Speech Replay Detection with *x*-Vector Attack Embeddings and Spectral Features

Jennifer Williams and Joanna Rownicka

The Centre for Speech Technology Research, University of Edinburgh, United Kingdom

j.williams@ed.ac.uk and j.m.rownicka@sms.ed.ac.uk

Abstract

We present our system submission to the ASVspoof 2019 Challenge Physical Access (PA) task. The objective for this challenge was to develop a countermeasure that identifies speech audio as either bona fide or intercepted and replayed. The target prediction was a value indicating that a speech segment was bona fide (positive values) or "spoofed" (negative values). Our system used convolutional neural networks (CNNs) and a representation of the speech audio that combined *x*-vector attack embeddings with signal processing features. The *x*-vector attack embeddings were created from mel-frequency cepstral coefficients (MFCCs) using a time-delay neural network (TDNN). These embeddings jointly modeled 27 different environments and 9 types of attacks from the labeled data. We also used sub-band spectral centroid magnitude coefficients (SCMCs) as features. We included an additive Gaussian noise layer during training as a way to augment the data to make our system more robust to previously unseen attack examples. We report system performance using the tandem detection cost function (tDCF) and equal error rate (EER). Our approach performed better that both of the challenge baselines. Our technique suggests that our *x*-vector attack embeddings can help regularize the CNN predictions even when environments or attacks are more challenging.
Index Terms: automatic speaker verification, spoofing countermeasures, speech replay detection

1. Introduction

Detecting replayed speech is a notoriously difficult task in speech processing. In particular, an adversary might obtain a snippet of recorded speech wherein the human target has used their voice authentically, such as when speaking a passphrase. When an automatic speaker verification (ASV) system is presented with fake speech intending to impersonate a live talker then this is commonly referred to as either "spoofing", a *replay attack* or a *presentation attack*. Building upon similar challenges from earlier years [1, 2, 3], detecting replay attacks was the basis for the Physical Access (PA) task of the ASVspoof 2019 Challenge [4]. The aim of our work was to detect whether or not a bona fide live speech recording had been intercepted and subsequently replayed back as non-live speech to an ASV system. There are numerous variables to be modeled including elements of how the attack was conducted as well as the type of ASV system being attacked. Our approach captured these conditions using a convolutional neural network (CNN) architecture, similar to the top system from the earlier ASVspoof 2017 replay detection challenge [3]. We explored several types of signal features such as those from [2] and our final submission was based on our own specially-trained *x*-vector [5] embeddings combined with signal features.

While there are many different kinds of speech spoofing attacks, including voice conversion and text-to-speech [6, 7, 8, 9], we have focused on replay attacks. Some work suggests that speech replay clues are found in the time and frequency domains [10]. Other work has explored energy-based features [11, 12], attention-based adaptive filters [13], and convolutional neural networks (CNNs) [14, 15]. It is particularly challenging to model different types of acoustic environments, playback devices, and recording devices [16]. Recent work has also shown that high-frequency sub-bands in the acoustic signal contain evidence of replay. For example, Inverted Mel Frequency Cepstral Coefficients (IMFCCs) consistently discriminate speech replay across several frequency sub-bands [17]. IMFCCs come from inverting filters in the frequency domain to capture more detail from higher frequencies. Other recent work has shown that Sub-band Spectral Centroid Magnitude Coefficients (SCMCs) are the best and most consistent signal feature [2] across experiments, while the Constant-Q Cepstral Coefficient (CQCC) features are also promising [18].

This paper makes three main contributions. First, we introduce novel *x*-vector attack embeddings capturing the recording conditions of an utterance (attack and environment variables). Second, we analyze how well the *x*-vector embeddings model factors of variation from different recording conditions. Finally, we demonstrate that the combination of signal features and *x*-vector embeddings out-performs all baselines for both metrics on the ASVspoof 2019 development and evaluation datasets.

2. Feature Development

2.1. Speech Signal Features

Following from the features analysis for replay attack detection that was presented in [2] for the ASVspoof 2017 challenge, we extracted the following features: Mel Frequency Cepstral Coefficients (MFCCs), Inverted Mel Frequency Cepstral Coefficients (IMFCCs), Rectangular Filter Cepstral Coefficients (RFCCs), Linear Frequency Cepstral Coefficients (LFCCs), Sub-band Spectral Centroid Magnitude Coefficients (SCMCs), and Constant Q Cepstral Coefficients (CQCCs) [1]. A description of the features is provided in Table 1. We used static features because preliminary experiments indicated that the dynamic features were not as useful in the spoofing task, especially the second-order features.

We used the IDIAP *Bob.ap* signal processing library to extract this set of signal processing features from speech audio files [19, 20]. In the case of CQCCs, we used the code provided by the challenge organizers and also described in [1]. For each audio file, the feature extractor output is an NxM*-dimensional* matrix with N as the number of coefficients and M as the duration of the file in frames.

Each audio file in the ASVspoof 2019 dataset was of a different length. We pre-processed the extracted features to handle this length variability and to create same-sized feature vectors as input to our CNN classifier. We used a down-sampling

Features	Coeff. num. (N)	$f_{min} - f_{max}$ (Hz)
MFCC	70	300-8000
IMFCC	60	200-8000
RFCC	30	200-8000
LFCC	70	100-7800
SCMC	40	100-8000
CQCC	50	15.62-8000

Table 1: *Description of signal features extracted from the raw speech audio. Note CQCC used a specific f_{min} from [1]*

technique in the feature space. This means that for a given coefficient in a given audio file, we down-sampled the number of frames to a constant. The technique preserved the original per-coefficient distributions in a file while also setting the number of frames to a constant. For re-sampling we used the Fast Fourier Transform (FFT) so that the spacing between the original frames, $s = \delta x$, then became: $s = \delta x * M/M'$. By doing this, we set the number of down-sampled frames M' to a constant where $M' = 10$. This effectively shortened the audio file duration to 10 frames while preserving the mean and standard deviation of each coefficient in a given file [21].

After this, our signal processing features were represented as a $Nx10$-*dim* matrix. We selected $M' = 10$ frames based on two motivations: 1) to reduce the overall size and overhead of the dataset for later processing, and 2) to allow our countermeasures to operate on very short audio examples. We then stacked the coefficients on a per-frame basis, in an effort to preserve some temporal nature of the original signal. Finally, each of the signal feature values were re-scaled to be between -1 and $+1$ using per-feature max values from the training set, applied later to development and evaluation sets. We re-scaled the values in order to align with our selection of activation function for the regression task described in Section 3, which outputs a value between -1 (spoofed) and $+1$ (authentic) .

2.2. *X*-vector Embedding Creation

In this work we also used *x*-vectors [5] as auxiliary features for the CNN model described in Sec. 3. Our aim was to extract meaningful fixed-size utterance-level vectors representing the factors of variation, namely environment and attack conditions, in the spoofing task. We used our extracted representations to account for these factors in the final spoofing detection task. This effort was to improve the system robustness to unseen conditions by leveraging information about the environment and attack classes from the labels provided for each training example.

The Kaldi Toolkit [22] was used to extract *x*-vectors representing a joint environment+attack class (which we refer to as *env+attack*). The input features for the *x*-vector extractor were 40-*dim* MFCCs, with 80 filters in a filter bank. The *x*-vector extractor was a time-delay neural network (TDNN) with the same architecture as in [5], i.e. seven layers with batch normalisation and ReLU activations. The *x*-vectors were extracted from the sixth layer before the nonlinearity. Differently to the model in [5] though, ours was not trained to classify speakers. The extractor was trained to differentiate between classes jointly representing types of acoustic environments and types of attacks.

The joint *env+attack* classes were created by combining each category label of variation for the simulated acoustic environments and attack types, i.e. room size, T60 reverberation time, talker-to-ASV distance, attacker-to-talker distance, and replay device quality. From 10 attack type configurations, and 27 acoustic environment configurations (9 attacks plus authentic speech), we created 270 env+attack classes. Training an x-vector extractor to differentiate between *env+attack* classes enabled us to learn fixed-size representations, capturing both the type of attack and the type of acoustic environment. The classification accuracy of our joint *env+attack* *x*-vectors was around 85% for 270 unique classes on a held-out validation set (10% of training), hence these representations were meaningful.

After the *x*-vector extraction, we reduced the dimensionality of our *x*-vectors from 512-*dim* to 10-*dim* using Linear Discriminant Analysis (LDA). The LDA model also used *env+attack* classes for training. We selected 10-*dim* based on the EER from the development set. For 59,400 trials with non-target proportion of 50%, the EER in the *env+attack* verification task was 23.96% with the LDA backend.

2.3. *X*-vector Embedding Analysis

In this section we show an analysis of how the *x*-vector embeddings could differentiate between different types of attacks and different types of environments. There are 27 environment classes and 10 attack classes. It was easier to show the analysis for environments and attacks separately, compared to modeling all 270 classes for the jointly trained *env+attack* embeddings, which are the ones ultimately used in our system.

Environment classification was an easier task (accuracy 86% on the validation set) than attack type classification (accuracy 60% on the validation set), even though the number of classes for classifying attacks was smaller than for classifying environment. We hypothesize that this may be caused partly by data imbalance in the case of attack recognition, compared to the evenly distributed examples for the environment classes.

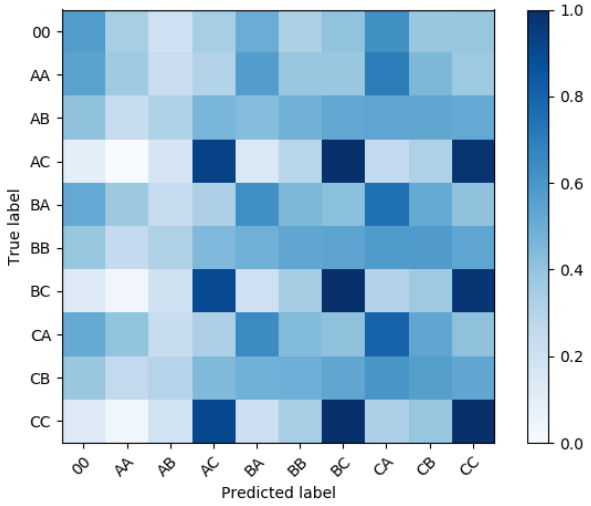

Figure 1: *Confusion matrix for attack class predictions. The scoring is based on the per attack mean development x-vectors against per attack mean training x-vectors. Class 00 is the bona fide class. For other labels, letters in the first position corresponds to the attacker-to-talker distance (A - lowest, C - highest), letters in the second position corresponds to the replay device quality (A - the best, C - the worst).*

To further investigate what the extracted *x*-vectors were capturing, we analyzed the accuracy scores from predicting attack and environment. Figure 1 shows a confusion matrix for

the scores for mean x-vectors per attack. First of all, it can be observed that the replay device quality is well captured by the *x*-vectors (labels with the same letter in the second position). However, the attack-to-talker distance does not seem to be modeled well with the attack x-vectors (e.g. scores for classes AC, BC, and CC are very close to each other, but different than for any other classes). The most evenly distributed scores can be observed for the medium replay device quality classes (AB, BB, CB). Our *x*-vector embeddings can detect when the replay device is very poor. However, if the replay device quality is near perfect, it is much more difficult to develop the spoofing countermeasures.

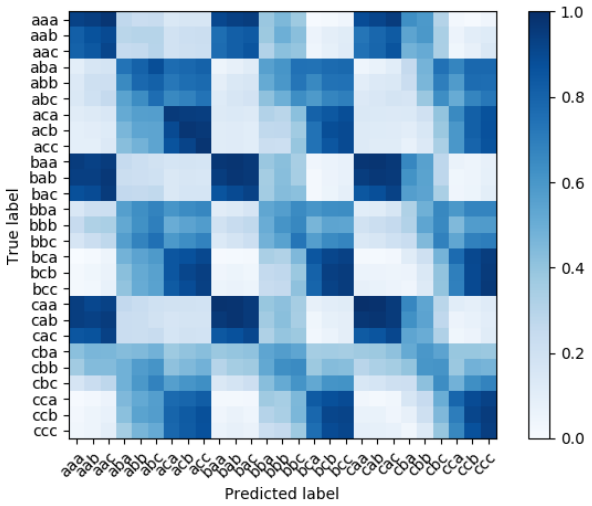

Figure 2: *Confusion matrix for environment class predictions using mean development x-vectors against mean training x-vectors. Letters in the first position of the label ID correspond to the room size (a - smallest, c - biggest). Letters in the second position correspond to T60 reverberation time (a - shortest, c - longest). Letters in the third position correspond to the talker-to-ASV distance (a - shortest, c - longest).*

Figure 2 shows the confusion matrix for how well the mean *x*-vectors discriminate environment classes. Again, the talker-to-ASV distance and the room size do not seem to be very well captured (first and third letter position in labels). However, the reverberation time (especially short and long) discriminates classes well. Both spoofed and bona fide recordings were simulated in a variety of environments. So the idea was to extract an embedding that would help to normalize out the effects of recording in different environmental conditions, to be able to generalize well to unseen conditions at test time.

We hypothesize that even though the *x*-vectors do not differentiate very well between every attack and every environment class, they do differentiate between some of them. Furthermore, *x*-vector embeddings for similar acoustic conditions are close to each other in the *x*-vector space. Therefore, these can be useful representations complimenting our signal processing features.

2.4. Feature Combination

In our experiments, we explored several combinations of our features while evaluating on the development set. For the first case, we evaluated signal processing features individually. Next, we evaluated *x*-vector embeddings individually. Finally, we combined signal processing features with *x*-vector embeddings. Before concatenating the LDA *x*-vectors to the signal processing features, the LDA *x*-vectors were scaled with $c = 0.1$ constant. We empirically found the scaling to have a good effect on the final EER and tDCF metrics. Scaling *x*-vectors before concatenation is conceptually similar to applying a fixed LDA-like transform in Kaldi (usually used when *i*-vectors are concatenated to the input features for normalisation in ASR), which is scaling down the dimensions that are "non-informative". This has the effect of encouraging stochastic gradient descent (SGD) to ignore non-informative values. Scaled and transformed *env+attack x*-vectors are denoted as *xEAs* in our paper. They were concatenated to the signal processing features at the input of the CNN model. We hypothesize that it enabled us to normalize out some factors of variation, subsequently enabling the CNN model to learn more robust representations for the final countermeasure task.

Our system submission to the ASVspoof 2019 Challenge was based on two features: SCMC signal features concatenated with the *xEAs* vectors (scaled and transformed as described above). For the submission, our dataset consisted of 54,000 training instances and 29,700 development instances. Our training data was therefore of size $(54000, 410)$. These dimensions were based on 40 SCMC coefficients by 10 frames per coefficient, plus the additional 10-dimension *x*-vectors. The two features were combined via concatenation. In each of the training and development sets, we found 5,400 instances had been labeled as bona fide while the remaining had been labeled as spoof, thus the dataset was very imbalanced for the two targets.

3. System Architecture

Our system[1] was implemented with the Keras library [23] with TensorFlow backend [24]. We designed our system to perform regression. To do this we converted categorical target labels to numerical values as follows: "spoof" became -1 and "bona fide" became $+1$. Since we used the hyperbolic tangent activation function, the output of our system was therefore a value between -1 and $+1$. The challenge evaluation plan called for values with greater negative magnitude to correspond to the "spoof" class while values with greater positive value to correspond to the "bona fide" class. Thus we intentionally set up the problem as a regression task. Our system output could be interpreted to represent the degree of authenticity of the audio especially considering that the countermeasures output by our system were later evaluated in tandem with an ASV system.

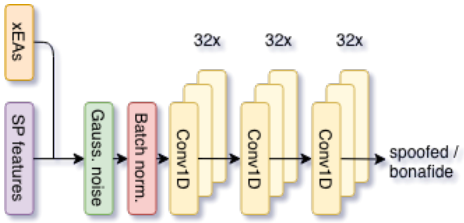

Figure 3: *Overview of our machine learning system architecture. Scaled LDA x-vectors (xEAs) were concatenated to the signal processing features (SCMCs). The DNN was a 3-layer convolutional neural network.*

[1] https://github.com/rhoposit/ASVchallenge2019.git

3.1. Convolutional Neural Networks (CNN)

Figure 3 shows an overview of our system architecture. The first layer of our CNN was an additive Gaussian noise layer [25, 26, 27]. We used this as a form of data augmentation to help the model generalize to unseen conditions. We determined the placement of this layer experimentally and also tried different values for the standard deviation of the noise distribution, finally deciding on $n_{std} = 0.001$. The next layer was a batch normalization layer. The CNN consisted of 3 Conv1D layers. The kernel size was set to 3 and we used 32 convolutional filters. Each Conv1D layer included a L2 regularizer [28]. Each Conv1d layer was followed by a max pooling layer with pool size and stride set to 2. Finally, we used a fully connected layer with a single output and the hypoerbolic tangent activation function (*tanh*) [29]. The activation function had the effect of restricting the output between -1 and 1.

3.2. System Training

We trained our model using a 10% subset of the training set as a validation set. During training we swept several different parameters. We explored values for standard deviation in the Gaussian additive noise layer, in the range of: [0.000001, 0.00001, 0.00005, 0.0001, 0.001]. We also explored incremental values for L2 regularization between [0.00001, 0.001]. For training, we measured loss using mean-squared-error (MSE). We used early-stopping [30], and monitored validation loss with change $delta = 0$ and patience $p = 5$ epochs. We used the Adam optimizer [31] with learning rate $lr = 0.001$ and remaining parameters set as default.

4. Results and Discussion

Our system was evaluated using two related metrics. The primary metric was the tandem detection cost function (tDCF) computed in conjunction with an ASV system that was kept hidden from participants [32]. This allowed the organizers to vary the ASV system to evaluate robustness of systems. The secondary metric was equal-error-rate (EER) based on the quality of the countermeasure alone to predict bona fide versus spoofed. We selected our best system for submission to the challenge based on performance with the tDCF metric.

In Table 2 we report our system performance on the development set using the signal features, the *x*-vector features, and our top features combined. We also report the official results for our submission on the evaluation set. For reference, we provide the evaluation performance for both baselines - which used a Gaussian Mixture Model (GMM) and signal features. They were: LFCC-GMM [33] and CQCC-GMM [1]. Our system performed better than the baselines for both metrics on development and evaluation sets (to 3-significant digits).

While the *x*-vectors alone did not distinguish well between spoofed and bona fide speech, we did find an improvement when the *x*-vectors were combined with signal features. Specifically, we found our best feature combination to be SCMC features with the scaled LDA *x*-vectors, *xEAs*, that jointly modeled environment and attack variations. The SCMC feature captures the magnitude of energy a sub-band, which can effectively distinguish two signals even if they share the same average energy. The SCMC feature was also one of the best-performing features from the analysis in [2] for the ASVspoof 2017 challenge, though was based on different data. It has been recognized as a stable feature across experimental conditions. Further experiments on the use of the Gaussian noise layer indicated that using the noise layer (shown by default in Table 2) always performed better than without it (shown as SCMC+xEAs-N in Table 2).

		Development		Evaluation	
		t-DCF	EER (%)	t-DCF	EER (%)
Signal Features	**MFCC**	**0.204**	**8.35**	-	-
	IMFCC	**0.199**	**7.98**	-	-
	RFCC	0.210	8.58	-	-
	SCMC	**0.209**	**8.47**	-	-
	LFCC	0.229	8.90	-	-
	CQCC	0.275	10.9	-	-
x-vectors	**xA**	**0.814**	**31.5**	-	-
	xE	0.971	41.5	-	-
	xEA	**0.820**	**31.6**	-	-
	xAs	**0.815**	**31.4**	-	-
	xEs	0.970	41.7	-	-
	xEAs	**0.820**	**31.9**	-	-
Combo	**SCMC+xEAs**	**0.194**	**7.74**	**0.235**	**9.15**
	SCMC+xEAs-N	0.225	9.16	-	-
	IMFCC+xEs	0.197	7.47	-	-
	MFCC+IMFCC	0.206	7.96	-	-
	LFCC-GMM	0.255	11.9	0.301	13.5
	CQCC-GMM	0.195	9.87	0.245	11.0

Table 2: *Our system using different features on the development set, and the evaluation set for our challenge submission. Two baselines from the organizers are included last, for reference.*

In our earlier analysis, we had found that differences in the replay device quality was captured by the *x*-vectors. When the replay device quality is very good, or perfect, then it is much more difficult to develop spoofing countermeasures. While not reported in this paper, the official detailed performance results from the ASVspoof 2019 challenge organizers also indicate that certain attack types are much more difficult, specifically with the high-quality replay device. That is an important finding for ASV research and our analysis in this paper also supports it.

A potential limitation of our approach is due to our use of the *tanh* activation function for the CNN regression output. Typically, this activation forces output near the boundaries of -1 and +1, making it more difficult to obtain output scores near the centerline. An interesting analysis of our system might include how the output values are situated within the range of -1 and +1 for various attack types and environmental conditions.

As future work, we would also like to experiment with different input features for the *x*-vector extraction. In this paper, we used frame-level 40-*dim* MFCC features, without restricting the frequency range. We encourage future work to look more closely at the best frequency range and the best type of frame-level features to capture differences in the acoustic conditions of an utterance with these *x*-vector embeddings. Different normalization techniques could also be investigated.

5. Acknowledgements

This work was supported in part by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh. It was also supported by a PhD studentship from the DataLab Innovation Centre, Ericsson Media Services, and Quorate Technology. The authors thank: Erfan Loweimi, Ondrej Klejch, and Joachim Fainberg (CSTR), and Michael Camilleri (IANC) for their helpful discussions.

6. References

[1] M. Todisco, H. Delgado, and N. Evans, "Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification," *Computer Speech & Language*, vol. 45, pp. 516–535, 2017.

[2] R. Font and M. J. Cano, "Experimental Analysis of Features for Replay Attack Detection - Results on the ASVspoof 2017 Challenge," in *Interspeech*, 2017, pp. 7–11.

[3] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio Replay Attack Detection with Deep Learning Frameworks," *Interspeech*, pp. 82–86, 2017.

[4] ASVspoof 2019: Automatic Speaker Verification Spoofing and Countermeasures Challenge. [Online]. Available: http://www.asvspoof.org/

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN Embeddings for Speaker Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[6] A. Janicki, F. Alegre, and N. Evans, "An Assessment of Automatic Speaker Verification Vulnerabilities to Replay Spoofing Attacks," *Security and Communication Networks*, vol. 9, no. 15, pp. 3030–3044, 2016.

[7] S. Shiota, F. Villavicencio, J. Yamagishi, N. Ono, I. Echizen, and T. Matsui, "Voice Liveness Detection Algorithms Based on Pop Noise Caused by Human Breath for Automatic Speaker Verification," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[8] ——, "Voice Liveness Detection for Speaker Verification Based on a Tandem SingleDouble Channel Pop Noise Detector," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2016, pp. 259–263.

[9] L. Blue, L. Vargas, and P. Traynor, "Hello, Is It Me You're Looking For?: Differentiating Between Human and Electronic Speakers for Voice Interface Security," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 2018, pp. 123–133.

[10] C.-I. Lai, A. Abad, K. Richmond, J. Yamagishi, N. Dehak, and S. King, "Attentive Filtering Networks for Audio Replay Attack Detection," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6316–6320.

[11] M. Kamble, H. Tak, and H. Patil, "Effectiveness of Speech Demodulation-Based Features for Replay Detection," in *Proceedings of Interspeech 2018*, 2018, pp. 641–645. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1675

[12] M. R. Kamble and H. A. Patil, "Analysis of Reverberation via Teager Energy Features for Replay Spoof Speech Detection," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2607–2611.

[13] M. Liu, L. Wang, J. Dang, S. Nakagawa, H. Guan, and X. Li, "Replay Attack Detection Using Magnitude and Phase Information with Attention-Based Adaptive Filters," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6201–6205.

[14] B. Chettri, S. Mishra, B. L. Sturm, and E. Benetos, "A Study On Convolutional Neural Network Based End-To-End Replay Anti-Spoofing," *arXiv preprint arXiv:1805.09164*, 2018.

[15] I. Himawan, S. Madikeri, P. Motlicek, M. Cernak, S. Sridharan, and C. Fookes, "Voice Presentation Attack Detection Using Convolutional Neural Networks," in *Handbook of Biometric Anti-Spoofing*. Springer, 2019, pp. 391–415.

[16] B. Chettri, B. L. Sturm, and E. Benetos, "Analysing Replau Spoofing Countermeasure Performance Under Varied Conditions," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2018, pp. 1–6.

[17] M. Witkowski, S. Kacprzak, P. Zelasko, and K. Kowalczyk, "Audio Replay Attack Detection Using High-Frequency Features," *Interspeech*, pp. 82–86, 2017.

[18] P. Nagarsheth, E. Khoury, K. Patil, and M. Garland, "Replay Attack Detection Using DNN for Channel Discrimination," *Interspeech*, pp. 97–101, 2017.

[19] A. Anjos, L. E. Shafey, R. Wallace, M. Günther, C. McCool, and S. Marcel, "Bob: A Free Signal Processing and Machine Learning Toolbox for Researchers," in *20th ACM Conference on Multimedia Systems (ACMMM), Nara, Japan*, Oct. 2012.

[20] A. Anjos, M. Günther, T. de Freitas Pereira, P. Korshunov, A. Mohammadi, and S. Marcel, "Continuously Reproducing Toolchains in Pattern Recognition and Machine Learning Experiments," in *International Conference on Machine Learning (ICML)*, Aug. 2017.

[21] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–, [Online; accessed ¡today¿]. [Online]. Available: http://www.scipy.org/

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[23] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[24] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A System for Large-Scale Machine Learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265–283.

[25] S. Dutta, B. Tripp, and G. W. Taylor, "Convolutional Neural Networks Regularized by Correlated Noise," in *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE, 2018, pp. 375–382.

[26] G. An, "The Effects of Adding Noise During Backpropagation Training on a Generalization Performance," *Neural computation*, vol. 8, no. 3, pp. 643–674, 1996.

[27] C. M. Bishop, "Training With Noise is Equivalent to Tikhonov Regularization," *Neural Computation*, vol. 7, no. 1, pp. 108–116, 1995.

[28] A. Y. Ng, "Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance," in *Proceedings of the Twenty-First International Conference on Machine Learning*. ACM, 2004, p. 78.

[29] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[30] L. Prechelt, "Early Stopping - But When?" in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[31] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[32] 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," *arXiv preprint arXiv:1804.09618*, 2018.

[33] M. Sahidullah, T. Kinnunen, and C. Hanilçi, "A Comparison of Features for Synthetic Speech Detection," *International Speech Communication Association (ISCA)*, 2015.