Interspeech 2022
18-22 September 2022, Incheon, Korea

Overlapped Frequency-Distributed Network: Frequency-Aware Voice Spoofing Countermeasure

Sunmook Choi[1], *Il-Youp Kwak*[2], *Seungsang Oh*[1]

[1]Korea University, Republic of Korea
[2]Chung-Ang University, Republic of Korea

felixchoi@korea.ac.kr, ikwak2@cau.ac.kr, seungsang@korea.ac.kr

Abstract

Numerous IT companies around the world are developing and deploying artificial voice assistants via their products, but they are still vulnerable to spoofing attacks. Since 2015, the competition "Automatic Speaker Verification Spoofing and Countermeasures Challenge (ASVspoof)" has been held every two years to encourage people to design systems that can detect spoofing attacks. In this paper, we focused on developing spoofing countermeasure systems mainly based on Convolutional Neural Networks (CNNs). However, CNNs have translation invariant property, which may cause loss of frequency information when a spectrogram is used as input. Hence, we propose models which split inputs along the frequency axis: 1) Overlapped Frequency-Distributed (OFD) model and 2) Non-overlapped Frequency-Distributed (Non-OFD) model. Using ASVspoof 2019 dataset, we measured their performances with two different activations; ReLU and Max feature map (MFM). The best performing model on LA dataset is the Non-OFD model with ReLU which achieved an equal error rate (EER) of 1.35%, and the best performing model on PA dataset is the OFD model with MFM which achieved an EER of 0.35%.

Index Terms: Deep learning, audio deep synthesis, spoofing, fake audio detection, countermeasure

1. Introduction

As technology advances, the usage of intelligent virtual assistants, such as Samsung Bixby, Apple Siri, Google Assistant, and Amazon Alexa, has been steadily increasing in recent years. Such assistants also support commands related to users' security or privacy, which could be targeted by criminals. Therefore, to prevent disasters, voice assistants should detect spoofing attacks. The two main scenarios of spoofing attacks are logical access (LA) scenarios and physical access (PA) scenarios. The data in the LA condition is generated by text-to-speech synthesis and voice conversion technology, and the data in the PA condition is generated by replaying the recorded victim's voice.

Several competitions in the 2010s, such as INTERSPEECH 2013 [1], AVspoof 2015 [2], ASVspoof 2015 [3], ASVspoof 2017 [4], and ASVspoof 2019 [5], encouraged people to design better voice spoofing detection systems. Many methods, such as Short-time Fourier Transform (STFT) and Constant-Q Transform (CQT), have been used to transform raw voice data into 2D features in a time-frequency domain, and the 2D features, called spectrograms, are applied to machine learning or deep learning algorithms. Especially in deep learning, a spectrogram feature of an audio data is usually applied to Convolutional Neural Networks (CNNs) as an input.

In ASVspoof 2019, most of top-ranked teams usually used CNN-based networks. The team T45 [6] ranked second for both LA and PA scenarios, resulting 1.86% and 0.54% in equal error rate (EER), respectively. The team used an ensemble model of LCNN-based networks with various input features extracted by CQT, Fast Fourier Transform, and Discrete Cosine Transform. The team T44 [7] ranked third for PA scenario resulting 0.59% in EER, and the team used an ensemble of networks based on SENet and ResNet with log power magnitude spectra and constant Q cepstral coefficients as input features. The team T60 [8] ranked third for LA scenario resulting 2.64% in EER. They used not only CNN-based deep models (CNN, Convolutional Recurrent Neural Networks, Wave-U-Net) but also shallow models (Gaussian Mixture Models and Support Vector Machines).

In CNNs, convolutional layers find common features in an entire image using significantly fewer parameters than fully connected layers. Therefore, they are used in deep models to control computational cost or memory issues as well as to add nonlinearities. Additionally, max pooling layers have *translation invariant property*, which takes a crucial role in image classification problems. However, since a spectrogram feature contains frequency information on the vertical axis, the translation invariant property of CNN may cause a loss of useful frequency information. Thus, CNN layers should be applied in a sophisticated way for spectrogram features.

Unlike usual 2D images, even if one feature (*e.g.* an edge) that appears at the top of a spectrogram also appears at the bottom of the spectrogram, that does not mean the same thing because the feature is in a different frequency range. Thus, it would be more effective to individually process features in the high-, mid-, and low-frequency ranges. We tried to partition a spectrogram along the frequency axis to obtain multiple sub-bands. We then applied convolutional layers to each sub-band to find features in different frequency ranges individually. We can also expect that the network will not suffer from the translation invariant problem due to the splitting process. Additionally, we considered splitting a spectrogram allowing overlapping parts to avoid missing any important features near the splitting lines.

In this paper, we introduce two types of models: Overlapped frequency-distributed (OFD) model and Non-overlapped frequency-distributed (Non-OFD) model. These models take a spectrogram as input, and split it along the frequency axis. Using the CQT feature, we evaluated their performances by EER on LA and PA data from ASVspoof 2019 competition with two different activations; ReLU [9] and Max feature map (MFM) [10]. The top performing model on LA set is a Non-OFD model with ReLU which achieved an EER of 1.35%, and the top performing model on PA set is an OFD model with MFM which achieved an EER of 0.35%. Moreover, since our proposed models consist of blocks which are designed to be light-weight, the numbers of parameters of our top performing models are only 7.1% and 18.0% of the numbers of parameters of the models submitted by the team T45.

2. Methodology

2.1. Feature engineering

Mel-spectrogram and CQT features are widely used in spoofing countermeasure systems. After a few trials using those features, CQT features performed better with our proposed models, and we focused on using CQT features. Since each audio data has a different length, the sample length is fixed at 9 seconds. If a sample is longer than 9 seconds, the first 9 seconds are used; otherwise, the sample is replayed to fill 9 seconds. Each data is sampled at 16kHz and the python package librosa [11] is used to extract CQT features. The number of frequency bins are set to 120. The minimum frequency of our CQT feature is set to 1 Hz. With these settings, the input size of a model becomes $(120, 282, 1) \in \mathbb{R}^{H\times W\times C}$.

2.2. MFM activation

Max feature map (MFM) is a specific version of Maxout activation [12] for convolutional layers introduced in LCNN architecture [10]. Consider the case when we want a feature map $F \in \mathbb{R}^{H\times W\times C}$ as the output of a convolutional layer. When MFM is applied, we take $2C$ number of filters in the layer to produce a feature map $G \in \mathbb{R}^{H\times W\times 2C}$. Then, we divide G into two parts along the channel axis, obtaining G_1 and G_2 of the size $H \times W \times C$. The output F of the layer through MFM is the elementwise maximum between them, that is,

$$F = \max\{G_1, G_2\} \in \mathbb{R}^{H\times W\times C}. \tag{1}$$

2.3. Overlapped frequency-distributed block

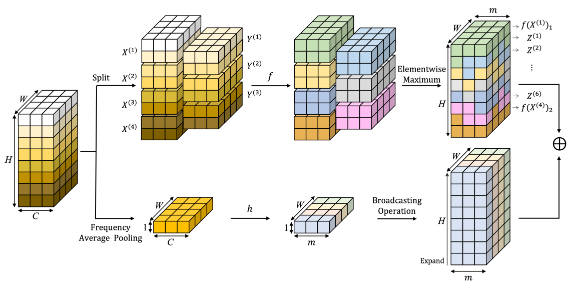
Each overlapped frequency-distributed (OFD) block has two streams, each of which learns frequency-related features and temporal features, and it is described in Figure 1(a). By taking elementwise addition of the two streams, the block builds new features of the input from two different sources.

In the first stream, the input of a block is split into multiple parts along the frequency axis, and each split part covers the entire time axis at a specific frequency interval. The block then looks for individual frequency-related features in each split part. The first stream was inspired by FreqCNN model [13]. At the same time, in the second stream, the entire image is averaged along the frequency axis to learn only temporal features, and then reconstructed back into a 2D image. The second stream was inspired by BC-ResNet model [14, 15]. By adding the results of these two streams, the block can combine frequency-related and temporal features obtained independently. The details are described below.

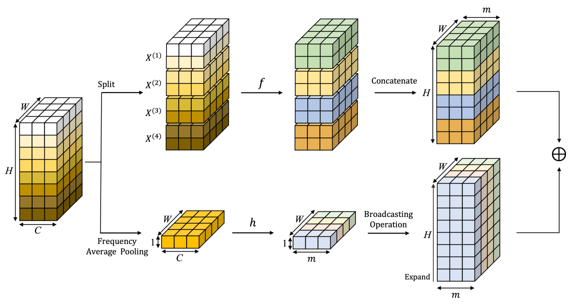
The first stream of the block learns frequency-related features of an input. Let $X = (x_{ij}) \in \mathbb{R}^{H\times W}$ be the input with the frequency and temporal dimensions, H and W, respectively. Here, the channel dimension is omitted for a brief explanation. Let n be the number of splits along the frequency axis. If necessary, X is zero-padded along the frequency axis so that H is divisible by $2n$. Let $s = \frac{H}{2n}$. First the input X is split into n disjoint parts $\{X^{(k)}\}_{k=1}^{n} \subseteq \mathbb{R}^{2s\times W}$ and $n-1$ *overlapped parts* $\{Y^{(k)}\}_{k=1}^{n-1} \subseteq \mathbb{R}^{2s\times W}$ where

$$\begin{cases} X^{(k)} = (x_{ij}), & 1{+}2(k{-}1)s \leq i \leq 2ks \\ Y^{(k)} = (x_{ij}), & 1{+}(2k{-}1)s \leq i \leq (2k{+}1)s. \end{cases} \tag{2}$$

Notice that the upper half of $Y^{(k)}$ and the lower half of $X^{(k)}$ are exactly the same, and so are the lower half of $Y^{(k)}$ and the upper half of $X^{(k+1)}$. This is why we call $Y^{(k)}$'s overlapped



(a) OFD block description for $n = 4$.



(b) Non-OFD block description for $n = 4$.

Figure 1: *OFD block and Non-OFD block.*

parts. Let f be a composite of $k_1 \times 1$ convolution, batch normalization (BN) [16], ReLU activation, $k_1 \times 1$ convolution, and BN. Note that no activation is used after the second convolution. Moreover, setting stride to 1 and using zero-padding are necessary in each convolution to keep the size unchanged. Now, the images $f(X^{(k)})$'s and $f(Y^{(k)})$'s are combined to reconstruct a full 2D image of the size $H \times W$. Each of the images is equally divided into two parts; the upper half $f(X^{(k)})_1$ or $f(Y^{(k)})_1$, and the lower half $f(X^{(k)})_2$ or $f(Y^{(k)})_2$ of the size $s \times W$. Let

$$\begin{cases} Z^{(2k-1)} &= \max\left\{f(X^{(k)})_2,\ f(Y^{(k)})_1\right\} \\ Z^{(2k)} &= \max\left\{f(X^{(k+1)})_1,\ f(Y^{(k)})_2\right\}, \end{cases} \tag{3}$$

where max is an elementwise maximum. Here, max acts as an MFM activation and it justifies no activation after the second convolution in the function f. Finally, we obtain a set of sub-images,

$$\left\{f(X^{(1)})_1,\ Z^{(1)},\ \ldots,\ Z^{(2n-2)},\ f(X^{(n)})_2\right\}, \tag{4}$$

each of which has the size $s \times W$. Then, the final output of the first stream is obtained by concatenating these $2n$ sub-images in given order along the frequency axis, resulting the size $H \times W$.

The second stream of the block learns temporal features of the input. First, the input X is averaged out along the entire frequency axis, resulting in a feature map $\overline{X} \in \mathbb{R}^{1\times W}$. Let h be a composite of $1 \times k_2$ depthwise convolution [17, 18], BN, swish activation [19], 1×1 pointwise convolution, ReLU activation, and spatial dropout [20]. For the depthwise convolution, we used dilated convolution with dilation rate 4 to consider temporal features in a wide range. Moreover, setting stride to 1 and using zero-padding are again required to keep the temporal dimension unchanged. The dropout rate is fixed to 0.5 in the spatial dropout layer, which randomly drops out entire feature maps, helping the model promote independence between feature maps. Lastly, the feature map $h(\overline{X})$ is expanded along the frequency axis using broadcasting operation [14, 15] to reconstruct a feature map of the size $H \times W$. That is, the output of the second stream is $Y = (y_{ij}) \in \mathbb{R}^{H \times W}$, where

$$(y_{i1}, \ldots, y_{iW}) = h(\overline{X}) \qquad (5)$$

for all $i = 1, \ldots, H$. This operation is indeed a simple upsampling layer.

Finally, the output of the block is obtained by adding the results of these two streams. Note that the summation is possible if the number of filters used in the second convolution in f is equal to that used in the second convolution in h.

2.4. Non-overlapped frequency-distributed block

We additionally considered a non-overlapped version of OFD block, which we call "Non-overlapped frequency-distributed (Non-OFD) block". The only difference from OFD block is the splitting part in the first stream as described in Figure 1 (b). The details are described below.

Let $X = (x_{ij}) \in \mathbb{R}^{H \times W}$ be an input of the block and let n be the number of splits along the frequency axis. If necessary, X is zero-padded along the frequency axis so that H is divisible by n. Let $s = \frac{H}{n}$. We split X only into n disjoint parts $\{X^{(k)}\}_{k=1}^{n}$ where

$$X^{(k)} = (x_{ij}) \in \mathbb{R}^{s \times W}, \;\; 1 + (k-1)s \leq i \leq ks. \qquad (6)$$

Let f be a composite of $k_1 \times 1$ convolution, BN, ReLU, $k_1 \times 1$ convolution, BN, and ReLU. All images $f(X^{(k)})$ are concatenated in the same order along the frequency axis to reconstruct a full 2D image, resulting the size $H \times W$.

2.5. Non-split block

We also defined a non-split block for deeper blocks. Similarly, this block consists of two streams, and its second stream is identical to OFD and Non-OFD blocks, whereas its first stream is slightly different in that it does not split the input. Let f be a composite of $k_1 \times 1$ convolution, BN, ReLU, $k_1 \times 1$ convolution, BN, and ReLU. Then, for input X, the output of the first stream is just $f(X)$.

2.6. Model architecture

We introduce a model that consists of either six OFD blocks or six Non-OFD blocks, which we call OFD model or Non-OFD model, respectively. The architecture of the models is described in Figure 2.

When a CQT feature is given as an input of the model, it is first applied to a convolutional layer with 16 filters of the size 5×5 followed by ReLU activation. Next, it is applied to six blocks, which are either OFDs or Non-OFDs. Each block is followed by 2×2 max pooling to reduce the size and computational resource. The same number m of filters are used in convolutions in both streams in a given block. Each block is expressed as 'Block(T/F, n, k_1, k_2, m)' in Figure 2, where T/F denotes whether the overlap is true (T) or false (F), n denotes the number of splits, and k_1 and k_2 denote the filter sizes in the functions f and h, respectively. After six blocks and max pooling layers, global average pooling is applied, resulting in 128 units, which is the number of channels of the output of the last block. The final layer is a dense layer of two units with softmax, each of which corresponds to bona fide or spoof.

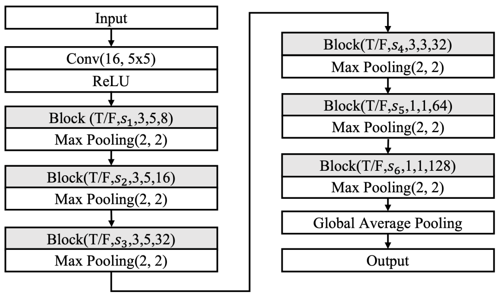


Figure 2: *Model architecture for both OFD and Non-OFD.*

The first hyperparameter overlap in all blocks of an OFD (resp. Non-OFD) model are set to T (resp. F). But if we set $n = 0$ in a block, then the block defines a non-split block regardless of the overlap. Moreover, we set $k_1 = k_2 = 1$ in the 5th and 6th blocks to control the receptive field in the network [21].

3. Experiments

3.1. Experimental setup

We had an experiment with logical access (LA) and physical access (PA) data from ASVspoof 2019 competition [5]. Each dataset contains the training, validation, and evaluation sets. Our network is trained with the training set, and the performance is evaluated with the evaluation set. We used an equal error rate (EER) for the performance measure, which was the evaluation rule of ASVspoof 2019 competition.

The experiment was conducted with various splits to find a best performing model. Each OFD or Non-OFD model is expressed with a tuple $(s_1, s_2, s_3, s_4, s_5, s_6)$ which shows the numbers of splits used in six blocks, that is, s_i is the number of splits in the ith block of the model.

We trained each model for 30 epochs with adam optimizer [22]. The batch size is set to 16. The learning rate is set to start at 10^{-3} and end at 10^{-5} with sigmoidal decay. Furthermore, since the number of spoof data far outpaced the number of bona fide data, the training was conducted by giving weight to bona fide data. The weight is heuristically chosen to be 5.0.

3.2. MFM activation instead of ReLU activation

The usage of MFM activation in LCNN [10] was proven to work well for spoofing attacks via ASVspoof competitions in 2017 [23] and 2019 [6]. Moreover, ResMax model [24] is designed to be a spoofing countermeasure system with MFM after the competition, and it performed well compared to the top five models from ASVspoof 2019 competition. Therefore, we additionally experimented using MFM. For OFD and Non-split blocks, MFM replaces every ReLU in the function f in the first stream. For Non-OFD blocks, MFM only replaces the first ReLU in the function f in the first stream. Also, we changed the order of BN and activation when MFM was applied.

3.3. Experiment results

The training and evaluation sessions were conducted ten times for each model since random weight initialization may lead to different converged weights. The EERs for validation set and evaluation set in Table 1 are the average values of the top five EERs among ten experiments.

For LA scenarios, the top performing model is Non-OFD model with (2,2,2,2,2,2) splits and ReLU activation, which achieved an EER of 1.35% with 106K number of parameters. Even though the EER for evaluation set did not fall below 1%, it ranked 2nd among the submitted models with significantly fewer parameters. Moreover, it is notable that the top 3 performing models among ours for LA scenarios are Non-OFD models with 2 splits in the first block. These models indirectly imply that it is effective to focus on high- and low-frequency ranges separately for spoofing data in LA conditions.

For PA scenarios, the top performing model is OFD model with (2,2,2,2,2,2) splits and MFM activation, which achieved an EER of 0.35% with 200K number of parameters. The model outperformed all models submitted to the competition with much fewer parameters. Moreover, we can notice that the top 2 performing models among ours for PA scenarios are OFD models with 2 splits in the first block. These models indirectly show that it is more effective to focus not only on high- and low-frequency ranges but also on mid-frequency range with overlapping to detect spoofing data in PA conditions.

Additionally, we mention that Non-OFD model with (2,2,0,0,0,0) splits and MFM also performed excellently for both LA and PA data with only 51K number of parameters. Even if we construct a model which splits more in former blocks and doesn't split in latter blocks, it still performs well with not too many parameters. For example, both Non-OFD model with (8,4,2,0,0,0) splits and MFM and OFD model with (8,4,2,0,0,0) splits with ReLU performed great with only 70K and 62K number of parameters, respectively.

4. Discussion

Recently, the Audio Deep Synthesis Detection Challenge (ADD 2022) [25] was launched, which is an IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2022 Signal Processing Grand Challenge. We participated in track 1 of ADD 2022 [26], which is to detect LA spoofing attacks in noisy environments such as real-world noises and background music effects, and we took 3rd place using an ensemble which includes OFD model with (6,3,2,2,2,2) splits and ReLU activation. To overcome the noisy environment, we introduced a data augmentation method; frequency feature masking (FFM) [26]. This method allows a model to focus on noise-free areas by masking a specific frequency band. However, even though FFM wasn't applied to OFD model, the performance was similar to that of the models to which FFM was applied. We can infer that OFD model does separately consider information in different frequency ranges.

However, even though OFD and Non-OFD models show great performance, several hyperparameters are needed to define our models. This causes that finding an optimal model is a time-consuming task. Therefore, a further study to reduce time for hyperparameter tuning must be followed.

5. Conclusions

We checked that our proposed blocks and models perform well to resolve spoofing problems. In comparison to the submitted models in ASVspoof 2019 competition, Non-OFD model with (2,2,2,2,2,2) splits and ReLU activation achieved an EER of 1.35% for LA scenarios, and OFD model with (2,2,2,2,2,2) splits and MFM activation achieved an EER of 0.35% for PA scenarios, ranking second and first, respectively. Additionally, our solutions using MFM achieved comparable performance with other solutions that use ReLU. It was preferable to perform a frequency split of 2 in the ASVspoof 2019 dataset. However, in various audio classification issues, it will be required to attempt several splits and find a model with decent results.

Table 1: *Our model performances on the ASVspoof 2019 validation set and evaluation set compared to the top three models submitted to the competition. The performance is measured by an equal error rate(EER, %). #Mo and #Params denote the number of different models in the ensemble system and the number of total parameters in the model, respectively.*

LA

#	Model	Valid EER	Eval EER	#Mo	#Params
1	T05	-	0.22	-	-
2	***Non-OFD*** ***(2,2,2,2,2,2)-ReLU***	0.356	1.35	1	106K
3	***Non-OFD*** ***(2,2,2,2,2,2)-MFM***	0.352	1.45	1	138K
4	***Non-OFD*** ***(2,2,0,0,0,0)-MFM***	0.495	1.67	1	51K
5	***OFD*** ***(4,2,0,0,0,0)-MFM***	0.423	1.75	1	58K
6	***Non-OFD*** ***(8,4,2,0,0,0)-MFM***	0.379	1.815	1	70K
7	***OFD*** ***(8,4,2,0,0,0)-ReLU***	0.383	1.819	1	62K
8	T45 [6]	0.000	1.86	5	1484K
9	T60 [8]	0.0	2.64	4	-

PA

#	Model	Valid EER	Eval EER	#Mo	#Params
1	***OFD*** ***(2,2,2,2,2,2)-MFM***	0.171	0.35	1	200K
2	T28	-	0.39	-	-
3	***OFD*** ***(2,2,2,2,2,2)-ReLU***	0.163	0.40	1	152K
4	***Non-OFD*** ***(2,2,0,0,0,0)-MFM***	0.222	0.44	1	51K
5	***OFD*** ***(4,2,0,0,0,0)-MFM***	0.156	0.49	1	58K
6	T45 [6]	0.015	0.54	3	1113K
7	***Non-OFD*** ***(8,4,2,0,0,0)-MFM***	0.216	0.55	1	70K
8	***OFD*** ***(8,4,2,0,0,0)-ReLU***	0.249	0.57	1	62K
9	T44 [7]	0.129	0.59	5	5811K

6. Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2020R1C1C1A01013020) and Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00033, 50%, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity).

7. References

[1] N. W. Evans, T. Kinnunen, and J. Yamagishi, "Spoofing and countermeasures for automatic speaker verification," in *Proc. Interspeech 2013*, 2013, pp. 925–929.

[2] S. K. Ergunay, E. Khoury, A. Lazaridis, and S. Marcel, "On the vulnerability of speaker verification to realistic voice spoofing," in *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, Sep. 2015, pp. 1–6.

[3] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilci, M. Sahidullah, and A. Sizov, "Asvspoof 2015: The first automatic speaker verification spoofing and countermeasures challenge," in *Proc. Interspeech 2015*. Dresden: ISCA, 2015, pp. 2037–2041.

[4] 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*. Stockholm: ISCA, 2017, pp. 2–6.

[5] M. Todisco, X. Wang, V. Vestman, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. H. Kinnunen, and K. A. Lee, "ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection," in *Proc. Interspeech 2019*, 2019, pp. 1008–1012.

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

[7] C.-I. Lai, N. Chen, J. Villalba, and N. Dehak, "ASSERT: Anti-Spoofing with Squeeze-Excitation and Residual Networks," in *Proc. Interspeech 2019*, 2019, pp. 1013–1017.

[8] 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*, 2019, pp. 1018–1022.

[9] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 807–814.

[10] 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, Nov 2018.

[11] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015.

[12] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 28, no. 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1319–1327.

[13] Y. Wu, H. Mao, and Z. Yi, "Audio classification using attention-augmented convolutional neural network," *Knowledge-Based Systems*, vol. 161, pp. 90–100, 2018.

[14] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

[15] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," in *Proc. Interspeech 2021*, 2021, pp. 4538–4542.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 448–456.

[17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[18] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, pp. 4510–4520.

[19] P. Ramachandran, B. Zoph, and Q. V. Le, "Swish: a self-gated activation function," *arXiv: Neural and Evolutionary Computing*, 2017.

[20] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun 2015, pp. 648–656.

[21] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1987–2000, 2021.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[23] 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.

[24] I.-Y. Kwak, S. Kwag, J. Lee, J. H. Huh, C.-H. Lee, Y. Jeon, J. Hwang, and J. W. Yoon, "ResMax: Detecting Voice Spoofing Attacks with Residual Network and Max Feature Map," in *25th International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society, 2021, pp. 4837–4844.

[25] J. Yi, R. Fu, J. Tao, S. Nie, H. Ma, C. Wang, T. Wang, Z. Tian, Y. Bai, C. Fan, S. Liang, S. Wang, S. Zhang, X. Yan, L. Xu, Z. Wen, and H. Li, "Add 2022: the first audio deep synthesis detection challenge," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. Singapore: IEEE, 2022, pp. 9216–9220.

[26] I.-Y. Kwak, S. Choi, J. Yang, Y. Lee, and S. Oh, "Cau_ku team's submission to add 2022 challenge task 1: Low-quality fake audio detection through frequency feature masking," *arXiv preprint arXiv:2202.04328*, 2022.