The DKU Replay Detection System for the ASVspoof 2019 Challenge: On Data Augmentation, Feature Representation, Classification, and Fusion

Weicheng Cai[1,2], *Haiwei Wu*[1,2], *Danwei Cai*[1], *and Ming Li*[1]

[1]Data Science Research Center, Duke Kunshan University, Kunshan, China
[2]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China

ming.li369@dukekunshan.edu.cn

Abstract

This paper describes our DKU replay detection system for the ASVspoof 2019 challenge. The goal is to develop spoofing countermeasure for automatic speaker recognition in physical access scenario. We leverage the countermeasure system pipeline from four aspects, including the data augmentation, feature representation, classification, and fusion. First, we introduce an utterance-level deep learning framework for anti-spoofing. It receives the variable-length feature sequence and outputs the utterance-level scores directly. Based on the framework, we try out various kinds of input feature representations extracted from either the magnitude spectrum or phase spectrum. Besides, we also perform the data augmentation strategy by applying the speed perturbation on the raw waveform. Our best single system employs a residual neural network trained by the speed-perturbed group delay gram. It achieves EER of 1.04% on the development set, as well as EER of 1.08% on the evaluation set. Finally, using the simple average score from several single systems can further improve the performance. EER of 0.24% on the development set and 0.66% on the evaluation set is obtained for our primary system.

Index Terms: anti-spoofing, replay detection, deep learning, data augmentation, speed perturbation

1. Introduction

Automatic speaker verification (ASV) refers to automatically accept or reject a claimed identity by analyzing speech utterances, and nowadays it is widely used in real-world biometric authentication applications [1–3]. Recently, a growing number of studies have confirmed the severe vulnerability of state-of-the-art ASV systems under a diverse range of intentional fraudulent attacks on various databases [4–6]. The initiative of the series ASVspoof challenge aims to promote the development of spoofing countermeasure studies [7]. The task in previous ASVspoof 2015 challenge was to discriminate genuine human speech from speech produced using text-to-speech and voice conversion attacks [8]. The ASVspoof 2017 challenge is to assess audio replay spoof attack detection “in the wild”, and the spoofing recordings are created from real uncontrolled replay setup [9–11].

The ASVspoof 2019 challenge extends the previous challenge in several directions [12]. In this paper, we only focus on the replay detection sub-task in the physical access scenario. A controlled setup in the form of replay attacks simulated using a range of real replay devices and carefully controlled acoustic conditions are adopted. The goal is to develop countermeasure systems that can distinguish between bona fide and the spoofed speech by replay.

Recently, the Gaussian Mixture Model (GMM) classifier trained with constant Q cepstral coefficient (CQCC) feature has been the benchmark for various anti-spoofing tasks [13,14]. The CQCC feature is a perceptually-inspired time-frequency analysis extracted from a constant-Q transform (CQT) [15, 16]. The GMM is an unsupervised generative model and is widely used to depict the probability distribution of audio features. It assumes that all frames of features are independent and identically distributed, and features are grouped to estimate the parameters of GMM in the training stage. Given the GMM classifier, the final utterance-level scores are derived by averaging the frame-level log-likelihood altogether.

Although the CQCC-GMM official baseline is an effective spoofing countermeasure, there remains much potential to improve. Previous work in [17–19] have investigated the efficiency of deep learning approach compared to the GMM classifier. Various kinds of feature representation other than CQCC, such as short-time Fourier transform (STFT) gram [18], group delay gram (GD gram) [19] are also explored and show superior performance. Besides, Cai *et al.* have also demonstrated that suitable data augmentation (DA) can also significantly improve the replay detection system performance [20].

Taking the DKU system for ASVspoof 2019 challenge as the pivot, we aim to explore the countermeasure system in physical access scenario from the following four aspects, including the data augmentation, feature representation, classification, and fusion. First, we introduce an utterance-level deep learning framework for anti-spoofing. Considering the variable-length nature of speech, the deep neural network (DNN) receives a feature sequence of arbitration duration and outputs the utterance-level posteriors directly. Based on the framework, we try out various kinds of input feature representations from either magnitude spectrum or phase spectrum. They include the CQCC, linear frequency cepstral coefficients (LFCC), inverted Mel-frequency cepstral coefficients (IMFCC), STFT gram, and GD gram. Besides, we also perform a simple yet effective DA strategy by applying the speed perturbation on the raw waveform to increase the quantity of training data.

2. Methods

2.1. Utterance-level DNN framework

All of our systems are built upon a unified utterance-level DNN framework. We first use the approach for the task of speaker and

This research was funded in part by the National Natural Science Foundation of China (61773413), Natural Science Foundation of Guangzhou City (201707010363), Six talent peaks project in Jiangsu Province (JY-074), Science and Technology Program of Guangzhou City (201903010040), and Huawei. We thank Weixiang Hu, Yu Lu, Zexin Liu and Lei Miao from Huawei Digital Technologies Co., Ltd, China who provided insight and expertise that greatly assisted this research.

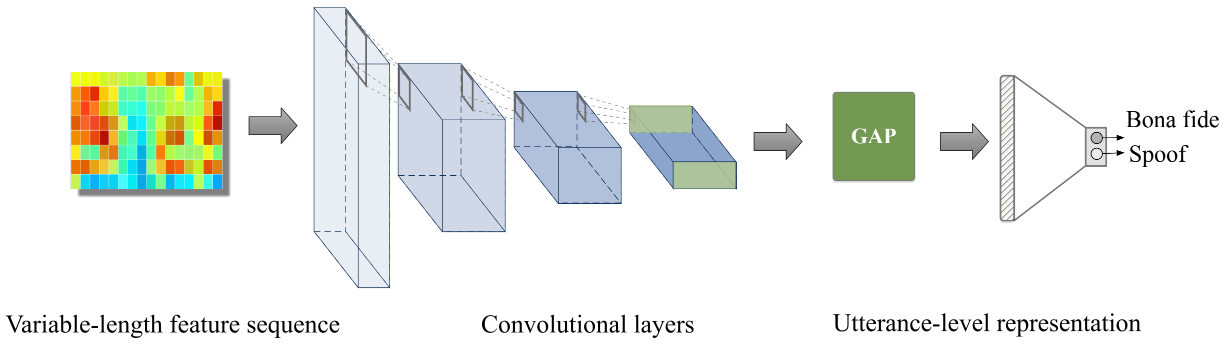
Bona fide
Spoof
GAP
Variable-length feature sequence
Convolutional layers
Utterance-level representation

Figure 1: *Utterance-level DNN framework for anti-spoofing. It accepts input data sequence with variable length, and produces an utterance-level result directly from the output of the DNN.*

language recognition in previous works [21–23]. As demonstrated in Fig. 1, the DNN framework accepts variable-length feature sequence and produces an utterance-level result from the output unit directly.

The network structure here is somewhat similar to that one in [19]. However, there exist two main differences: (i) the input feature sequence is first either truncated or padded to a fixed length along the time axis, and then further resized to a 512×256 "image" before feeding into the DNN in [19]. In contrast, we employ a variable-length training strategy, and in the testing stage, the raw full-length feature sequences are fed into our DNN directly. (ii) the training procedure in [19] contains the complicated two-stage flow. Both of the two-stage networks rely on a pre-trained model from the large-scale Imagenet [24] dataset. In contrast, we train a single DNN from scratch only using the ASVspoof 2019 training set.

We first transform the raw waveform into a frame-level feature sequence based on hand-crafted filters. The output sequence of feature extraction is a matrix of size $D \times L$, where D is the feature dimension along the frequency axis, and L denotes the frame length along the time axis.

As depicted in Fig. 1, we use a deep convolutional neural network (CNN) to further transform the raw feature into a high-level abstract representation. For a given feature sequence of size $D \times L$, typically, the CNN learned descriptions are a three-dimensional tensor block of shape $C \times H \times W$, where C denotes the number of channels, H and W denotes the height and width of the feature maps. Generally, H and W are much smaller than the original D and L, since we have many downsample operations within the CNN structure.

The CNN acts as a local pattern extractor, and the learned representation by CNN is still with temporal order. The remaining question is: how to aggregate the whole sequence together over time? Concerning about that, we adopt a global average pooling (GAP) layer on the top of CNN [25]. Given CNN learned feature maps $\boldsymbol{F} \in \mathbb{R}^{C \times H \times W}$, the GAP layer accumulates mean statistics along with the time–frequency axis, and the corresponding output is defined as:

$$v_i = \frac{1}{H \times W} \times \sum_{j=1}^{j=H} \sum_{k=1}^{k=W} \boldsymbol{F}_{i,j,k} \tag{1}$$

Therefore, we get fixed-dimensional utterance-level representation $\boldsymbol{V} = [v_1, v_2, \cdots, v_C]$ from the output of GAP.

We further process the utterance-level representation through a fully-connected feed-forward network and build an output layer on top. The two units in the output layer are represented as bona fide and spoof categories. We can optimize the whole countermeasure system in an end-to-end manner with a cross-entropy loss. The final utterance-level score can be directly fetched from the DNN output.

2.2. Feature representation

In this section, we investigate different input feature representations upon the introduced DNN framework in Section 2.1.

2.2.1. CQCC

The CQCC feature is obtained by perceptually-aware CQT coupled with traditional cepstral analysis. It is reported to be sensitive to the general form of spoofing attack, and yields superior performance among various kinds of features [16]. More details of CQCC can be found in [14].

2.2.2. LFCC

The official baseline systems adopt the CQCC feature as well as the LFCC feature.

LFCC is a kind of cepstral features based on triangle filterbank similar to the widely-used Mel-frequency cepstral coefficients (MFCC). It is extracted the same way as MFCC, but the filters are in the same triangular shape rather than on mel scale. Therefore, LFCC might have better resolution in the higher frequency region [26].

2.2.3. IMFCC

IMCC is also a kind of filter bank based cepstral features. The difference with MFCC is that IMFCC uses filters that are linearly placed in "inverted-mel" scale, which lays more stress on the high-frequency region [26].

2.2.4. STFT gram

Let $x(n)$ be a given speech sequence and $X_n(\omega)$ its STFT after applying a window $w(n)$ on the speech signal $x(n)$. $X(\omega)$ can be expressed as

$$X_n(\omega) = |X_n(\omega)| e^{j\theta_n(\omega)}. \tag{2}$$

where $|X_n(\omega)|$ corresponds to the short-time magnitude spectrum and $\theta_n(\omega)$ corresponds to the phase spectrum.

The square of the magnitude spectrum is called the STFT power spectrum. We adopt the logarithm of the power spectrum as the STFT gram.

2.2.5. Group delay gram

Most spectral features are derived from the STFT magnitude spectrum, while the short-time phase spectrum is not used, considering that the fact that the human ear is phase "deaf". However, the phase spectrum has been used for various speech processing tasks including synthetic speech detection and audio replay detection recently [19, 27–29]. Paliwal *et al.* have shown that deviations in the phase that are not linear are important for perception [30], and deviations in phase can be represented using the group delay function.

$$\tau(\omega) = -\frac{d(\theta(\omega))}{d\omega} \tag{3}$$

where the phase spectrum $\theta(\omega)$ of a signal is defined as a continuous function of ω. The group delay function can also be computed from the signal as in [28] using

$$\begin{aligned} \tau_x(\omega) &= -\Im\left[\frac{d(\log(X(\omega)))}{d\omega}\right] \\ &= \frac{X_{\Re}(\omega)Y_{\Re}(\omega) + Y_{\Im}(\omega)X_{\Im}(\omega)}{|X(\omega)|^2} \end{aligned} \tag{4}$$

where $\Re$ and $\Im$ refer to the real and imaginary parts of the Fourier transform. $X(\omega)$ and $Y(\omega)$ are the Fourier transforms of $x(n)$ and $nx(n)$, respectively.

2.2.6. Joint gram

Regarding that the speech signal is completely characterized by both the short-time magnitude and phase spectrum, here we combine that STFT gram and GD gram together to form a new feature representation. Typically, we can consider the input feature sequence as a $1 \times D \times T$ "image", where 1 denotes the image channel, D denotes the image height, T denotes the image width. Thus the joint gram is formed as a two-channel "image". For example, given separated STFT gram and JD gram of size $512 \times T$ from the same utterance, a joint gram of size $2 \times 512 \times T$ is formed as the DNN input.

2.3. Data augmentation

DA, which is a common strategy adopted to increase the quantity of training data, has been shown to be effective for neural network training to make robust predictions. Many methods such as the x-vector system rely on external datasets/resource to perform data augmentation. However, external speech data is strictly forbidden for the ASVspoof 2019 challenge. Considering that, we adopt the DA strategy by applying a simple speed perturbation. Specifically, we apply standard 3-way speed perturbation using factors of 0.9, 1.0 and 1.1 [31]. After that, we have training data of 3 times the original for both bona fide and spoof category.

2.4. Model selection and fusion

Ideally, we can use the development dataset to select the best DNN checkpoint and then train the ensemble parameter on the development set for fusion. However, the evaluation plan indicates that we should submit both development and evaluation set scores. If we do so, the performance on the development set will be overfitting and lead to a rather low error rate.

Regarding this fact, we treat the development set the same as the evaluation set. This means that we only use the training set for both DNN training and system fusion. Since we have no separated validation set, the converged model after the last optimization step is selected for evaluation. As for the system fusion, we only adopt a simple score-level average operation to get the final ensemble score.

Table 1: *Detailed network structure and number of parameters for each module. It downsamples at the last three blocks (Res2, Res3, and Res4) using stride=2 for their first convolutional layer. The total number of parameters is about 1.33M.*

Layer	Input size	Output size	Structure	#Params
Conv1	$1\times512\times L$	$16\times512\times L$	3×3, stride 1	144
Res1	$16\times512\times L$	$16\times512\times L$	$\begin{bmatrix} 3\times3, 16 \\ 3\times3, 16 \end{bmatrix}\times3$	$4K \times 3$
Res2	$16\times512\times L$	$32\times256\times\frac{L}{2}$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix}\times4$	$18K \times 4$
Res3	$32\times256\times\frac{L}{2}$	$64\times128\times\frac{L}{4}$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times6$	$72K \times 6$
Res4	$64\times128\times\frac{L}{4}$	$128\times64\times\frac{L}{8}$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times3$	$288K \times 3$
GAP	$128\times64\times\frac{L}{8}$	128	Pooling	0
FC	128	32	Fully-connected	4K
Output	32	2	Fully-connected	66

3. Experiments

3.1. Data protocol and evaluation metrics

We strictly respect the official protocols defined in the evaluation plan and submit both development and evaluation set scores. We have 54 000 training utterances altogether, including 5400 bona fide audio and 48 600 spoofed audio from various replay conditions.

The primary metric is the minimum normalized tandem decision cost function (min-tDCF) defined by the organizer [32]. An alternative equal error rate (EER) is adopted as a secondary metric.

3.2. Feature extraction

The LFCC feature, CQCC feature, as well as the GMM classifier, is exactly followed the official baseline provided by the organizer.

For IMFCC feature, 20-dimensional static coefficients are augmented with their delta and double delta coefficients, making 60-dimensional IMFCC feature vectors.

For STFT gram and GD gram, the FFT bins are set to 1024. The frame length is 25 ms, with a frameshift 10 ms. Finally, we get 512 feature vector for each frame.

3.3. Network setup

To get higher-level abstract representation, we design a deep CNN based on the well-known residual neural network (ResNet) [33]. We adopt the ResNet-34 backbone, but our network is slightly "thin" that the output channels of the front-end ResNet are only up to 128, which leads to a small number of total parameters. The core modules and the corresponding parameters are described in Table 1. We use batch normalization followed by the ReLU activation functions after every convolutional layer. The convolutional layer is all with kernels of 3×3.

We adopt the common stochastic gradient descent (SGD) algorithm with momentum 0.9 and weight decay 1e-4. The learning rate is set to 0.1, 0.01, 0.001 and is switched when the training loss plateaus. In the training stage, the model is

Table 2: *Comparison of different classifiers on the development set*

Feature	Classifier	EER(%)	min-tDCF
CQCC	GMM	9.87	0.1953
CQCC	**ResNet**	**4.77**	**0.1127**
LFCC	GMM	11.96	0.2554
LFCC	**ResNet**	**2.62**	**0.0609**

Table 3: *Comparison of different types of feature on the development set*

Feature	EER(%)	min-tDCF
CQCC	4.77	0.1127
LFCC	2.62	0.0609
IMFCC	3.66	0.0890
STFT gram	4.11	0.1070
GD gram	**1.81**	**0.0467**

trained with a mini-batch size of 128. We design a data loader to generate the variable-length training examples on the fly. For each training step, an integer L within [150,350] interval is randomly generated, and each data in the mini-batch is truncated or extended to L frames. Therefore, a dynamic mini-batch of data with the shape of $128 \times D \times L$ is generated prior to each training step, where D is the input feature dimension, and L is a batch-wise variable number indicating the frame length.

We choose the output from the bona fide unit as our final score. In the testing stage, the full-length feature sequence is directly fed into the network, without any truncate or padding operation. Due to the fact that utterances may have arbitrary durations, we feed the testing audio to the trained network one by one.

3.4. Results on back-end classification

To compare the back-end classifier, we fix the front-end feature representation as CQCC/LFCC in this section. From Table 2, we can see that no matter for CQCC or LFCC feature, the ResNet classifier based on our introduced utterance-level DNN framework is significantly superior to the baseline GMM classifier. Therefore, we fix the back-end classifier as the ResNet model for the following experiments.

3.5. Results on front-end feature representation

In this section, we fix the ResNet back-end classifiers and perform experiments on different types front-end feature representations. From Table 3, we can see that the performance of the countermeasure system strongly depends on the choice of the feature representation. The GD gram based on the phase spectrum yields the best among the investigated five features.

3.6. Results on data augmentation

The results in Table 4 show the effectiveness of the DA based on speed perturbation strategy. 18% relative EER reduction is achieved on the development set for the STFT gram system, and 48% for the GD gram system.

Table 4: *Effect of the DA strategy on the development set*

Feature	DA	EER(%)	min-tDCF
STFT gram	✗	4.11	0.1070
STFT gram	✓	**3.35**	**0.0904**
GD gram	✗	1.81	0.0467
GD gram	✓	**1.03**	**0.0265**

Table 5: *Overall performance on both the development set and the evaluation set. EER are written before the slash, min-tDCF are behind. N/R: Not Reported.*

ID	System	DA	Dev	Eval
1	CQCC–GMM	✗	9.87/0.1953	11.04/0.2454
2	LFCC–GMM	✗	11.96/0.2554	13.54/0.3017
3	LFCC–ResNet	✗	2.62/0.0609	N/R
4	IMFCC–ResNet	✗	3.66/0.0890	N/R
5	STFT gram–ResNet	✓	3.35/0.0904	N/R
6	GD gram–ResNet	✗	1.81/0.0467	1.79/0.0439
7	GD gram–ResNet	✓	**1.03/0.0265**	**1.08/0.0282**
8	Joint gram–ResNet	✓	1.14/0.0209	1.23/0.0305
Fusion 3+4+5+6+7+8			**0.24/0.0064**	**0.66/0.0168**

3.7. Results on fusion and generalization

Table 5 shows the system performance on both the development and the evaluation set. Since only four systems can be submitted at most, some system performance on the evaluation set could not be reported at this stage. Our best GD gram–ResNet system achieves relative 90% EER reduction as well as min-tDCF on both development set and evaluation set. Finally, using the simple average score from several single systems can further improve the performance. EER of 0.24% on the development set and 0.66% on the evaluation set is obtained for our primary system. This reveals the complementariness of the system outputs from classifiers trained with different features.

The performance on the development set and evaluation set are very close, reveals the robustness of our proposed system.

4. Conclusion

In this paper, we take the DKU system for ASVspoof 2019 challenge as the pivot and explore the design of the countermeasure system against replay attack through data augmentation, feature representation, classification, and fusion. First, we utilize a ResNet back-end classifier based on the introduced utterance-level deep learning framework. Second, we investigate various kinds of front-end feature representations and find that the GD gram based on the phase spectrum is pretty effective for replay detection. Moreover, a simple data augmentation based on speed perturbation can also significantly improve the performance, especially for the GD gram–ResNet system. Finally, both the single and primary fusion systems dramatically decrease the EER and min-tDCF on not only on the development set but also on the evaluation set.

5. References

[1] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE Transactions on Speech & Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[2] 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.

[3] J. H. L. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[4] N. Evans, T. Kinnunen, and J. Yamagishi, "Spoofing and countermeasures for automatic speaker verification," in *Proc. INTERSPEECH 2013*, 2013.

[5] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, "Spoofing and countermeasures for speaker verification: A survey," *Speech Communication*, vol. 66, pp. 130–153, 2015.

[6] S. K. Ergunay, E. Khoury, A. Lazaridis, and S. Marcel, "On the vulnerability of speaker verification to realistic voice spoofing," in *IEEE International Conference on Biometrics Theory, Applications and Systems*, 2015, pp. 1–6.

[7] Z. Wu, J. Yamagishi, T. Kinnunen, C. Hanilci, M. Sahidullah, A. Sizov, N. Evans, M. Todisco, and H. Delgado, "Asvspoof: the automatic speaker verification spoofing and countermeasures challenge," *IEEE Journal of Selected Topics in Signal Processing*, vol. PP, no. 99, pp. 1–1, 2017.

[8] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilc, S. M., and S. Aleksandr, "Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *Proc. INTERSPEECH 2015*, 2015.

[9] T. Kinnunen, M. Sahidullah, M. Falcone, L. Costantini, R. G. Hautamki, D. Thomsen, A. Sarkar, Z. Tan, H. Delgado, and M. Todisco, "Reddots replayed: A new replay spoofing attack corpus for text-dependent speaker verification research," in *Proc. ICCASP 2017*, 2017.

[10] 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*, 2017, pp. 2–6.

[11] H. Delgado, M. Todisco, M. Sahidullah, N. Evans, T. Kinnunen, K. A. Lee, and J. Yamagishi, "Asvspoof 2017 version 2.0: meta-data analysis and baseline enhancements," in *Proc. Speaker Odyssey 2018*, 2018, pp. 296–303.

[12] Asvspoof 2019 evaluation plan. [Online]. Available: http://www.asvspoof.org/asvspoof2019/asvspoof2019_evaluation_plan.pdf

[13] C. Hanili, T. Kinnunen, M. Sahidullah, and A. Sizov, "Classifiers for synthetic speech detection: A comparison," in *Proc. INTERSPEECH 2015*, 2015.

[14] M. Todisco, H. Delgado, and N. Evans, "A new feature for automatic speaker verification anti-spoofing: Constant q cepstral coefficients," in *Proc. Speaker Odyssey 2016*, 2016.

[15] J. C. Brown, "Calculation of a constant q spectral transform," *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.

[16] 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.

[17] C. Zhang, C. Yu, and J. H. L. Hansen, "An investigation of deep-learning frameworks for speaker verification antispoofing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 684–694, June 2017.

[18] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio replay attack detection with deep learning frameworks," in *Proc. INTERSPEECH 2017*, 2017, pp. 82–86.

[19] F. Tom, M. Jain, and P. Dey, "End-To-End Audio Replay Attack Detection Using Deep Convolutional Networks with Attention," in *Proc. INTERSPEECH 2018*, 2018, pp. 681–685.

[20] W. Cai, D. Cai, W. Liu, G. Li, and M. Li, "Countermeasures for automatic speaker verification replay spoofing attack : On data augmentation, feature representation, classification and fusion," in *Proc. INTERSPEECH 2017*, 2017, pp. 17–21.

[21] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," in *Proc. Speaker Odyssey*, 2018, pp. 74–81.

[22] W. Cai, Z. Cai, W. Liu, X. Wang, and M. Li, "Insights into end-to-end learning scheme for language identification," in *Proc. ICASSP 2018*, 2018, pp. 5209–5213.

[23] W. Cai, J. Chen, and M. Li, "Analysis of length normalization in end-to-end speaker verification system," in *Proc. INTERSPEECH 2018*, 2018.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. CVPR 2009*, 2009, pp. 248–255.

[25] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. ICLR 2014*, 2014.

[26] M. Sahidullah, T. Kinnunen, and C. Hanili, "A comparison of features for synthetic speech detection," in *Proc. INTERSPEECH 2015*, 2015, pp. 2087–2091.

[27] H. A. Murthy and V. Gadde, "The modified group delay function and its application to phoneme recognition," in *Proc. ICASSP 2003*, vol. 1, 2003, pp. I–68.

[28] R. M. Hegde, H. A. Murthy, and V. R. R. Gadde, "Significance of the Modified Group Delay Feature in Speech Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 190–202.

[29] I. Saratxaga, J. Sanchez, Z. Wu, I. Hernaez, and E. Navas, "Synthetic speech detection using phase information," *Speech Communication*, vol. 81, pp. 30 – 41, 2016.

[30] K. K. Paliwal and L. D. Alsteris, "On the usefulness of stft phase spectrum in human listening tests," *Speech Communication*, vol. 45, no. 2, pp. 153 – 170, 2005.

[31] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. INTERSPEECH 2015*, 2015, pp. 3586–3589.

[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," in *Proc. Speaker Odyssey 2018*, 2018, pp. 312–319.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR 2016*, 2016, pp. 770–778.