CONAN: Spoof Detection with COmputed Noiselessness And Non-linear Fusion

Jacob Li
Electrical & Computer Eng.
Carnegie Mellon University
Pittsburgh, PA 15213
jacobli@andrew.cmu.edu

Yuxuan Wu
Music & Technology
Carnegie Mellon University
Pittsburgh, PA 15213
yuxuanw2@andrew.cmu.edu

Zhongyuan Zhai
Electrical & Computer Eng.
Carnegie Mellon University
Pittsburgh, PA 15213
zhongyuz@andrew.cmu.edu

Yuxiang Zi
Electrical & Computer Eng.
Carnegie Mellon University
Pittsburgh, PA 15213
yzi@andrew.cmu.edu

Abstract

Along with the rapid advancement of the audio synthesis technology, the necessity to confidently discriminate between human and synthesized speeches becomes more and more salient. Intending to further the development of countermeasures against various sorts of speech spoofing attacks without presupposing any ad hoc spoofing strategies, CONAN, an automatic speaker verification and spoofing countermeasure, is proposed. Gaussian Mixture Model (GMM) is implemented as the base classifier to fit on several different audio feature representations along with a 'silence' feature. Then non-linear fusion is conducted on a diverse set of weak models as well as silence measurement to yield a more robust and reliable performance. With this nonlinear fusion, an EER of 4.84% and a t-DCF of 0.1321 is achieved, ranking No.9 on the leaderboard of Automatic Speaker Verification Spoofing and Countermeasures Challenge (ASVspoof) 2019 challenge.

1 Introduction

Forgery of human speeches spread across the Internet has manifested to incur atrocious social and economic consequences. As audio synthesis technology advances rapidly, the need of reliable countermeasures grows fast. Intended to promote development of countermeasures against various sorts of speech spoofing attacks without presupposing any ad hoc spoofing strategies, the Automatic Speaker Verification (ASV) and Spoofing Countermeasures Challenge (ASVspoof) serves as an superior biannual event to proffer training, development and evaluation data set as well as to devise realization baselines. Two tasks, Logical Access (LA) and Physical Access (PA) constitute the ASVspoof 2019. Our programming implementation resides in here

2 Related Work

Typical spoofing attacks involve voice conversion, replay, and speech synthesis [1]. Different ASV schemes tend to differ in vulnerability when subjected to different spoofing attacks.

Drawing upon the raised noise and the flattened spectrum, an SVM-based system, which takes the spectral ratio and modulation indices as input features, is proven effective to counter replay by reducing the false acceptance rate from 68% to 0% [2]. An alternative method also capitalizes the channel noise introduced by the replaying devices [3].

Utilizing Constant Q Cepstral Coefficient (CQCC) [4] with Gaussian Mixture Model (GMM) provides promising results on ASV tasks [5]. Other feature representations, such as inverse Mel frequency cepstral coefficients (IMFCC) [6], linear frequency cepstral coefficients (LFCC) [7], and linear filter bank energy (LFBE) [8], have shown good results as well.

The bona fide data set in AVSspoof [9] tend to incorporate more extensive leading and trailing silences as opposed to their counterpart spoofed instances [10]. With the leading and trailing silences trimmed away, the equal error rate(EER) dramatically deteriorates from 3.6% up to 15.5%.

In order to design a robust spoofing countermeasure system, some Known and Unknown data augmentation (DA) methods are introduced [11].

A recent approach incorporating simple sub-band classifiers through non-linear fusion has demonstrated to deliver near state-of-the-art performance [12], where each sub-band classifier is tuned to detect certain type of attack.

3 Dataset

The training, development and evaluation dataset in ASVspoof 2019 are adopted. The LA task is deliberated as the benchmark for its algorithmic transferability.

The prime mover is a series of bona fide recordings and their spoofing counterparts generated by text-to-speech (TTS), voice conversion (VC) or hybrid algorithms, then transmitted through either a PSTN or a VoIP network equipped with some particular codec.

The ASVspoof 2019 dataset is based upon a standard multi-speaker speech synthesis database entitled VCTK. Note that the development data here are not conventional. ASVspoof 2019 challenge claimed that the spoofed speech in development dataset is generated according to one of the same spoofing algorithms [13][14] used to generate the training dataset [9]. For the evaluation dataset, the spoofed data are generated according to diverse unseen spoofing algorithms which are variants of the spoofing algorithms used to generate the development dataset [9]. In conclusion, in ASVspoof 2019, we are allowed to use development data to design and optimize the spoofing countermeasures so that it can accurately detect new spoofed data generated with different or unseen spoofing algorithms.

However, due to the training constraint which will be discussed in detail in Section 6 and time constraint, we decide not to use the development data. In this project, all the experiment were trained on the training dataset and evaluated on the evaluation dataset.

Table 1: ASVspoof 2019 dataset for LA task

Dataset Type	Bona fide Sample	Spoofed Sample	Total Sample
Training	2580	22800	25380
Development	2548	22296	24844
Evaluation	8051	63882	71933

4 Method

4.1 Baseline

The baseline system provided by ASVspoof 2019 Challenge consists of a CQCC or LFCC front-end and a Gaussian Mixture Model (GMM) back-end. The front-end transforms the input audio into a particular feature representation, through a 20ms window length, a 10ms shift and a filterbank of 20 filters. Meanwhile, the differential and acceleration characteristics of the cepstral coefficients are incorporated into the feature represenation to gain insights into the dynamics of the power spectra and trajectories over time, resulting in a 60-dimensional feature for every frame (20 coefficients, 20 Δ coefficients, and 20 $\Delta\Delta$ coefficients). The GMM back-end classifiers are comprised of two 512-component mixture models. One classifier is trained on bona fide data, hence is modeled for

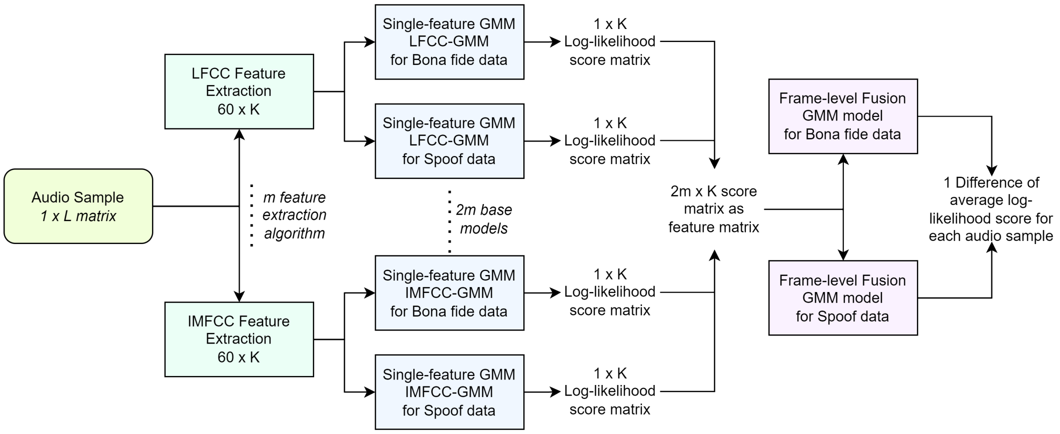
Audio Sample
1 x L matrix
LFCC Feature Extraction 60 x K
IMFCC Feature Extraction 60 x K
m feature extraction algorithm
Single-feature GMM LFCC-GMM for Bona fide data
Single-feature GMM LFCC-GMM for Spoof data
2m base models
Single-feature GMM IMFCC-GMM for Bona fide data
Single-feature GMM IMFCC-GMM for Spoof data
1 x K Log-likelihood score matrix
1 x K Log-likelihood score matrix
1 x K Log-likelihood score matrix
1 x K Log-likelihood score matrix
2m x K score matrix as feature matrix
Frame-level Fusion GMM model for Bona fide data
Frame-level Fusion GMM model for Spoof data
1 Difference of average log-likelihood score for each audio sample

Figure 1: Illustation of frame-level fusion

bona fide type audio; the other is trained on spoofed samples, hence is modeled for spoofed type audio. The Python version of the ASVspoof 2019's baseline model is adopted.

4.2 Improvements

4.2.1 Window Length Optimization

Raw audio files in time domain are transformed into cepstral coefficients in frequency domain through Short-time Fourier Transform. Audio is at first segmented into frames and window functions are used to prevent spectral leakage. The window length is a primary hyper parameter during this process. In general, a window length too long will cause more non-stationarity, and a window length too short will bring more information loss in spectrum.

Since spoofing artifacts are best detected at high spectral resolution within sub-band and too high spectral resolution can generate noisy feature [15], single feature GMM models with LFCC or MFCC front-ends using window length of 5ms, 25ms, and 30ms were tested. And the 25ms window length processed models outperforms other window length configuration on all three feature types. At the same time, baseline model with window length configured to 25ms observed a significant 21.27% drop in min t-DCF and a 17.92% drop in EER compared to the original baseline model as seen in table. It terms of number of filters for filterbank, 70 is tested to be optimal [12]. Thus, all the subsequent experiments are configured with 25ms window length, 10ms shift length, and a filterbank with 70 filters.

4.2.2 Silence as Feature

Müller, et al.[10] demonstrated that silent components were prone to be overlooked in speech audio, and when silence trimmed, the performance can but they didn't thus a way to exploit the temporal characteristics of the leading and trailing silences is to model these silences into features.

We adopted a method similar to voice activity detection (VAD) to extract silent fragments from speech audio. Hand-tuned short-time energy and zero-crossing rate thresholds were used to determine the boundaries of the active speech. The lengths of leading and trailing silences will act as two features in addition to the original features.

4.2.3 Non-linear Fusion of Models

Assimilating various features into the model can potentially attain higher robustness and better performance. There are different methods to combine features into a single system: one is to directly concatenate single features in more complex features; second is to combine the output of models with single features. Since the baseline model back-ends are relative simple GMM systems, pushing more diverse features into single model could easily create over-fitting or under-fitting situations or greatly increase complexity of single model in order to properly model the data, which will dramatically

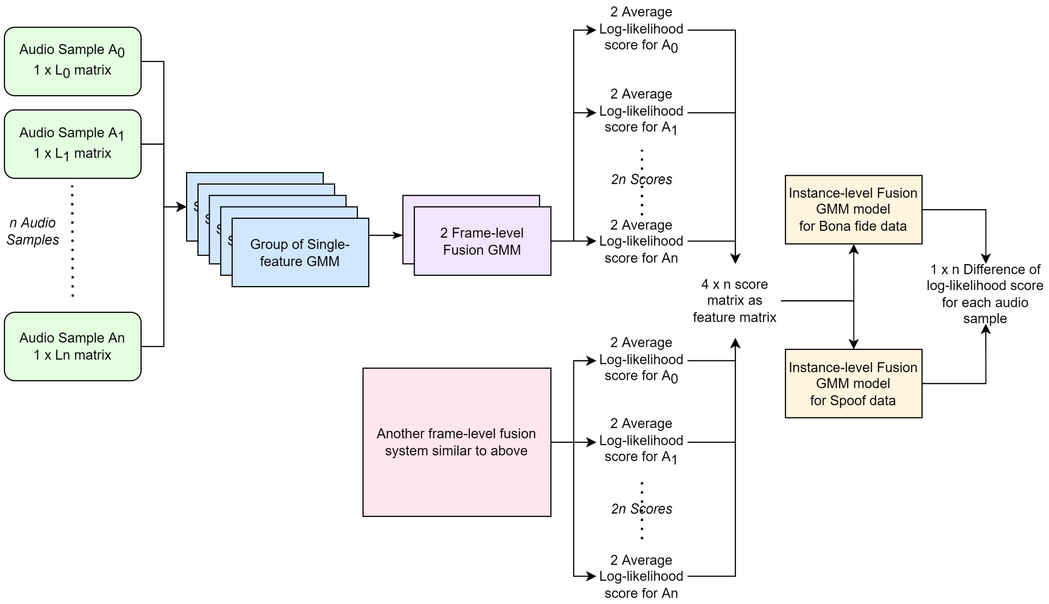
Audio Sample A_0
1 x L_0 matrix
Audio Sample A_1
1 x L_1 matrix
n Audio Samples
Audio Sample An
1 x Ln matrix
Group of Single-feature GMM
2 Frame-level Fusion GMM
2 Average Log-likelihood score for A_0
2 Average Log-likelihood score for A_1
2n Scores
2 Average Log-likelihood score for An
4 x n score matrix as feature matrix
Instance-level Fusion GMM model for Bona fide data
Instance-level Fusion GMM model for Spoof data
1 x n Difference of log-likelihood score for each audio sample
Another frame-level fusion system similar to above
2 Average Log-likelihood score for A_0
2 Average Log-likelihood score for A_1
2n Scores
2 Average Log-likelihood score for An

Figure 2: Illustation of instance-level fusion

increase computational cost of training and testing. Thus, combining outputs of simple models is chosen.

We proposed a hierarchical structure to combine feature performance both at frame-level and at instance-level. At frame level, as shown in Figure. 1, the base models trained with a diverse set of features will produce a log-likelihood score for every frame. Stacking N output scores vertically will create a new feature of N dimensions. This feature is used to train the frame-level fusion GMM models, for bona fide and spoof data respectively. At instance-level, for which every audio sample is counted as a data instance, the base model will produce a aggregate score by averaging the log-likelihood across all frames, as shown in Figure. 2. For M base models, each instance will obtain M aggregated log-likelihood score and those score can be stacked to create a new M-dimension feature for every audio sample. The combinations of model output is defined as fusion in our experiment.

Approaches to fusion include linear fusion, such as AdaBoost and Logistic Regression, and non-linear fusion, such as Support-Vector Machine (SVM) and GMM. Since baseline system is using GMM back-ends and the combination of GMM output would also follow Gaussian distribution, it is safe to assume using another GMM to model the Gaussian distribution of different scores would be appropriate. Fusion models at different levels were optimized separately.

5 Result

5.1 Evaluation Metric

The performance is measured by the Equal Error Rate (EER) and the minimum tandem detection cost function (min t-DCF). EER marks the equality of the miss and false alarm rates as detection threshold slides. In a joint system of ASV and CM, the t-DCF metric reflects its potential consequences of decision errors in various practical settings[16]. Specifically, t-DCF measures cost (difficulty) of each data instance by its ASV performance and CM performance. In other words, data samples with worse ASV performance will weight more in evaluation.

The t-DCF function is laid forth as:

$$\text{t-DCF}(s) = C_1 P^{\text{cm}}_{\text{miss}}(s) + C_2 P^{\text{cm}}_{\text{fa}}(s) \tag{1}$$

where C_1 and C_2 are assigned constants, $P^{cm}_{miss}(s)$ represents the miss rate on bona fide data and $P^{cm}_{fa}(s)$ represents the false alarm rate on spoofed data at threshold s. The two rates are respectively formulated as:

$$P^{\text{cm}}_{\text{miss}}(s) = \frac{\#\{\text{bona fide trails with CM score} \leq s\}}{\#\{\text{Total bona fide trials}\}} \tag{2}$$

$$P^{\text{cm}}_{\text{fa}}(s) = \frac{\#\{\text{spoof trails with CM score} > s\}}{\#\{\text{Total spoof trials}\}} \tag{3}$$

5.2 Experiment Results

5.2.1 Performance on Various Window Length

The baseline model performance is compared with our single feature model in order to demonstrate the impact of different window length as shown in Table 2. As expected, the increasing window length would reduce the impact of noise in audio processing and thus improve performance.

Table 2: Result Comparison with Baseline on Evaluation Data

Feature	Model	Window Length (ms)	min t-DCF	EER (%)
MFCC	GMM	25	0.2203	9.37
MFCC	GMM	30	0.2361	10.10
LFCC	GMM	25	0.1666	6.64
LFCC	GMM	30	0.1805	7.20
∗ LFCC	GMM	20	0.2116	8.09

∗: Published baseline model performance from ASVspoof 2019

5.2.2 Performance on Different Levels of Fusion

Frame level and instance level fusions on several base models are performed and have manifested decent performances. As the GMM model components continue doubling up, both metrics keep non-trivially decreasing. Through combining more feature representations into the frame-level models, the pertinent EER scores still continue to drop even though the min t-DCF maintains relatively stable. Satisfactorily, by incorporating instance-level fusion technique, the min t-DCF and EER rate can be further lowered by a significant margin. Notice that with silence fused in, the overall performance deteriorates. A detailed interpretation of this result is in Section 6.

Table 3: Result Comparison on Different Fusion Models Evaluation Data

Feature	Model-#Component	Model Type	min t-DCF	EER (%)
MFCC	GMM-512	single	0.2203	19.37
LFCC	GMM-512	single	0.1666	6.64
IMFCC	GMM-512	single	0.1848	9.25
LFCC-IMFCC	GMM-64	frame-level	0.1616	7.60
LFCC-IMFCC	GMM-128	frame-level	0.1538	7.11
LFCC-IMFCC	GMM-256	frame-level	0.1497	6.79
LFCC-IMFCC	GMM-512	frame-level	0.1474	6.78
LFCC-MFCC-IMFCC	GMM-128	frame-level	0.1548	5.90
LFCC-MFCC-IMFCC-NGCC	GMM-256	frame-level	0.1472	5.30
LFCC-MFCC-IMFCC-NGCC	GMM-512	frame-level	0.1421	5.22
∗ 3 frame-level model	GMM-128	instance-level	0.1321	4.84
∗ 3 frame-level model w/ silence	GMM-128	instance-level	0.1433	5.48

∗: 3 frame-level models are: LFCC-MFCC-IMFCC(128), LFCC-MFCC-IMFCC-NGCC(256),
LFCC-MFCC-IMFCC-NGCC(512)

5.2.3 Performance Benchmark with Prior Work

To be more concise, we've compared the performance of the official baseline model, several outstanding teams participated the ASVspoof2019 challenge with our instance-level non-linear fushion model. As shown in Table 4, we were able to achieve a high rank among ASVspoof 2019 contestants. Our performance went near T32, which was the best performing non-neural network system on the ranking list.

Table 4: Benchmark with ASVspoof 2019 Competition Teams on Evaluation Data

Team ID/Model Name	Ranking	Model Type	min t-DCF	EER (%)
⋆ LFCC-GMM	28	Non-NN	0.2116	8.09
⋆ CQCC-GMM	33	Non-NN	0.2366	9.57
T05	1	NN	0.0069	0.22
T32	8	Non-NN	0.1239	4.92
T58	9	Non-NN	0.1333	6.14
T22	12	NN	0.1545	6.20
∗ 3 frame-level model	-	Non-NN	0.1321	4.84
∗ 3 frame-level model w/ silence	-	Non-NN	0.1433	5.48

∗: our implementation, Non-NN: Non-Neural Network models, NN: Neural Network models, ⋆: official baseline models from ASVspoof 2019

6 Discussion and Analysis

6.1 Training Method Experiment

Due to the large amount of training data and a configuration of 512 components in GMM models, the training process demands more memory space than most computers have in their RAM. In fact, We tested with a 64Gb laptop but the training process still failed. Therefore, we alternatively divide training data into multiple chunks and train our GMM models one chunk at a time until we fit in all the data. We employed the warm start mechanism to keep the consistency among chunks, which means previously trained parameters are used to initialize the following iteration.

Experiments are performed to validate this process. First, we compared training chunk by chunk to training on a smaller dataset excerpted from the original. Results showed that iterative training greatly surpassed training on a small dataset, which indicates that by training chunk by chunk the model can actually learn from all chunks seen. Second, we compared training with different number of chunks (ranging from 2 to 20, as long as the computer is capable of that configuration). It turned out that there is no significant difference, which showed the division from whole data into chunks doesn't affect the training result. However, due to a limitation of equipment, we could not compare our way of training to training as a whole.

Based on our equipment's capability, the number of chunks vary from 10 to 20 in our training.

6.2 Window Length Experiment

In our prior experiments with different window lengths and hop lengths, the performance of 25ms window length and 10ms hop length proved to be better than all other choices. And even the same model with the same feature selection can differ significantly if different window lengths are chosen.

There is one major constraint of frame-level fusion approach about window length configuration. All the front-end feature processing across fusion models have to conform to the same window length. As various features would have different characteristics, different features might require different window length and shift length to optimize its performance.

6.3 Fusion Methods Experiment

We conducted our experiments adding one base model at a time, and at both frame-level and instance-level. Generally, the more elements in fusion, the better the performance we get. And regardless of the performance of base models individually, as long as they effectively modeled both bona fide speech and spoof speech, they turned out having a positive contribution to the fusion model. The only exceptant was the silence feature, which is discussed below.

6.4 Modeling Silence

With leading and trailing silence lengths extracted as new features, we conducted experiments to test the classification performance when with other features. The t-DCF rose from 0.132 to 0.143 on evaluation set after silence features are included. However, our previous experiment showed that silence features individually had a 0.80 accuracy on spoof data in development set. These results showed that silence features are slightly effective when trained alone, but they did not bring improvement to other features in the fusion.

One possible reason is that as the evaluation set was generated with different spoofing techniques from the development set, which led to a quite different distribution of leading and trailing silences. Also, it's probable that differently forged spoof speech and differently recorded bona fide speech differ in their silence segments' acoustic features, so hand-tuned silence extracting algorithms didn't perform optimally and robustly on all data.

7 Conclusion

In this study, we proposed a new anti-spoof technique that incorporates non-linear model fusion and computed silence features, and conducted our experiments on ASVspoof 2019 dataset to testify its effectiveness. Experiment results showed great improvements compared to base models and our baseline, and ranked high among all ASVspoof 2019 contestants. We also discussed some crucial issues during model configuration and training, including the training procedure, selection of hyper parameters and the effect of different components.

An immediate but also rather time-consuming task that can be incorporated into the future work is that a more diverse set of window lengths can be entertained for the instance level fusion implementation, for which a uniform window length is currently experimented. Also, using machine learning techniques to obtain a more robust way to model silence might contribute to the fusion. Furthermore, the intermittent silence periods within each type of audio files may be considered as a more telling silence feature to contribute to the classification.

References

[1] Zhizheng Wu, Nicholas Evans, Tomi Kinnunen, Junichi Yamagishi, Federico Alegre, and Haizhou Li. Spoofing and countermeasures for speaker verification. *Speech Communication*, 66:130–153, 2015.

[2] Jesús Villalba and Eduardo Lleida. Detecting replay attacks from far-field recordings on speaker verification systems. In *BioID'11 Proceedings of the COST 2101 European conference on Biometrics and ID management*, pages 274–285, 2011.

[3] Zhi-Feng Wang, Gang Wei, and Qian-Hua He. Channel pattern noise based playback attack detection algorithm for speaker recognition. In *2011 International Conference on Machine Learning and Cybernetics*, volume 4, pages 1708–1713, 2011.

[4] Massimiliano Todisco, Héctor Delgado, and Nicholas WD Evans. A new feature for automatic speaker verification anti-spoofing: constant q cepstral coefficients. In *Odyssey*, volume 2016, pages 283–290, 2016.

[5] Massimiliano Todisco, Héctor Delgado, and Nicholas Evans. Constant q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45, 02 2017.

[6] Sandipan Chakroborty, Anindya Roy, and Goutam Saha. Improved closed set text-independent speaker identification by combining mfcc with evidence from flipped filter banks. *World Academy of Science, Engineering and Technology, International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 2:2554–2561, 2008.

[7] Md Sahidullah, Tomi Kinnunen, and Cemal Hanilçi. A comparison of features for synthetic speech detection. 09 2015.

[8] Mari Ganesh Kumar, Suvidha Rupesh Kumar, MS Saranya, B Bharathi, and Hema A Murthy. Spoof detection using time-delay shallow neural network and feature switching. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1011–1017. IEEE, 2019.

[9] Massimiliano Todisco, Xin Wang, Ville Vestman, Md Sahidullah, Héctor Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi Kinnunen, and Kong Aik Lee. Asvspoof 2019: Future horizons in spoofed and fake audio detection. *arXiv preprint arXiv:1904.05441*, 2019.

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

[11] Rohan Kumar Das. Known-unknown Data Augmentation Strategies for Detection of Logical Access, Physical Access and Speech Deepfake Attacks: ASVspoof 2021. In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 29–36, 2021.

[12] Hemlata Tak, Jose Patino, Andreas Nautsch, Nicholas Evans, and Massimiliano Todisco. Spoofing attack detection using the non-linear fusion of sub-band classifiers. *arXiv preprint arXiv:2005.10393*, 2020.

[13] Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE TRANSACTIONS on Information and Systems*, 99(7):1877–1884, 2016.

[14] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[15] Hemlata Tak, Jose Patino, Andreas Nautsch, Nicholas Evans, and Massimiliano Todisco. An explainability study of the constant q cepstral coefficient spoofing countermeasure for automatic speaker verification. *arXiv preprint arXiv:2004.06422*, 2020.

[16] Tomi Kinnunen, Kong Aik Lee, Hector Delgado, Nicholas Evans, Massimiliano Todisco, Sahidullah, Junichi Yamagishi, and Douglas A. Reynolds. t-dcf: a detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification. In *Speaker Odyssey 2018 The Speaker and Language Recognition Workshop*, pages 312–319, 2018.

[17] Xingming Wang, Xiaoyi Qin, Tinglong Zhu, Chao Wang, Shilei Zhang, and Ming Li. The DKU-CMRI System for the ASVspoof 2021 Challenge: Vocoder based Replay Channel Response Estimation. In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 16–21, 2021.

[18] Shentong Mo, Haofan Wang, Pinxu Ren, and Ta-Chung Chi. Automatic speech verification spoofing detection. *CoRR*, abs/2012.08095, 2020.