

Availability of data and materials
The used dataset is freely available as part of the ASVSpoof 2019 challenge at https://datashare.is.ed.ac.uk/handle/10283/3336.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 22 September 2020 Accepted: 10 March 2021
Published online: 06 April 2021

References

1. B. Dolhansky, J. Bitton, B. Pflaum, R. Lu, R. Howes, M. Wang, C. C. Ferrer, The deepfake detection challenge dataset. CoRR 2006.07397 (2020)
2. L. Verdoliva, Media forensics and deepfakes: an overview. CoRR 2001.06564 (2020)
3. Deepfakes github. https://github.com/deepfakes/faceswap
4. Y. Li, M. Chang, S. Lyu, in *IEEE International Workshop on Information Forensics and Security (WIFS)*. In ictu oculi: exposing AI created fake videos by detecting eye blinking (IEEE, Hong Kong, 2018)
5. D. Güera, E. J. Delp, in *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*. Deepfake video detection using recurrent neural networks (IEEE, Auckland, 2018)
6. F. Matern, C. Riess, M. Stamminger, in *IEEE Winter Applications of Computer Vision Workshops (WACVW)*. Exploiting visual artifacts to expose deepfakes and face manipulations (IEEE, Waikoloa, 2019)
7. N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, S. Tubaro, in *International Conference on Pattern Recognition (ICPR)*. Video face manipulation detection through ensemble of CNNs (Springer, Milan, 2020)
8. A. Lieto, D. Moro, F. Devoti, C. Parera, V. Lipari, P. Bestagini, S. Tubaro, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Hello? Who am i talking to? A shallow CNN approach for human vs. bot speech classification (IEEE, Brighton, 2019), pp. 2577–2581
9. E. A. AlBadawy, S. Lyu, H. Farid, in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Detecting AI-synthesized speech using bispectral analysis (Computer Vision Foundation/IEEE, Long Beach, 2019), pp. 104–109
10. M. Schröder, M. Charfuelan, S. Pammi, I. Steiner, in *Conference of the International Speech Communication Association (INTERSPEECH)*. Open source voice creation toolkit for the MARY TTS platform (ISCA, Florence, 2011)
11. M. Morise, F. Yokomori, K. Ozawa, WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. IEICE Trans. Inf. Syst. **99**, 1877–1884 (2016)
12. A. V. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: a generative model for raw audio. CoRR 1609.03499 (2016)
13. M. Sahidullah, T. Kinnunen, C. Hanilçi, in *Conference of the International Speech Communication Association (INTERSPEECH)*. A comparison of features for synthetic speech detection (ISCA, Dresden, 2015)
14. C. Zhang, C. Yu, J. H. Hansen, An investigation of deep-learning frameworks for speaker verification antispoofing. IEEE J. Sel. Top. Sig. Process. **11**, 684–694 (2017)
15. A. Janicki, in *Sixteenth Annual Conference of the International Speech Communication Association*. Spoofing countermeasure based on analysis of linear prediction error (ISCA, Dresden, 2015)
16. M. Todisco, X. Wang, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. Kinnunen, K. A. Lee, in *Conference of the International Speech Communication Association (INTERSPEECH)*. ASVspoof 2019: future horizons in spoofed and fake audio detection (ISCA, Graz, 2019)
17. 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, Z.-H. Ling, ASVspoof 2019: a large-scale public database of synthesized, converted and replayed speech. Comput. Speech Lang. **64**, 101–114 (2020)
18. E. Moulines, F. Charpentier, Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. Speech Comm. **9**, 453–467 (1990)
19. A. J. Hunt, A. W. Black, in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*. Unit selection in a concatenative speech synthesis system using a large speech database (IEEE, Atlanta, 1996)
20. A. Black, N. Campbell, in *EUROSPEECH*. Optimising selection of units from speech databases for concatenative synthesis (ISCA, Madrid, 1995)
21. S. P. Panda, A. K. Nayak, A waveform concatenation technique for text-to-speech synthesis. Int. J. Speech Technol. **20**, 959–976 (2017)
22. T. Masuko, K. Tokuda, T. Kobayashi, S. Imai, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Speech synthesis using HMMs with dynamic features (IEEE, Atlanta, 1996)
23. K. Tokuda, H. Zen, A. W. Black, in *IEEE Speech Synthesis Workshop*. An HMM-based speech synthesis system applied to English (IEEE, Santa Monica, 2002)
24. M. K. Reddy, K. S. Rao, Robust pitch extraction method for the HMM-based speech synthesis system. IEEE Sig. Process. Lett. **24**, 1133–1137 (2017)
25. H. Dudley, Remaking speech. J. Acoust. Soc. Am. **11**, 169–177 (1939)
26. H. Kawahara, I. Masuda-Katsuse, 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 Comm. **27**, 187–207 (1999)
27. Y. Agiomyrgiannakis, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vocaine the vocoder and applications in speech synthesis (IEEE, Brisbane, 2015)
28. J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, Y. Wu, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions (IEEE, Calgary, 2018)
29. N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, K. Kavukcuoglu, Efficient neural audio synthesis. CoRR 1802.08435 (2018)
30. M. R. Kamble, H. B. Sailor, H. A. Patil, H. Li, Advances in anti-spoofing: from the perspective of ASVspoof challenges. APSIPA Trans. Sig. Inf. Process. **9**, 18 (2020). https://www.cambridge.org/core/journals/apsipa-transactions-on-signal-and-information-processing/article/advances-in-antispoofing-from-the-perspective-of-asvspoof-challenges/6B5BB5B75A49022EB869C7117D5E4A9C
31. M. Todisco, H. Delgado, N. Evans, Constant Q cepstral coefficients: a spoofing countermeasure for automatic speaker verification. Comput. Speech Lang. **45**, 516–535 (2017)
32. X. Xiao, X. Tian, S. Du, H. Xu, E. S. Chng, H. Li, in *Sixteenth Annual Conference of the International Speech Communication Association*. Spoofing speech detection using high dimensional magnitude and phase features: the NTU approach for ASVspoof 2015 challenge (ISCA, Dresden, 2015)
33. H. Dinkel, N. Chen, Y. Qian, K. Yu, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. End-to-end spoofing detection with raw waveform CLDNNS (IEEE, New Orleans, 2017)
34. G. Fant, The source filter concept in voice production. Speech Transm. Lab. Q. Prog. Status Rep. **1**, 21–37 (1981)
35. Linear prediction in narrowband and wideband coding (John Wiley & Sons, Ltd, Hoboken, 2005), pp. 91–112. Chap. 4

36. J. Franke, A Levinson-Durbin recursion for autoregressive-moving average processes. Biometrika. **72**, 573–581 (1985)
37. VCTK corpus. http://dx.doi.org/10.7488/ds/1994. Accessed 23 Mar 2021
38. X. Wang, J. Lorenzo-Trueba, S. Takaki, L. Juvela, J. Yamagishi, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. A comparison of recent waveform generation and acoustic modeling methods for neural-network-based speech synthesis (IEEE, Calgary, 2018)
39. Z. Wu, O. Watts, S. King, in *Speech Synthesis Workshop (SSW)*. Merlin: an open source neural network speech synthesis system (Sunnyvale, ISCA, 2016)
40. C. Hsu, H. Hwang, Y. Wu, Y. Tsao, H. Wang, in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. Voice conversion from non-parallel corpora using variational auto-encoder (IEEE, Jeju, 2016)
41. D. Matrouf, J. Bonastre, C. Fredouille, in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*. Effect of speech transformation on impostor acceptance (IEEE, Toulouse, 2006)
42. K. Tanaka, H. Kameoka, T. Kaneko, N. Hojo, WaveCycleGAN2: time-domain neural post-filter for speech waveform generation. CoRR 1904.02892 (2019)
43. X. Wang, S. Takaki, J. Yamagishi, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Neural source-filter-based waveform model for statistical parametric speech synthesis (IEEE, Brighton, 2019)
44. H. Zen, Y. Agiomyrgiannakis, N. Egberts, F. Henderson, P. Szczepaniak, in *Conference of the International Speech Communication Association (INTERSPEECH)*. Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices (ISCA, San Francisco, 2016)
45. Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, z. Chen, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu, in *Advances in Neural Information Processing Systems (NIPS)*. Transfer learning from speaker verification to multispeaker text-to-speech synthesis (Curran Associates, Inc., Montreal, 2018)
46. D. Griffin, J. Lim, Signal estimation from modified short-time Fourier transform. IEEE Trans. Acoust. Speech Sig. Process. (TASLP). **32**, 236–243 (1984)
47. K. Kobayashi, T. Toda, S. Nakamura, Intra-gender statistical singing voice conversion with direct waveform modification using log-spectral differential. Speech Commun. **99**, 211–220 (2018)
48. T. Kinnunen, J. Lorenzo-Trueba, J. Yamagishi, T. Toda, D. Saito, F. Villavicencio, Z. Ling, in *The Speaker and Language Recognition Workshop*. A spoofing benchmark for the 2018 voice conversion challenge: leveraging from spoofing countermeasures for speech artifact assessment (ISCA, Les Sables d'Olonne, 2018)
49. 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, E. Duchesnay, Scikit-learn: machine learning in Python. J. Mach. Learn. Res. (JMLR). **12**, 2825–2830 (2011)

Publisher's Note