Audio Deepfake Detection under Post-Processing Attack

Karla Schäfer
Fraunhofer SIT, ATHENE
Darmstadt, Germany
karla.schaefer@sit.fraunhofer.de

Jeong-Eun Choi
Fraunhofer SIT, ATHENE
Darmstadt, Germany
jeong-eun.choi@sit.fraunhofer.de

Martin Steinebach
Fraunhofer SIT, ATHENE
Darmstadt, Germany
martin.steinebach@sit.fraunhofer.de

***Abstract*—Generalizable audio deepfake detection is a challenging task. Simple post-processing attacks like background noise or impulse response can significantly affect the performance of the detectors. We analysed the effects of 13 post-processing attacks on two detectors, one with a SSL (Self-Supervised Learning)-based front-end (Wav2Vec 2.0) the other using SincNet for feature extraction. Both detectors showed significant performance degradation when applying the post-processing attacks. For instance, we calculated an EER of 0.73% on the original data of the in-the-wild dataset using the SSL-based detector. The performance dropped to 4.37% after applying impulse response. To find the most effective attacks, we analysed the effects of post-processing on their signal quality using UTMOS. Additionally, we explored retraining strategies, improving the overall performance of our detectors by an EER of 0.22% and 0.33%.**

***Index Terms*—audio deepfakes, detection, post-processing, effective attacks.**

I. Introduction

Recent advances in artificial intelligence and speech synthesis have facilitated the creation of highly realistic speech recordings. With advances in voice conversion (VC) and text-to-speech synthesis (TTS), the demand for robust and generalizable deepfake detectors has increased. However, most of the current work uses the raw output as input for detection, neglecting possible deliberate manipulations, such as post-processing attacks.The ASVspoof 2021 [1] and ASVspoof5 [2] datasets introduced more realistic data by including distortions caused by telephone channel transmission and compression. However, these datasets fail to consider other post-processing techniques, such as adding background noise or impulse response effects. Consequently, these datasets do not fully explore the range of strategies attackers might use to bypass detection systems. Mostly, audio deepfake detectors can be divided in a front-end and a back-end (classifier). We trained two detectors, one with a SSL (self-supervised learning)-based front-end and one without SSL, using SincNet [3], and tested both detectors on various post-processing attacks. To assess their generalizability, we used the in-the-wild (ITW) dataset [4], a widely recognized dataset for this task, along with the ASVspoof 2021 LA and DF splits [1]. We applied 13 post-processing attacks with different parameter settings on the test sets. Furthermore, we performed a quality evaluation using UTMOS, allowing us to identify those audio post-processing attacks that are most effective, i.e. most harmful. Subsequently, we propose to improve and harden the detectors by re-training/fine-tuning them with harmful post-processing attacks, thereby increasing their robustness against these attacks.

This research work was supported by ATHENE in the project DREAM and the Federal Office for Information Security (BSI) in the project SecMedID.

II. Experimental Methodology

A. Implementation of the Detectors

As SSL-based detector, we used a Wav2Vec2.0 XLSR (2b) front-end with a GF back-end [5]. For the implementation we used the code of (LL)GF[1]. For Wav2Vec 2.0 XLSR we used the S3PRL library[2]. For comparison, we also trained a non-SSL based detector using a SincNet [3] front-end and AASIST [6] as back-end, based on our previous work [7] and the work of [8][3]. For training, we used a combination of an internal dataset, ASVspoof 2019 LA [9], WaveFake [10], its genuine counterpart LJSpeech [11], LibriSeVoc [12] and FoR [13]. The loss was calculated using cross-entropy loss for all back-ends. All recordings used for training were trimmed and padded to approx. 4 sec. of audio clips. The detectors were trained for 5 epochs (batch size: 16). For data augmentation we used RawBoost [14], the pedalboard library[4] (audio compressor, Ladderfilter, Reverb and/or a Limiter) and the audiomentations library[5] adding Gaussian noise and mp3 compression randomly. As evaluation metric, we used the commonly used equal error rate (EER).

B. Post-Processing Attacks

We applied 13 post-processing attacks on the spoofed samples of the test sets. The first eight post-processing methods were created using the audiomentations library. For the remaining attacks, we used torchaudio using PyTorch. The applied post-processing methods are:

- **Gaussian Noise** Add noise to the sample to blur the speech recording. We set the minimum and maximum noise application factor to min_amplitude=0.001 and max_amplitude=0.015.

[1]https://github.com/nii-yamagishilab/project-NN-Pytorch-scripts/tree/master/project/07-asvspoof-ssl

[2]https://s3prl.github.io/s3prl/

[3]https://github.com/TakHemlata/SSL_Anti-spoofing

[4]https://spotify.github.io/pedalboard/

[5]https://iver56.github.io/audiomentations/

- **Background (bkgd) Noise** We applied clapping and church bells using the ESC-50 dataset. (min_snr_in_db=3.0 dB, max_snr_in_db=30.0 dB).[6]
- **Room Impulse Response (IR)** We applied impulse response characteristics of a "bedroom" and "car" using the MIT McDermott dataset.[7]
- **RoomSimulator** Adds reverb without further libraries (ShoeBox room simulation).
- **BandpassFilter** A bandpass filter is a filter that only allows signals in one frequency band to pass through. The frequency ranges below and above the passband are blocked or significantly attenuated. We used its default setting, with minimum center frequency: 100 Hz, maximum center frequency: 6000 Hz.
- **LowShelfFilter** The low shelf filter attenuates all sound components above the set frequency threshold. We used its default setting, with minimum center frequency: 50 Hz, maximum center frequency: 4000 Hz.
- **Timestretch** Time stretching changes the playback speed of audio material without changing the pitch. We used 2 different parameter values for the change of the total duration of the signal, which were a min_rate of 0.6 and a min_rate of 0.8.
- **Resample** Resample reduces or increases the sampling rate (original 16.000 Hz). We resampled to 15.500 Hz, 15.000 Hz, 16.500Hz and 17.000 Hz, i.e. minor change of -1000 to +1000 Hz.
- **Volume** Change the volume of the audio recording. We used the gain type amplitude with gains of 0.5 and 0.1 (positive amplitude ratio).
- **Fading** Adding a fade in and fade out to a waveform. We used three different fade shapes: linear, logarithmic, exponential and a fade in length of 1 second, fade out length of 2 seconds, which were the default values. For shorter audio files, the length of the file was used.
- **Mp3Compression** Compress the audio using an MP3 encoder to lower the audio quality. We used the audiomentations library with min bitrate of 8 kBit/s and max bitrate of 64 kBit/s and torchaudio with a bit rate of 192 kBit/s.
- **Encoder vorbis** Encode the audio using a vorbis encoder. We used the AudioEffector implementation of torchaudio and used the vorbis encoder, an audio format for lossy audio data reduction, to encode to the ogg format. ¡
- **Encoder opus** Encode the audio using an opus encoder with the format ogg. We used the AudioEffector implementation of torchaudio and used the opus encoder to encode to the ogg format.

III. Quality Evaluation

First, we tested the quality of the post-processed audio recordings using the UTMOS metric[8]. UTMOS expresses quality in the range of 1-5, in which UTMOS=1 represents the bad quality, hence worst quality and UTMOS=5 represents excellent quality. UTMOS can be used for predicting MOS (Mean Opinion Score for subjective quality evaluation) automatically and achieved the first place at the VoiceMOS Challenge 2022 [15]. The results of UTMOS on the spoofed samples can be found in Table I.

TABLE I
UTMOS of the post-processed Spoofs (ITW) - [bold: UTMOS greater or equal to the mean UTMOS of original (spoofed) samples; gray: comparable UTMOS]

Post-processing Method	median	mean	min	max
original data	2.62	2.57	1.23	4.28
gaussian noise	1.36	1.59	1.23	3.92
bkgd noise (clapping)	2.62	**2.57**	1.23	4.28
bkgd noise (church bells)	1.80	1.91	1.22	4.07
IR (bedroom)	1.78	1.86	1.23	3.59
IR (car)	1.63	1.80	1.23	3.58
roomsimulator	2.53	2.47	1.24	4.21
timestretch (0.8)	1.30	1.35	1.22	3.71
timestretch (0.6)	1.29	1.31	1.22	3.43
bandpassfilter	1.42	1.66	1.23	4.03
lowshelffilter	2.57	2.53	1.23	4.26
resample: 15.500	2.62	**2.58**	1.24	4.21
resample: 15.000	2.63	**2.59**	1.23	4.30
resample: 16.500	2.60	2.55	1.23	4.24
resample: 17.000	2.59	2.53	1.23	4.18
volume (gain:0.5)	2.61	2.56	1.23	4.29
volume (gain:0.1)	2.41	2.35	1.22	4.22
fading (linear)	2.55	2.50	1.22	4.21
fading (logarithmic)	2.61	**2.57**	1.23	4.27
fading (exponential)	2.47	2.43	1.22	4.17
Encoder vorbis (ogg)	2.61	2.56	1.23	4.27
Encoder opus (ogg)	2.50	2.43	1.23	4.20
mp3compression	2.44	2.36	1.23	4.18
AudioEffector; mp3	2.61	**2.57**	1.23	4.29

The lowest UTMOS were observed in samples manipulated with *time stretching* (mean: 1.35 and 1.31). In comparison, the original spoof samples had a mean UTMOS of 2.57. Notably, only 5 post-processing attacks were able to maintain similar UTMOS (see bold in Table I). Additionally, certain attacks, such as the *lowshelffilter*, achieved comparable UTMOS (greater than 2.30, see gray in Table I), while 7 post-processing attacks achieved UTMOS scores below 2.

IV. Detectors: Initial Performance

The performance of the detectors on the original dataset and the post-processed samples of the ITW and ASVspoof 2021 LA and DF test sets is presented in Table II. The results deteriorated due to the different post-processing methods. Interestingly, for both, the ASVspoof 2021 LA and DF splits and for both detectors, the top5 most effective attacks, i.e. attacks with the most deteriorated EER, were *background noise* (clapping and church bells), *bandpassfilter*, and *impulse response* (bedroom and car).

Similarly, for ITW, the worst results were calculated on the attacks *impulse response* (bedroom) with an EER of 4.37%, followed by *time stretching* (min_rate 0.6) with an EER of 2.17% and an EER of 1.78% for min_rate 0.8. For the non-SSL-based detector, on the original ITW test set we calculated an EER of 2.52%. The worst results were observed with *background noise* (clapping and church bells), yielding EERs of 16.77% and 7.64%, respectively, followed by the *impulse*

[6] https://github.com/karolpiczak/ESC-50
[7] https://mcdermottlab.mit.edu/Reverb/IR_Survey.html
[8] https://huggingface.co/spaces/sarulab-speech/UTMOS-demo

TABLE II
INITIAL PERFORMANCE (EER; %) ON THE POST-PROCESSED ITW AND ASVSPOOF 2021 LA AND DF DATASETS - [BOLD: BETTER SCORES THAN ORIGINAL. GREY: TOP5 MOST EFFECTIVE ATTACKS (I.E. ATTACKS WITH THE MOST DETERIORATED EER). RED: UTMOS ITW MEAN < 2 (SEE TABLE I)]

	ITW		ASV2021 LA		ASV201 DF	
Post-processings	SSL	NoSSL	SSL	NoSSL	SSL	NoSSL
original data	0.73	2.52	1.93	19.82	0.35	11.35
gaussian noise	1.05	4.15	**1.92**	**18.58**	0.44	11.70
bkgd (clapping)	1.67	16.77	3.52	29.08	1.15	22.27
bkgd (church bells)	1.28	7.64	3.06	28.77	0.88	20.13
IR (bedroom)	4.37	6.45	6.14	32.91	2.18	21.81
IR (car)	1.14	6.44	2.78	28.94	0.73	19.82
roomsimulator	0.81	2.61	2.47	22.54	0.50	15.17
timestretch (0.8)	1.78	2.64	**1.65**	**17.35**	0.46	**11.17**
timestretch (0.6)	2.17	3.52	1.93	21.41	0.57	14.85
bandpassfilter	1.75	5.26	3.57	25.03	1.38	16.86
lowshelffilter	0.75	2.75	2.23	**19.62**	0.38	11.56
resample: 15.500	0.75	**2.46**	2.02	**19.65**	**0.33**	**11.20**
resample: 15.000	0.78	**2.22**	2.03	**19.77**	**0.34**	**11.27**
resample: 16.500	0.74	2.81	2.23	**19.81**	0.38	**11.23**
resample: 17.000	0.75	3.12	2.40	20.01	0.42	**11.24**
volume (gain:0.5)	0.75	3.02	2.15	21.45	0.35	11.48
volume (gain:0.1)	0.81	6.45	2.17	23.54	0.35	12.13
fading (linear)	**0.64**	3.39	**1.52**	20.65	**0.24**	**10.61**
fading (logarithmic)	**0.67**	2.80	**1.52**	20.52	**0.24**	**11.14**
fading (exponential)	**0.68**	4.14	**1.54**	21.08	**0.24**	**10.58**
Encoder vorbis (ogg)	**0.65**	2.74	**1.52**	**19.56**	**0.24**	11.54
Encoder opus (ogg)	0.74	**2.12**	**1.86**	**15.44**	**0.28**	**8.40**
mp3compression	0.83	3.53	2.23	20.58	0.47	12.93
AudioEffector; mp3	0.74	2.60	2.12	20.09	**0.34**	11.59

response filter bedroom and car with EERs of 6.45% and 6.44% respectively. In general, we observed similar behaviours across the datasets.

Interestingly, using *fading* (in all three settings) resulted in improved performance on all three test sets when using the SSL-based detector. For example, on the ITW the EER improved from 0.73% to 0.64% (linear), 0.67% (logarithmic), and 0.68% (exponential). A similar behaviour can be seen on the *vorbis encoder*, *opus encoder* and reducing the *sample rate* by -500 and -1000 Hz.

To evaluate the effectiveness of the different attacks, we compared the mean UTMOS of Table I with the EERs of Table II, focusing on the ITW test set. For the SSL-based detector we could observe modest negative correlations between UTMOS and EER with a pearson correlation coefficient (PCC) of -0.574 and spearman rank correlation coefficient (SRCC) of -0.615. In terms of attacks, we found that attacks producing samples with relatively lower UTMOS compared to the original (< 2, red written in Table II), generally resulted in EERs higher than 0.9% (e.g. *Gaussian noise*, *background noise*), while attacks with relatively high UTMOS (> 2.3) achieved an EER lower than 0.83%, with the only exception being *background noise* with clapping.

For the non-SSL-based detector, the SRCC between UTMOS and EER was -0.470, indicating also a similar modest correlation, though weaker than the correlation observed with the SSL-based model (-0.615). However, the PCC was only -0.082, showing almost no direct correlation between EERs and UTMOS, but rather a correlation in terms of ranking. In terms of attacks, we observed slight differences in behaviour where lower UTMOS scores did not always result in higher EERs (*timestretching*, rate 0.8) and where relatively high UTMOS scores still led to higher EERs (*background noise* clapping, *resample*, *volume* (0.5; 0.1), *fading* and *mp3compression*).

These results highlight significant correlations between the UTMOS of post-processed samples and the performance of the detectors, as measured by EER. We found that the quality of the post-processed samples influences the performance of the detectors, particularly for the SSL-based model, which we will explore in more depth in the following section.

V. EFFECTIVENESS OF ATTACKS: IMPACT OF QUALITY

In a subsequent analysis, we evaluated the effectiveness of the attacks in depth by identifying VERY EFFECTIVE, EFFECTIVE and LESS EFFECTIVE attacks (see Table III). First, we identified VERY EFFECTIVE attacks, such that the post-processed samples had relatively higher UTMOS (> 2.3) compared to the original UTMOS and their EERs were also relatively higher ($> 2.9\%$) compared to other attacks (marked in red, e.g. *background noise* clapping).

We also identified some EFFECTIVE attacks, whose EERs were relatively higher ($> 2.9\%$) compared to the original EER, but which had lower UTMOS scores (<2.3) compared to the original UTMOS. Different from the VERY EFFECTIVE attacks, this suggests that the higher EERs could be influenced by the reduced quality resulting from the post-processing itself (marked in grey in Table III).

Further, we observed more fine-granular correlations between the UTMOS of the post-processed samples and the corresponding detection scores of each post-processing attack using PCC and SRCC. For most of the attacks that were identified as VERY EFFECTIVE or EFFECTIVE (marked in grey and red) there were minimal correlations between the UTMOS of the post-processed samples and the corresponding detection scores ($|correlation| < 0.2$, marked in bold). A similar behaviour can be seen when examining the correlation between the UTMOS of the original samples and detection scores. For LESS EFFECTIVE manipulations, however, the EERs appear to correlate to the UTMOS scores of the post-processed samples. Due to space constraints, these scores are omitted in this paper.

Moreover, we tried to find other common features shared by the successful attacks. For that, we observed the original spoofed samples that were very susceptible to the post-processing attacks. For the SSL-based detector, there were 9 samples and for the non-SSL based detector there were 22 samples that were successfully attacked with all attacks (total of 23 successful attacks). Upon closer observation of these samples, we found that most of them had been correctly identified as spoofs before modification (as original samples), except for 3 out of 9 samples in the SSL-based detector.

Hereupon, we observed the mean successful post-processing attack of each original samples that were initially not detected. We found that if the original samples were not detected, the post-processing attacks on these samples are also more likely to succeed. For the SSL-based model, the mean successful attack of the original samples that were initially not detected was 18.42. In contrast, if the original samples were detected,

TABLE III
EFFECTIVENESS OF POST-PROCESSING ATTACKS: CORRELATION UTMOS POST-PROCESSED SAMPLES VS. DETECTION SCORES - [RED: VERY EFFECTIVE MANIPULATIONS (HIGH EER + HIGH UTMOS), GREY: EFFECTIVE MANIPULATIONS (HIGH EER + LOW UTMOS), BOLD: LOW CORRELATION, HENCE $|correlation| < 0.2$]

Post-processing	non-SSL		SSL	
Method	PCC	SRCC	PCC	SRCC
gaussian noise	**-0.131**	**-0.185**	**-0.038**	**-0.077**
bkgd (clapping)	**-0.175**	**-0.175**	**-0.162**	-0.215
bkgd (church bells)	**-0.125**	**-0.12**	**-0.013**	**-0.049**
IR bedroom	**-0.07**	**-0.112**	**0.101**	**0.059**
IR car	**-0.087**	**-0.094**	**-0.082**	**-0.198**
roomsimulator	-0.328	-0.332	-0.207	-0.345
timestretch (0.8)	**-0.008**	**-0.05**	**-0.055**	**-0.155**
timestretch (0.6)	**0.01**	**0.001**	**-0.023**	**-0.045**
bandpassfilter	**-0.018**	**-0.005**	**0.044**	**0.048**
lowshelffilter	-0.254	-0.265	**-0.192**	-0.33
resample: 15.000	-0.246	-0.256	**-0.196**	-0.316
resample: 15.500	-0.261	-0.279	-0.204	-0.329
resample: 16.500	-0.263	-0.277	-0.215	-0.354
resample: 17.000	-0.254	-0.266	-0.209	-0.363
volume (gain: 0.5)	-0.225	-0.248	-0.198	-0.335
volume (gain: 0.1)	**-0.139**	**-0.159**	**-0.142**	-0.321
fading (linear)	**-0.17**	**-0.187**	**-0.187**	-0.302
fading (logarithmic)	-0.232	-0.249	-0.203	-0.334
fading (exponential)	**-0.122**	**-0.14**	**-0.172**	-0.287
Encoder vorbis (ogg)	-0.261	-0.279	-0.232	-0.373
Encoder opus (ogg)	-0.314	-0.332	**-0.197**	-0.335
mp3compression	**-0.168**	**-0.184**	**-0.139**	-0.251
AudioEffector (mp3)	-0.26	-0.277	-0.209	-0.341
mean $|correlation|$	0.179	0.194	0.149	0.251

TABLE IV
INCREASED ROBUSTNESS OF THE SSL/NON-SSL-BASED DETECTOR ON THE ITW TEST SET, IN EER (%) - [BOLD: ATTACKS USED FOR TRAINING, GREY: IMPROVED SCORES ALTHOUGH NOT USED FOR TRAINING, IR: IMPULSE RESPONSE, BN: BACKGROUND NOISE, BF: BANDPASSFILTER]

Post-processing	SSL			non-SSL		
Method	before	from scratch (IR+BN)	ft (IR+BF)	before	from scratch (BN)	ft (BN)
original data	0.73	1.09	0.86	2.52	3.76	7.64
gaussian noise	1.05	0.64	1.20	4.15	3.13	3.71
bkgd (clapping)	1.67	**0.63**	1.61	16.77	**2.56**	**0.62**
bkgd (church bells)	1.28	2.28	1.48	7.64	10.62	9.25
IR bedroom	4.37	**1.30**	**1.66**	6.45	5.97	15.30
IR car	1.14	1.01	0.82	6.44	4.43	18.25
roomsimulator	0.81	1.15	0.85	2.61	5.02	8.32
timestretch (0.8)	1.78	0.67	0.42	2.64	5.00	8.30
timestretch (0.6)	2.17	0.78	0.45	3.52	5.83	10.15
bandpassfilter	1.75	2.28	**1.16**	5.26	2.91	9.46
lowshelffilter	0.75	1.11	0.85	2.75	3.83	7.89
resample: 15.500	0.75	1.10	0.84	2.46	3.42	7.40
resample: 15.000	0.78	1.19	0.90	2.22	4.26	7.76
resample: 16.500	0.74	1.08	0.77	2.81	3.00	8.02
resample: 17.000	0.75	1.12	0.86	3.12	2.88	8.18
volume (gain: 0.5)	0.75	1.07	0.86	3.02	3.67	7.81
volume (gain: 0.1)	0.81	1.13	0.91	6.45	4.38	7.64
fading (linear)	0.64	1.01	0.68	3.39	3.39	8.46
fading (logarithmic)	0.67	0.93	0.67	2.80	3.45	8.07
fading (exponential)	0.68	1.03	0.67	4.14	3.49	8.78
Encoder vorbis (ogg)	0.65	0.75	0.68	2.74	2.71	7.50
Encoder opus (ogg)	0.74	0.91	0.82	2.12	3.52	6.76
mp3compression	0.83	0.81	0.86	3.53	1.46	8.33
AudioEffector; mp3	0.74	0.79	0.85	2.60	1.64	7.65
mean	1.13	1.08	0.91	4.26	3.93	8.39

the mean successful attack is only 0.47. For the non-SSL based model, the mean successful attack of the original samples that were initially not detected was 16.48 and for the original samples detected was only 1.54.

In summary, we observed the following:

- VERY EFFECTIVE and EFFECTIVE manipulations and the performances of our detectors are less correlated to/influenced by the quality of the samples (both original and post-processed samples).
- LESS EFFECTIVE manipulations are negatively correlated to the quality of the samples (both original and post-processed samples).
- If original samples were not detected as spoofs in the first place, it is more likely that they are also not detected after the attack.
- If all post-processing attacks were successful, it doesn't necessarily mean that the original samples were initially recognised as spoof.

VI. DETECTORS: IMPROVED ROBUSTNESS

After identifying the weaknesses of the detectors when confronted with post-processing attacks, we retrained the models. We tested fine-tuning and training the models from scratch using different augmentation settings. See Table IV for the results. Again, only some of the results are shown due to page restrictions. The results for fine-tuning were calculated when trained for 100 iterations because extending the training led to a decline in performance. For example, when fine-tuning on impulse response, the EER rose from 0.86% at 100 iterations to 0.97% at 200 iterations and further to 1.12% at 300 iterations.

Since the SSL-based detector performed worst against the *impulse response* (bedroom) attack across all three test sets, we first retrained the detector to this attack. For this, we trained the model from scratch, adding *impulse response* (IR) to the augmentation pipeline (p=0.2; p stands for the probability of applying this transform) and fine-tuned the model using *impulse response* (p=1.0). However, both, training from scratch and fine-tuning led to a slight decline in performance on the original dataset (EER training from scratch: 0.93%, EER fine-tuning: 0.86%; before: 0.73%). On the other hand, the model's performance on the *impulse response* (bedroom) attack and other post-processing attacks improved (original EER: 4.37%, EER training from scratch: 1.33%, EER fine-tuning: 1.21%). As the attack *background noise* (BN) resulted in the highest EER after training from scratch, we trained the model again, adding *background noise* (clapping) to the augmentation pipeline.

When fine-tuning the model with *impulse response* (bedroom), the results on this attack has improved. Then, the worst performance was calculated on the attack using *bandpassfilter* (EER 2.18%). Therefore, we fine-tuned the model again, on a combination of *impulse response* (bedroom) and *bandpassfilter* (BF; p=0.5). Eventually, the performance on both attacks improved (robustness to *impulse response* remains satisfactory with an EER of 1.66%). On the original dataset the EER was 0.86% which is a slight increase of 0.09%. For more insights, we calculated the mean EER over all attack types in the different training settings mentioned above and found that the final SSL-based model fine-tuned on *impulse response* (bedroom) and *bandpassfilter* had the overall best mean result of 0.91% which is an improvement by 0.22% (before: 1.13%).

For the non-SSL based detector, the worst results were initially calculated when applying *background noise* (clapping) and *impulse response* (bedroom). Therefore, we trained the model from scratch and used fine-tuning on these two attacks. Overall, the results when trained from scratch were better. The until now best and improved results on the ITW test set were achieved when training the model from scratch, adding *background noise* (clapping, p=0.2) to the augmentation pipeline. This improved the results on *background noise* (clapping) to an EER of 2.56% (ITW). However, this also increased the EER on the post-processing attack *background noise* (church bells) to an EER of 10.62% (previous EER: 7.64%). Overall, the best mean EER of 3.93% (before: EER 4.26%) was achieved when training from scratch with adding *background noise* to the augmentation pipeline (p=0.2).

VII. Effects of Re-training

In order to evaluate whether there are significant differences in behaviour for the final re-trained models (being the one with the best performance), we applied the same analysis on the correlation between the sound quality (UTMOS) and the new EERs as we did with the initial models. Different from the initial detectors, the correlations between the UTMOS scores of the post-processed samples and the EERs were -0.031 (before: -0.574) with PCC and 0.008 (before: -0.615) with SRCC for the SSL-based model. For the non-SSL based model the correlations were -0.431 (before: -0.082) with PCC and -0.435 (before: -0.470) with SRCC. Thus, after re-training, the SSL-based model became less correlated to the UTMOS scores of the post-processed samples while the non-SSL based model became more correlated.

The number of relatively effective or very effective attacks has dropped for both models: from 8 to 5 for the SSL-based and from 13 to 8 for the non-SSL based detector. Moreover, the SSL-based model also showed a decline in the mean and maximum of successful manipulation attacks for each original sample, regardless of whether the original samples were initially identified as spoofs or not. Similar improvements can be observed for the non-SSL based model. However, there is a decrease in performance on the original samples. For the SSL-based model, the total number of original samples that were not detected has increased from 86 to 102. For the non-SSL based model, the number has increased from 298 to 444. This indicates that the models have generalised across manipulation attacks, although they were minimally penalised in terms of overall EER for detecting original spoofed samples.

VIII. Conclusion

Our findings indicate that simple post-processing attacks impact detection performance. Especially adding background noise and impulse response proved to be effective in degrading detection performance of both detectors. The non-SSL-based detector is more susceptible to these attacks than the SSL-based detector. Both detectors can be improved by including the most harmful post-processing attacks in the training data. For the SSL-based detector, the fine-tuning and for the non-SSL based detector, the training from scratch improved the results the most. At the same time we found that a careful selection of dataset and retraining strategies are required while the best way of finding the best combinations still needs more research. In future work we plan to test the effects of combinations of post-processing attacks.

References

[1] Xuechen Liu, Xin Wang, Md Sahidullah, Jose Patino, Héctor Delgado, Tomi Kinnunen, Massimiliano Todisco, Junichi Yamagishi, Nicholas Evans, Andreas Nautsch, et al., "Asvspoof 2021: Towards spoofed and deepfake speech detection in the wild," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[2] Xin Wang, Hector Delgado, Hemlata Tak, Jee weon Jung, Hye jin Shim, Massimiliano Todisco, Ivan Kukanov, Xuechen Liu, Md Sahidullah, Tomi Kinnunen, Nicholas Evans, Kong Aik Lee, and Junichi Yamagishi, "Asvspoof 5: Crowdsourced speech data, deepfakes, and adversarial attacks at scale," *Proc. The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)*, 2024.

[3] Mirco Ravanelli and Yoshua Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE spoken language technology workshop (SLT)*. IEEE, 2018, pp. 1021–1028.

[4] Nicolas Michael Müller, Pavel Czempin, Franziska Dieckmann, Adam Froghyar, and Konstantin Böttinger, "Does audio deepfake detection generalize?," *Interspeech*, 2022.

[5] Xin Wang and Junichi Yamagishi, "Investigating Self-Supervised Front Ends for Speech Spoofing Countermeasures," in *Proc. Odyssey*, 2022, pp. 100–106.

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

[7] Karla Schäfer, Matthias Neu, and Jeong-Eun Choi, "Robust audio deepfake detection: Exploring front-/back-end combinations and data augmentation strategies for the asvspoof5 challenge," in *The Automatic Speaker Verification Spoofing Countermeasures Workshop 2024*, 2024.

[8] Hemlata Tak, Massimiliano Todisco, Xin Wang, Jee-weon Jung, Junichi Yamagishi, and Nicholas Evans, "Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation," in *The Speaker and Language Recognition Workshop*, 2022.

[9] Xin Wang, Junichi Yamagishi, Massimiliano Todisco, Héctor Delgado, Andreas Nautsch, Nicholas Evans, Md Sahidullah, Ville Vestman, Tomi Kinnunen, Kong Aik Lee, et al., "Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech," *Computer Speech & Language*, vol. 64, pp. 101114, 2020.

[10] Joel Frank and Lea Schönherr, "Wavefake: A data set to facilitate audio deepfake detection," *35th Conference on Neural Information Processing System*, 2021.

[11] Keith Ito and Linda Johnson, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[12] Chengzhe Sun, Shan Jia, Shuwei Hou, and Siwei Lyu, "Ai-synthesized voice detection using neural vocoder artifacts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023, pp. 904–912.

[13] Ricardo Reimao and Vassilios Tzerpos, "For: A dataset for synthetic speech detection," in *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*. IEEE, 2019, pp. 1–10.

[14] Hemlata Tak, Madhu Kamble, Jose Patino, Massimiliano Todisco, and Nicholas Evans, "Rawboost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6382–6386.

[15] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," *ArXiv*, vol. abs/2204.02152, 2022.