An Initial Investigation for Detecting Partially Spoofed Audio

Lin Zhang, Xin Wang, Erica Cooper, Junichi Yamagishi, Jose Patino, Nicholas Evans

An Initial Investigation for Detecting Partially Spoofed Audio

Lin Zhang[1,2], *Xin Wang*[1], *Erica Cooper*[1], *Junichi Yamagishi*[1,2], *Jose Patino*[3], *Nicholas Evans*[3]

[1]National Institute of Informatics, Japan [2]SOKENDAI, Japan
[3]Digital Security Department, EURECOM, France

{zhanglin, wangxin, ecooper, jyamagis}@nii.ac.jp, {patino, evans}@eurecom.fr

Abstract

All existing databases of spoofed speech contain attack data that is spoofed in its entirety. In practice, it is entirely plausible that successful attacks can be mounted with utterances that are only partially spoofed. By definition, partially-spoofed utterances contain a mix of both spoofed and bona fide segments, which will likely degrade the performance of countermeasures trained with entirely spoofed utterances. This hypothesis raises the obvious question: *'Can we detect partially-spoofed audio?'* This paper introduces a new database of partially-spoofed data, named PartialSpoof, to help address this question. This new database enables us to investigate and compare the performance of countermeasures on both utterance- and segmental- level labels. Experimental results using the utterance-level labels reveal that the reliability of countermeasures trained to detect fully-spoofed data is found to degrade substantially when tested with partially-spoofed data, whereas training on partially-spoofed data performs reliably in the case of both fully- and partially-spoofed utterances. Additional experiments using segmental-level labels show that spotting injected spoofed segments included in an utterance is a much more challenging task even if the latest countermeasure models are used.

Index Terms: partially-spoofed attack, countermeasures, variable-length input, segmental-level, deepfakes

1. Introduction

To address the problem of presentation attacks [1] on automatic speaker verification (ASV) systems, the ASVspoof challenge, which aims to promote the study and development of spoofing countermeasures (CMs), has been held biennially since 2015. Previous challenges [2, 3, 4] focused on different types of spoofing attacks at the utterance level, including the logical access (LA) scenario (i.e., speech synthesis and voice conversion attacks) and the physical access (PA) scenario (i.e., replay attacks). Year by year, various types of high-performance CMs have been proposed. In the latest competition, ASVspoof 2019 [4], several discriminative (Light CNN [5], ResNet [6], Wave-U-Net [7], RawNet [8]) and generative (GMM-UBM) models were proposed which achieved promising results.

When we use spoofing CMs for more general situations beyond presentation attacks against ASV, such as audio deepfake detection, it is obvious that the assumption that attacks will consist of entirely-spoofed utterances does not always hold – speech synthesis and voice conversion technologies may be used to generate only a part of an utterance, and such spoofed segment(s) may be injected into a bona fide utterance. For example, attackers may use speech synthesis to replace specific phrases that they want to manipulate. This leads us to consider the following new scientific questions: *'Can the latest CMs discriminate such partially-spoofed audio from bona fide audio reliably? Can we construct a new CM model that can detect such injected partially-spoofed segments?'* To investigate this as-yet neglected attack scenario deeply, we first collected a new partially-spoofed database, named "PartialSpoof", and used it to evaluate existing CMs in terms of both of utterance- and segmental-level detection of partially spoofed audio.

As for the existing CMs, we use discriminatively trained Light Convolutional Neural Networks (LCNN) [4, 5] using the P2SGrad loss function [9] as it is reported as the best single model. We train them using either utterance- or segmental-level labels and investigate their detection performance. We also revise the LCNN architecture slightly so that LCNN-based CMs can handle temporal information better when only small fractions of an utterance may be spoofed.

For experiments on utterance-level detection of partially spoofed audio, we have compared CMs trained to detect fully-spoofed data of the ASVspoof 2019 dataset with equivalent CMs trained on partially-spoofed data from the new PartialSpoof database. For experiments on segmental-level detection of partially spoofed audio, we demonstrate the performance of CMs using the segmetal-level labels instead of utterance-level labels.

This paper is structured as follows: Section 2 overviews the construction process of the PartialSpoof Database. Section 3 briefly introduces the CMs used for our investigations. Section 4 shows experimental conditions and results, and our findings are summarized in Section 5.

2. PartialSpoof Database

We built the PartialSpoof database[1] based on the ASVspoof 2019 LA database [10] since the latter covers 17 types of spoofed data produced by advanced speech synthesizers, voice converters, and hybrids. We used the same set of bona fide data from the ASVspoof 2019 LA database but created partially-spoofed audio from the ASVspoof 2019 LA data by following the steps below:

Step 1: We ran three types of publicly-available voice activity detection (VAD) algorithms: an energy-based VAD from Kaldi [11], another energy-based VAD [12], and an LSTM-based VAD from Pyannote [13, 14]. Then, we used their voting results to decide the boundaries of speech segments in order to reduce VAD errors. Specifically, we considered a segment to be speech if it was detected by at least two out of three VAD systems[2].

Step 2: According to the boundaries determined by the above VAD results, we replaced a randomly chosen segment from a bona fide utterance with a spoofed segment. We also considered the opposite direction for segment replacement, i.e., randomly

[1]Database: https://zenodo.org/record/4817532#.YLd8Yi2l1hF

[2]This voting of VAD systems can achieve an 11.98% detection error rate [15] when evaluated on TIMIT [16].

substituting a spoofed segment into a bona fide segment. There are a few restrictions: 1) The same inserted segment cannot appear more than once in a given carrier utterance; 2) The segment to be inserted must be close in duration to the original segment it replaces.

Step 3: To avoid potential artifacts when fusing the waveform of the inserted segment into the carrier audio file, we computed the time-domain cross correlation between the replacement segment and its adjacent segments to find the best fusion point. The fusion was then conducted through waveform overlap-add after waveform amplitude normalization using SV56 [17]. We kept 50% of the non-speech part in the head and tail of the segments so that overlap-add only happened in such non-speech parts without modifying the speech segment.

Step 4: We labeled each segment in the newly-created audio as *bona fide* if the segment is originally from bona fide audio, otherwise as *spoofed*. The utterance-level label of a partially-spoofed utterance is of course *spoofed*.

Step 5: We repeated Step 2-4 until we obtained the same number of spoofed trials as the ASVspoof 2019 database.

Thus, numbers of spoofed trials [3] in the train, dev, and eval sets are the same as those of the ASVspoof 2019 database.

3. Spoofing Countermeasures

To determine whether the latest CMs can discriminate partially-spoofed utterances from bona fide ones, we built a series of CMs based on the top single LCNN model in the ASVspoof 2019 LA task [4, 5] but with a few enhancements [9]. Here we explain how we train them for utterance- and segmental-level detection, respectively.

3.1. CM training for utterance-level detection

The LCNN used by the top single CM in the ASVspoof 2019 LA scenario only accepts fixed-size inputs [18]. The CM hence operates on trimmed or padded input speech [5] and is trained to predict an utterance-level score. This is unsuitable for the partially-spoofed scenario because the intervals of interest, e.g., the replaced, substituted, or spoofed intervals, might well be among the trimmed segments.

We therefore need to slightly enhance the LCNN with temporal pooling strategies to better process variable-length speech inputs [9]. Let $\boldsymbol{x}_{1:N^{(j)}} \equiv (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{N^{(j)}}) \in \mathbb{R}^{N^{(j)} \times D}$ be the input feature sequence of the j-th trial with $N^{(j)}$ frames, where $\boldsymbol{x}_n$ is the feature for the n-th frame. Instead of trimming or padding $\boldsymbol{x}_{1:N^{(j)}}$ to a fixed shape, we let the LCNN transform $\boldsymbol{x}_{1:N^{(j)}}$ into $\boldsymbol{h}_{1:N^{(j)}/L} = (\boldsymbol{h}_1^{(j)}, \cdots, \boldsymbol{h}_{N^{(j)}/L}^{(j)})$, where L is decided by the convolution stride. Then, we pool an utterance-level vector $\boldsymbol{o}_j = \sum_{m=1}^{N^{(j)}/L} w_m^{(j)} \boldsymbol{h}_m^{(j)}$ and use it for scoring. The pooling weight $w_m^{(j)}$ can be computed using a self-attentive pooling (SAP) strategy [19] or be uniform, i.e., the average pooling (AP) strategy. The enhanced LCNNs are illustrated in Figure 1.

We may optionally use a bi-directional LSTM to process $\boldsymbol{h}_{1:N^{(j)}/L}$ before pooling. The reason is that convolution in an LCNN has a fixed receptive field, and each $\boldsymbol{h}_m$ covers only a fixed number of input frames. A pure LCNN hence neglects the temporal change that may be useful in the partially-spoofed scenario. In practice, we add a skip-connection over the Bi-LSTM layer(s) to stabilize the training process. This optional

[3] Samples: https://nii-yamagishilab.github.io/zlin-demo/IS2021/index.html

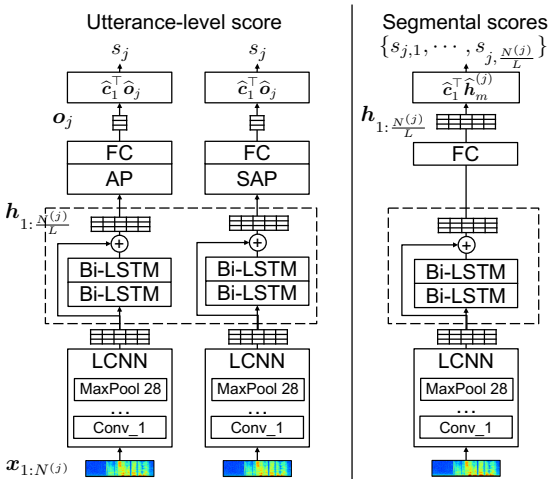

Figure 1: *Enhanced LCNNs for conventional and partially-spoofed scenarios. The LCNN part is identical to that in [5] (from layer Conv_1 to MaxPool_28). FC denotes a fully-connected layer. AP and SAP denote average and self-attentive pooling, respectively. The Bi-LSTM block in the dashed frame is optional.*

Bi-LSTM block is illustrated in Figure 1.

The enhanced LCNNs are trained given pairs of input audio and utterance-level labels $\{\boldsymbol{x}_{1:N^{(j)}}, y_j\}_{j=1}^{|\mathcal{D}|}$, where $|\mathcal{D}|$ is the size of the training data set $\mathcal{D}$. In this paper, we used a new loss function called *MSE for P2SGrad* since it was found to be more efficient than cross-entropy with variants of softmax on this task [9]. The loss is computed as

$$\mathcal{L}^{(\text{p2s})} = \frac{1}{|\mathcal{D}|} \sum_{j=1}^{|\mathcal{D}|} \sum_{k=1}^{C} (\cos\theta_{j,k} - \mathbb{1}(y_j = k))^2, \quad (1)$$

where $\mathbb{1}(\cdot)$ is an indicator function, C is the number of target classes, and $\cos\theta_{j,k} = \widehat{\boldsymbol{c}}_k^\top \widehat{\boldsymbol{o}}_j$ is the cosine distance between the length-normed vector $\widehat{\boldsymbol{o}}_j = \boldsymbol{o}_j / ||\boldsymbol{o}_j||$ and the class vector $\widehat{\boldsymbol{c}}_k$ for the k-th target class. In this paper, we set $C = 2$ and use $k = 1$ and $k = 2$ to denote *bona fide* and *spoofed*, respectively. During inference, the model uses $s_j = \cos\theta_{j,1} = \widehat{\boldsymbol{c}}_1^\top \widehat{\boldsymbol{o}}_j$ as the utterance-level score for the j-th test trial.

3.2. Deriving segmental scores from an utterance-level score

The enhanced LCNNs produce one score per trial. If they are used for segmental-level detection rather than utterance-level detection, then we need to decompose the utterance-level score and derive a score for each segment of the input trial. Here we describe our decomposition procedures.

Suppose the LCNN has conducted $\boldsymbol{x}_{1:N^{(j)}} \mapsto \boldsymbol{h}_{1:M_j} \mapsto \boldsymbol{o}_j = \sum_{m=1}^{M_j} w_m^{(j)} \boldsymbol{h}_m^{(j)}$, where $M_j = \frac{N^{(j)}}{L}$. With the utterance-level score $s_j = \cos\theta_{j,1} = \widehat{\boldsymbol{c}}_1^\top \frac{\boldsymbol{o}_j}{||\boldsymbol{o}_j||}$, we get

$$s_j = \widehat{\boldsymbol{c}}_1^\top \frac{\sum_{m=1}^{M_j} w_m^{(j)} \boldsymbol{h}_m^{(j)}}{||\boldsymbol{o}_j||} = \frac{1}{M_j} \sum_{m=1}^{M_j} \tilde{w}_m^{(j)} \cos\theta_{j,1,m}, \quad (2)$$

where $\tilde{w}_m^{(j)} = w_m^{(j)} M_j \frac{||\boldsymbol{h}_m^{(j)}||}{||\boldsymbol{o}_j||}$ and $\cos\theta_{j,1,m} = \widehat{\boldsymbol{c}}_1^\top \widehat{\boldsymbol{h}}_m^{(j)}$. We define $s_{j,m} \equiv \tilde{w}_m^{(j)} \cos\theta_{j,1,m}$ as the score of the m-th segment

in the j-th trial, which measures the weighted cosine distance between the bona fide class vector $\widehat{\boldsymbol{c}}_1$ and the feature vector $\widehat{\boldsymbol{h}}_m$. Note that $s_{j,m}$ can be larger than one, and the average of $s_{j,m}$ is equal to s_j. Also note that the decomposition is valid even if $\boldsymbol{o}_j = \mathcal{F}(\sum_{m=1}^{M_j} w_m^{(j)} \boldsymbol{h}_m^{(j)})$ where $\mathcal{F}(\cdot)$ is a linear or affine transformation (i.e., a FC layer).

3.3. CM training for segmental-level detection

In the previous section, we derive segmental scores from an utterance-level detection score. Alternatively, we may train CMs with segment-level labels included in the PartialSpoof database and we may infer a sequence of segment-level scores directly, as Figure 1 illustrates.

This segmental CM may use the same LCNN as those in Sec. 3.1 but with the pooling layer excluded. After converting $\boldsymbol{x}_{1:N^{(j)}} \mapsto \boldsymbol{h}_{1:M_j}$ using the LCNN, the segmental CM computes $\cos\theta_{j,k,m} = \widehat{\boldsymbol{c}}_k^\top \widehat{\boldsymbol{h}}_m^{(j)}, \forall m \in [1, M_j]$ for each segment. Thus, its training loss becomes

$$\mathcal{L}^{(\text{seg})} = \frac{1}{|\mathcal{D}|}\frac{1}{M_j}\sum_{j=1}^{|\mathcal{D}|}\sum_{m=1}^{M_j}\sum_{k=1}^{C}(\cos\theta_{j,k,m} - \mathbb{1}(y_{j,m}=k))^2, \tag{3}$$

where $y_{j,m}$ is the label for the m-th segment in the j-th trial. During inference, we can directly use $s_{j,m} = \cos\theta_{j,1,m}$ as the segment score for the m-th segment.[4]

4. Experiments

We introduce experimental configurations for our CMs in Sec. 4.1. Then, we discuss the performance of utterance- and segmental-level detection in Sec. 4.2 and Sec. 4.3, respectively.

4.1. Experimental configurations

All the CMs used linear frequency cepstral coefficients (LFCCs) as input acoustic features. LFCCs were extracted using the same configuration as the ASVspoof 2019 baseline: frame length of 20 ms, frame shift of 10 ms, 512-point FFT, linear filter-bank with 20 channels, and a combination of static, delta, and delta-delta coefficients, thus 60 dimensions for each frame. We did not use any data augmentation, voice activity detection, or feature normalization.

The LCNN component[5] in all the CMs was based on the original LCNN-based CM [5]. Accordingly, the input LFCCs $\boldsymbol{x}_{1:N^{(j)}}$ are converted into $\boldsymbol{h}_{1:N^{(j)}/16}$ before pooling. In this way, embedding can be extracted every 0.16 seconds. Training was conducted with the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$) [20] and a batch size of 64. The learning rate started from 3×10^{-4} and was halved for every 10 epochs. The LCNN was trained mutiple times with random initialization. Results shown in the next sections are the mean of six rounds [6]. All results are reproducible using the same random seed and GPU (Nvidia Tesla V100) environment.

Evaluation was conducted using the Equal Error Rate (EER) and minimum tandem detection cost function (min-tDCF) [21, 22], both of which are computed following the official routines from ASVspoof 2019. For min-tDCF evaluation, we defined 63,882 spoofed trials for the evaluation set and 22,296 for the development set, based on the new PartialSpoof database. Bona fide (target and non-target) trials are identical to those of the standard ASVspoof 2019 protocols.

[4] If it is necessary to produce a single utterance-level score, one possibility is to use $s_j = \min_m s_{j,m}$ as the score for the j-th trial. This is because a segment with a smaller score is more likely to be spoofed, and a spoofed segment declares a spoofed trial (see step 3 of Sec. 2).

[5] Specifically, from layer Conv_1 to MaxPool_28.

[6] More results can be found in arXiv: https://arxiv.org/abs/2104.02518

Table 1: *Ablation study of LCNN countermeasures against partially-spoofed audio.*

Pooling types	Smoothing types	EER(%) Dev.	EER(%) Eval.	min-tDCF Dev.	min-tDCF Eval.
AP	-	3.90	7.78	0.0981	0.1972
SAP	-	3.90	7.45	0.1033	0.1887
AP	Bi-LSTM	**3.68**	**6.19**	**0.1003**	0.1645
SAP	Bi-LSTM	3.84	6.23	0.1064	**0.1609**

Table 2: *Cross-database study for investigating how training data mismatch affects CM performance. The best architecture (Utterance + AP + Bi-LSTM) in Table 1 has been selected.*

	Train	ASVspoof 2019 Dev.	ASVspoof 2019 Eval.	PartialSpoof Dev.	PartialSpoof Eval.
EER(%)	ASVspoof 2019	0.21	2.65	9.59	15.96
	PartialSpoof	4.28	5.38	3.68	6.19
min-tDCF	ASVspoof 2019	0.0060	0.0640	0.1854	0.3003
	PartialSpoof	0.1156	0.1713	0.1003	0.1645

4.2. Experiments for utterance-level detection

4.2.1. Ablation study of LCNN CMs against PartialSpoof

We first show CM performance on utterance-level detection. Table 1 shows an ablation study of the enhanced LCNN CMs against partially-spoofed audio at the utterance level. All four CMs were trained with utterance-level labels (Sec. 3.1), and detections were also done at the utterance level. We can see that the AP + Bi-LSTM system yields superior performance; thus it was chosen for further cross-database experiments in Section 4.2.2.

4.2.2. Cross-database investigation for training data mismatch

We investigated how training data mismatch affects CM performance on utterance-level detection. More specifically, we investigated two questions: Can the LCNNs trained using partially(entirely)-spoofed audio detect entirely(partially)-spoofed utterances? Thus, we trained CMs on the PartialSpoof database and evaluated it on the evaluation set of the ASVspoof 2019 database and vice versa. We also included the matched (in-domain) cases as reference. Results are shown in Table 2.

By comparing results in each set of the ASVspoof 2019 dataset with counterparts of the PartialSpoof dataset, we can first see that the EER for the ASVspoof 2019 dataset is always lower than that for the PartialSpoof dataset on both the sets, showing that partially-spoofed data is more difficult to detect than fully-spoofed data.

Next we can see that when the ASVspoof 2019-trained model is evaluated on the PartialSpoof database, performance degrades significantly; the EER increases from 0.21% to 9.59% and from 2.65% to 15.96% for development and evaluation sets, respectively. On the other hand, the model trained on the partially-spoofed database is relatively robust and shows a stable EER when evaluated on both databases. The models trained on fully-spoofed utterances appear to lack generalization ability and thus overfit to the fully-spoofed case.

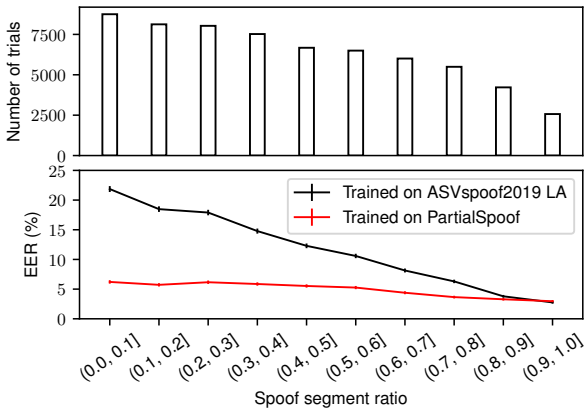

Figure 2: *Break-down of results of the cross-database study. (a) Histogram of number of trials having different spoof segment ratios. The ratio was quantized for visualization purposes. (b) EERs for each of the quantized spoof segment ratio classes with confidence intervals at a significance level of 5% [23]. The best LCNN network (AP + Bi-LSTM) in Table 1 has been selected.*

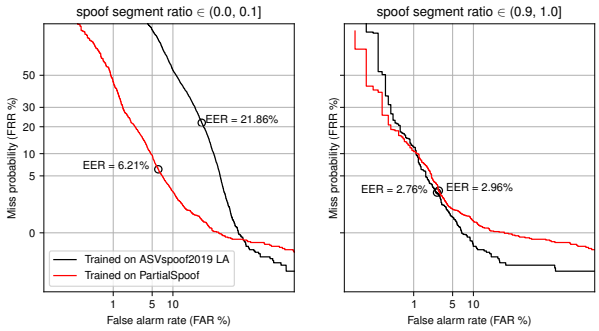

Figure 3: *Comparison of DET curves in the eval. set of the PartialSpoof database at areas of different spoof segment ratios.*

4.2.3. Analysis based on spoof segment ratios

Since the PartialSpoof database was constructed from the random replacement of speech segments, the total duration occupied by multiple injected spoofed segments within in an utterance can vary. We refer to the ratio of the total duration of spoofed segments within an entire length of audio as the "spoof segment ratio" and further investigate how the detection performance changes according to the spoof segment ratio of the trials. For this analysis, we quantized the spoof segment ratio into 10 bins and computed the EER for each bin separately.

Figure 2 shows the relationship between the quantized spoof segment ratio and performance. The top histogram presents the number of trials included in each bin. We evaluate EER for each bin using its corresponding spoofed trials and all bona fide trials and then obtain the bottom two curves in the lower plot. From this figure, we can confirm that the spoof segment ratio has a significant impact upon the CM performance. More specifically, we see that when the CM model is trained using fully-spoofed audio, as expected, the EER degrades with decreases in the spoof segment ratio. On the other hand, the CM model trained using partially spoofed audio is robust to changes in the spoof segment ratio. Even if the ratio changes, EER values do not change significantly.

Figure 3 shows the DET curves at the smallest and largest spoof segment ratios of the PartialSpoof evaluation set. From Figure 2 and Figure 3, we can reconfirm that the reliability of countermeasures trained to detect fully-spoofed data degrade substantially when tested with partially-spoofed data.

Table 3: *Comparison of segmental detection performance of CMs trained using segmental or utterance-level labels.*

Train labels	Pooling types	Smoothing types	EER(%) Dev.	EER(%) Eval.
Utterance	AP	Bi-LSTM	37.02	40.20
Segment	-	Bi-LSTM	6.81	16.21

4.3. Experiments for segmental-level detection

Next we focus on segmental-level detection. This is challenging because an individual segment can be very short. The main focus is a CM trained using the segmental labels instead of utterance labels as described in Sec. 3.3. For reference, we compare it with another CM trained using utterance labels and their segmental scores derived from an utterance-level score in the manner described in Sec. 3.2.

Table 3 shows the segmental detection results. Not surprisingly, the CM trained using the segmental labels is better than the one trained using utterance-level labels. This shows that the segmental labels included in the PartialSpoof database are useful for segmental detection, and that segment detection is feasible. But we can also see that this is a more challenging task than the utterance-level detection in the previous section, and the CMs have obvious room for further improvement.

5. Conclusions

To answer the original question: *'can we detect partially spoofed audio?'*, we built a new PartialSpoof database consisting of bona fide and partially-spoofed utterances based on ASVspoof 2019. Since PartialSpoof audio is composed of bona fide and spoofed segments(s), it can be trained and evaluated on both utterance- and segmental- level labels. For utterance-level detection, cross-database analyses on partially- and fully-spoofed data were conducted to investigate how data mismatch affects CM performance. We also carried out a more challenging segmental detection task to see whether CMs can spot short spoofed segments included in an utterance.

Generally, both utterance- and segmental-level detection on PartialSpoof are more challenging than on the fully-spoofed database. The reliability of countermeasures trained to detect fully-spoofed data was also found to degrade substantially when tested with partially-spoofed data, while training on partially-spoofed data led to stable performance when evaluating on both fully- and partially-spoofed utterances.

Future studies are needed to understand the data mismatch problem deeply. Furthermore, random segment selection and concatenation using cross-correlation may not be the best way to build a partially-spoofed database. Linguistic information, contextual information, and rhythm can be lost during this process. Further exploration of more appropriate databases and more robust CMs with higher precision are needed.

Acknowledgements Thanks Dr. Ville Vestman from University of Eastern Finland for sharing an ASV model for min-tDCF evaluation in this paper. This study was partially supported by the Japanese-French joint national VoicePersonae project supported by JST CREST (JPMJCR18A6) and the ANR (ANR-18-JSTS-0001), JST CREST Grants (JPMJCR20D3), MEXT KAKENHI Grants (16H06302, 18H04120, 18H04112, 18KT0051), Japan, and Google AI for Japan program.

6. References

[1] A. K. Jain, A. Ross, and S. Pankanti, "Biometrics: a tool for information security," *IEEE transactions on information forensics and security*, vol. 1, no. 2, pp. 125–143, 2006.

[2] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "ASVspoof 2015: the First Automatic Speaker Verification Spoofing and Countermeasures Challenge," in *Proc. Interspeech*, 2015, pp. 2037–2041.

[3] T. Kinnunen, M. Sahidullah, H. Delgado, M. Todisco, N. Evans, J. Yamagishi, and K. A. Lee, "The ASVspoof 2017 Challenge: Assessing the Limits of Replay Spoofing Attack Detection," in *Proc. Interspeech*, 2017, pp. 2–6.

[4] A. Nautsch, X. Wang, N. Evans, T. Kinnunen, V. Vestman, M. Todisco, H. Delgado, M. Sahidullah, J. Yamagishi, and K. A. Lee, "ASVspoof 2019: spoofing countermeasures for the detection of synthesized, converted and replayed speech," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2021.

[5] G. Lavrentyeva, S. Novoselov, A. Tseren, M. Volkova, A. Gorlanov, and A. Kozlov, "STC Antispoofing Systems for the ASVspoof2019 Challenge," in *Proc. Interspeech*, 2019, pp. 1033–1037.

[6] T. Chen, A. Kumar, P. Nagarsheth, G. Sivaraman, and E. Khoury, "Generalization of Audio Deepfake Detection," in *Proc. Odyssey*, 2020, pp. 132–137.

[7] B. Chettri, D. Stoller, V. Morfi, M. A. M. Ramírez, E. Benetos, and B. L. Sturm, "Ensemble Models for Spoofing Detection in Automatic Speaker Verification," in *Proc. Interspeech*, 2019, pp. 1018–1022.

[8] H. Tak, J. Patino, M. Todisco, A. Nautsch, N. Evans, and A. Larcher, "End-to-end anti-spoofing with rawnet2," in *Proc. ICASSP*, 2021, pp. 6369–6373.

[9] X. Wang and J. Yamagishi, "A comparative study on recent neural spoofing countermeasures for synthetic speech detection," in *Proc. Interspeech*, 2021 (to appear).

[10] 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 and Language*, vol. 64, p. 101114, 2020.

[11] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[12] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech communication*, vol. 52, no. 1, pp. 12–40, 2010.

[13] M. Lavechin, M.-P. Gill, R. Bousbib, H. Bredin, and L. Paola Garcia-Perera, "End-to-end Domain-Adversarial Voice Activity Detection," in *Proc. Interspeech*, 2020, pp. 3685–3689.

[14] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M. Gill, "Pyannote.audio: Neural building blocks for speaker diarization," in *Proc. ICASSP*, 2020, pp. 7124–7128.

[15] H. Bredin, "pyannote.metrics: a toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems," in *Proc. Interspeech*, Stockholm, Sweden, August 2017.

[16] V. Zue, S. Seneff, and J. Glass, "Speech database development at MIT: Timit and beyond," *Speech Communication*, vol. 9, no. 4, pp. 351–356, aug 1990.

[17] International Telecommunication Union, Recommendation G.191: Software Tools and Audio Coding Standardization, Nov 11 2005.

[18] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018.

[19] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-Attentive Speaker Embeddings for Text-Independent Speaker Verification," in *Proc. Interspeech*, 2018, pp. 3573–3577.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2014.

[21] T. Kinnunen, K. A. Lee, H. Delgado, N. Evans, M. Todisco, M. Sahidullah, J. Yamagishi, and D. A. Reynolds, "t-DCF: a detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification," in *Proc. Odyssey*, 2018, pp. 312–319.

[22] T. Kinnunen, H. Delgado, N. Evans, K. A. Lee, V. Vestman, A. Nautsch, M. Todisco, X. Wang, M. Sahidullah, J. Yamagishi, and D. A. Reynolds, "Tandem Assessment of Spoofing Countermeasures and Automatic Speaker Verification: Fundamentals," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2195–2210, 2020.

[23] S. Bengio and J. Mariéthoz, "A statistical significance test for person authentication," in *Proc. Odyssey*, 2004.