2022 26th International Conference on Pattern Recognition (ICPR)
August 21-25, 2022, Montréal, Québec, Canada

Light-weight Frequency Information Aware Neural Network Architecture for Voice Spoofing Detection

Sunmook Choi
and Seungsang Oh
Department of Mathematics
Korea University
Republic of Korea
Email: {felixchoi, seungsang}@korea.ac.kr

Jonghoon Yang
Yerin Lee
and Il-Youp Kwak
Department of Applied Statistics
Chung-Ang University
Republic of Korea
Email: {yjhoon2, dldpfls14, ikwak2}@cau.ac.kr

2022 26th International Conference on Pattern Recognition (ICPR) | 978-1-6654-9062-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICPR56361.2022.9956079

***Abstract*—The voice assistant market is overgrowing, and mainstream services like Bixby (Samsung), Alexa (Amazon), and Siri (Apple) are quickly being upgraded to support advanced commands. Such capabilities make them lucrative targets for attackers to exploit. Voice spoofing attacks involve recording voice commands of a target victim and simply replaying them through a loudspeaker. The "2019 Automatic Speaker Verification Spoofing And Countermeasures Challenge" (ASVspoof) competition aims to facilitate the design of highly accurate voice spoofing attack detection systems. However, most of the presented models do not take frequency-level modeling into account in their modeling architecture and do not consider model complexity. To design a light-weight system with frequency-level modeling, we propose two systems: 1) Double Depthwise Separable (DDWS) convolution and 2) BC-ResNet with max feature map (MFM) activation (BC-ResMax). We evaluate the accuracy by equal error rate (EER) using the ASVspoof 2019 dataset. Our single models of parallel DDWS, sequential DDWS, and BC-ResMax model achieved spoofing attack detection EER of 2.63%, 2.08% and 2.59% in the LA dataset, and 0.47%, 0.63% and 0.49% in the PA dataset, achieving comparable performance with other top ensemble systems from the competition. Furthermore, parallel DDWS, sequential DDWS, and BC-ResMax used only 45K, 28K and 29K numbers of parameters which are far fewer than existing models.**

I. Introduction

With the rapid growth of voice-related technologies, voice assistants now support security- and privacy-critical commands such as making purchases or controlling smart televisions. Such capabilities allows attackers to abuse the services to achieve lucrative targets. Voice spoofing attacks involve recording a target victim's voice and simply replaying them through an electronic speaker to bypass any voice biometric-based authentication service. One real funny event would be the wrong dollhouse orders from Amazon Echo. TV anchor said live on-air, 'Alexa, order me a dollhouse.' Amazon Echo ordered dollhouses in many American households after 'hearing' TV presenter talking. We could have prevented this accident if we had a good voice spoofing detection system.

Two main voice spoofing attack scenarios are Physical Access (PA) attacks and Logical Access (LA) attacks. The PA attacks are simple replay attacks that record the user's voice and play it back. The LA attacks include text to speech (TTS) attacks and voice conversion (VC) attacks. In the TTS attacks, the attacker collect lots of victim's voice samples and train the victim's voice using Google's Tacotron or Baidu's Deep Voice [1], [2].

Lots of competitions such as AVspoof 2015, ASVspoof 2015, ASVspoof 2017, ASVspoof 2019, and ASVspoof2021 were conducted to facilitate the design of highly accurate voice spoofing detection systems [3], [4], [5], [6], [7]. Many methods converted voice into a two-dimensional spectrogram arrangement of frequency and time, and then applied deep learning or machine learning methods [8], [9], [10], [11]. The Speech Technology Center (STC) applied the LCNN model to the spectrogram feature, showing that the Convolutional Neural Network (CNN) based model works well with the spectrogram feature, ranking first in 2017 and second in 2019 (in PA data) [8], [12], [13]. However, CNN-based models have a location invariance property and, due to this, frequency magnitude information in a spectrogram image gets obscured. Many existing CNN-based spoofing detection systems have not been sufficiently modeled for frequency-related information.

On the other hand, modeling for two-dimensional spectrograms that reflect frequency and time information has been attempted for other audio classification tasks such as acoustic scene classification and musical genre classification [14], [15], [16], [17]. Pons et al. (2016) experimented 1 by n and n by 1 convolutions for frequency- and temporal-related modeling [14]. Koutini et al. (2021) demonstrated that the appropriate receptive field size within a certain range should be maintained to make a better model [15], [18]. In the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 challenge on Acoustic Scene Classification (ASC), the best team proposed trident architecture that divides frequency level features into three parts [16]. In the DCASE 2021 challenge on ASC, the best team used Broadcasted Residual Network (BC ResNet) that processes frequency-related and temporal features separately in their residual network design [17], [19].

Other essential issues in the deep learning model are the latency and model complexity requirements imposed by real-life businesses. Due to users' timely response expectations and exploding server cost issues, businesses typically require model sizes to be less than a few megabytes (considering on-

device deployment scenarios) and detection (prediction) time to be less than a few ms. MobileNet and MobileNet2 are proposed to alleviate these issues [20], [21].

This paper experimented with two light-weight CNN-based neural network architectures that account for frequency information by using temporal depthwise convolution and frequency depthwise convolution: 1) Double Depthwise Separable (DDWS) convolution and 2) BC ResNet with max feature map (MFM) activation (BC-ResMax). Using the constant Q transform (CQT) feature, parallel DDWS, sequential DDWS, and BC-ResMax models achieved EERs of 2.63%, 2.08% and 2.59% in the LA set, and achieved EERs of 0.47%, 0.63% and 0.49% in the PA set. These results are comparable with other top-performing systems in the ASVspoof 2019 competition. Moreover, parallel DDWS, sequential DDWS, and BC-ResMax models utilized only 45K, 28K, and 29K numbers of parameters which are far fewer parameters than existing models.

II. Methodology

A. Feature Engineering

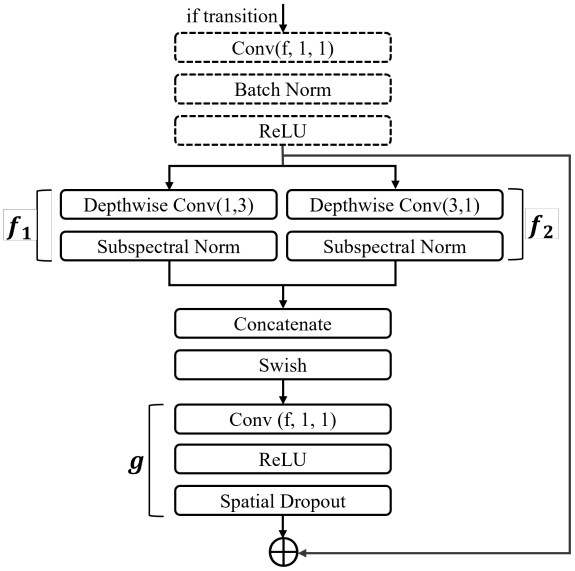
In audio analysis, rather than using a raw audio form of a 1D array, it is widely used to analyze audio by converting it into 2D spectrogram data of frequency and time. Two of the most commonly used spectral features in the ASV2019 competition are Short Time Fourier Transform (STFT) and Constant Q Transform (CQT) [6]. After a few trials and observing model accuracy, we detected that CQT was more compatible with the proposed model, indicating superior model accuracy. Therefore, the CQT feature was used to optimize the accuracy of the models .

CQT [22] searches the audio signal in its time-frequency representation by dividing the signal into shorter frames and analyzing the audio in the frequency domain. In addition, CQT is known to be more advantageous than the Fourier transform for processing audio because the frequency axis is converted into logarithmic units and the resolution is variously processed for low frequencies with low resolution and high frequencies with high resolution according to each frequency.

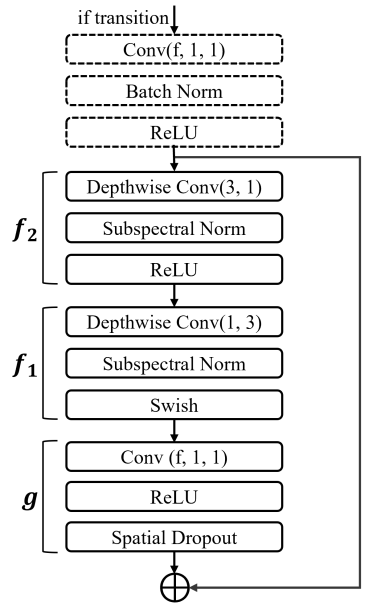
The length of the audio varies from sample to sample. Therefore, the sample length is set to 9 seconds. Samples longer than 9 seconds are used only for the first 9 seconds, and samples shorter than 9 seconds are repeated to fill 9 seconds. The librosa package [23] was used to extract CQT features, and the bin size was set to 120 and the minimum frequency was set to 1 .

B. Double Depthwise Separable Convolution

Depthwise Separable Convolution consists of two separate layers, depthwise convolution and 1×1 pointwise convolution. Depthwise convolution performs filtering per each channel of input, and then 1×1 pointwise convolution gathers the filtered feature maps by using a linear combination of the filtered feature maps[20], [24]. Therefore, input features are handled separately by considering 3D tensors in spatial and in-depth.



(a) Parallel DDWS ResBlock



(b) Sequential DDWS ResBlock

Fig. 1. DDWS ResBlocks

We introduce a variant of Depthwise Separable Convolution, which we call 'Double Depthwise Separable (DDWS) Convolution'. We use two distinct depthwise convolutional layers, temporal and frequency depthwise convolutions. The temporal depthwise convolution uses $1 \times k_1$ kernel and frequency depthwise convolution uses $k_2 \times 1$ kernel, instead of using a depthwise convolution of $k_2 \times k_1$ kernel. Then, with the following pointwise convolutional layer, DDWS convolution can build features by considering each dimension of a 3D-tensor separately.

We had two different ways of experiments of how to locate two depthwise convolutional layers. One way is to make two

streams of depthwise convolutional layers and then concatenate the outputs along the channel axis. The other way is to make a sequential order of two depthwise convolutional layers, locating frequency depthwise convolutional layer followed by temporal depthwise convolutional layer. We call the former 'Parallel DDWS' and the latter 'Sequential DDWS'.

1) Parallel DDWS ResBlock: Parallel DDWS ResBlock is defined in Figure 1(a). Let x be an input of the block. We define f_1 and f_2 to be temporal and frequency depthwise convolutions followed by Subspectral Normalization (SSN) [25], respectively. Then, we define $f^P(x) = Swish(concat(f_1(x), f_2(x)))$ where the concatenation is performed along the channel axis and Swish is a swish activation function [26]. Note that it is necessary to apply zero-padding to make the spatial dimension of outputs of f_1 and f_2 the same. We define g to be a composite of pointwise convolution, ReLU activation, and spatial dropout. Then, the Parallel DDWS ResBlock is computed as

$$y = x + g(f^P(x)).$$

We call the block which is computed as above a *normal block*. We also define a *transition block* which is applied when the input and output channel sizes are different. For the *transition block*, we use an additional pointwise convolution in the beginning of the block to change the number of channels and to keep the spatial resolution same. If we define a function h which is a composite of 1×1 convolution, batch normalization, and ReLU activation, then the *transition block* for Parallel DDWS ResBlock is computed as

$$y = h(x) + g(f^P(h(x))).$$

2) Sequential DDWS ResBlock: Sequential DDWS ResBlock is defined in Figure 1(b). This time, two depthwise convolutions are applied sequentially. We define f_1 to be a composite of temporal depthwise convolution, SSN, and a swish activation, and f_2 is defined to be a composite of frequency depthwise convolution, SSN, and ReLU activation. Then the *normal block* of Sequential DDWS Resblock is computed as

$$y = x + g(f_1(f_2(x))).$$

For the *transition block* of Sequential DDWS ResBlock, we applied the same modification, and it is computed as

$$y = h(x) + g(f_1(f_2(h(x)))).$$

C. BC-ResMax

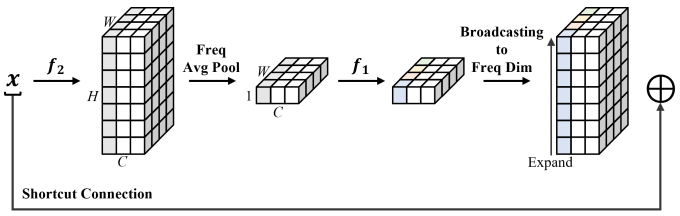


Fig. 2. Broadcasted residual block.

In this subsection, we proposed to use Broadcasted residual block with MFM (BC-ResMax block), which is a modification of Broadcasted Residual block (BC-ResNet block) [17], [19]. We combined the idea of a MFM from LCNN with BC-ResNet [27].

Figure 2 shows the structure of broadcasted residual block. Again, we consider both temporal and frequency depthwise convolutions, which are contained in the functions f_1 and f_2 in Figure 2, respectively. However, the key difference is that an average pooling is applied along the frequency axis. Thus, ignoring the channel dimension, the temporal depthwise convolution is applied on 1D temporal features. Finally, we have a broadcasting layer (BC) to reconstruct 2D features with frequency and temporal dimension.

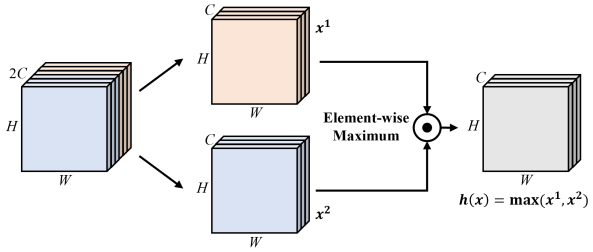


Fig. 3. Max feature map.

MFM used in LCNN is an activation function that takes the maximum value among the same location in two output feature maps with the same dimension (as shown in Fig. 3). This process improves the robustness of the model, has the effect of becoming lighter through filter selection, and is known to work well in detecting voice spoofing attacks [8], [12]. Thus, we used MFM as an activation function of a frequency depthwise convolution layer.

Figure 4 shows the structure of our proposed Broadcasted residual block with MFM (BC-ResMax block). Let x be an input of the block. We define f_2 to be a frequency depthwise convolution followed by MFM and SSN, and f_1 to be a temporal depthwise convolution followed by SSN and swish activation. We again define g to be a pointwise convolution followed by ReLU and spatial dropout. Then, the *normal block* is computed as

$$y = x + BC(g(f_1(avgpool(f_2(x))))),$$

where the average pooling and BC are performed along the frequency axis.

For the *transition block* of BC-ResMax block, we applied the same modification,

$$y = h(x) + BC(g(f_1(avgpool(f_2(h(x)))))),$$

where $h(\cdot)$ is a 1×1 convolution followed by Batch normalization and ReLU activation.

D. Model Architecture

The entire model architecture comprising of five normal blocks and four transition blocks is described in Figure 5.

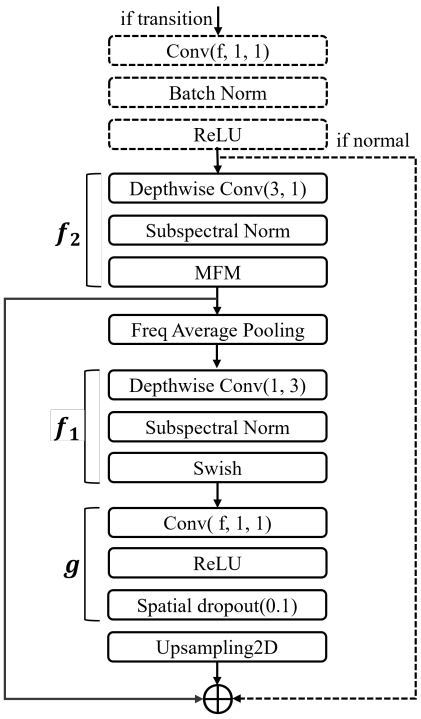


Fig. 4. BC-ResMax Block

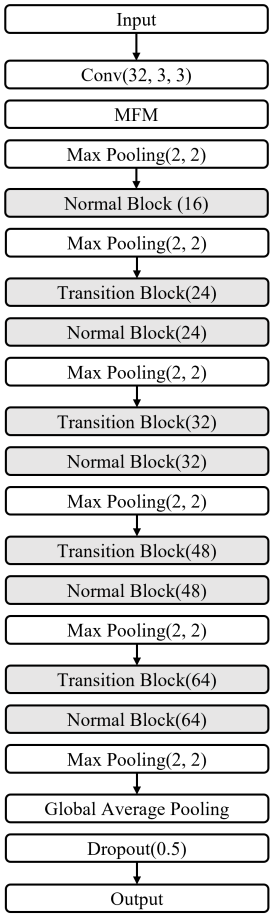


Fig. 5. Model architecture.

We define *parallel DDWS model*, *sequential DDWS model*, and *BC-ResMax* by putting normal and transition blocks of parallel DDWS ResBlocks (shown in Figure 1(a)), sequential DDWS ResBlocks (shown in Figure 1(b)), and BC-ResMax blocks (shown in Fig. 4), respectively.

The CQT feature of an audio data is an input feature of the model. An input is first applied to a convolutional layer with 32 filters followed by MFM activation which outputs 16 feature maps. Then, 2×2 max pooling, a normal block, and 2×2 max pooling is applied. Through 4 transition blocks, the number of filters is increased from 16 to 24, 32, 48, and 64, and a normal block is applied after each transition block. Each time it passes through the normal block, max pooling is applied to adjust the size of the data and to reduce computational resources. A global average pooling layer follows the last max pooling block. We then have a dropout layer, and the final layer is a dense layer with two units which outputs a result using a softmax function to model binary outcome.

III. Experiments

A. Experimental Setup

For the evaluation, we utilized the LA and PA data from ASVspoof 2019 competition [6], each of which contains a training set, a development set, and an evaluation set. For the LA set, the training, development, and evaluation sets contain $2,580$ bona fide and $22,800$ spoof utterances, $2,548$ bona fide and $22,296$ spoof utterances, and $7,355$ bona fide and $63,882$ spoof utterances, respectively. For the PA set, the training, development, and evaluation sets contain $5,400$ bona fide and $48,600$ spoof utterances, $5,400$ bona fide and $24,300$ spoof utterances, and $18,090$ bona fide and $116,640$ spoof utterances. We trained our models using only the training sets and evaluated the trained models on the development sets and evaluation sets.

To measure the detection accuracy, we primarily used equal error rate (EER), following the rules from the ASVspoof 2019 competition. The misclassified error rate for spoofing samples was referred to as the false rejection rate (FRR). The error rate for human voice samples was referred to as the false acceptance rate (FAR). Our binary classification models return the score that represents the likelihood of a voice being a spoofing sample and compute the scores for all samples in a given evaluation set. If the given score is less than the threshold, then the sample is judged as genuine. Otherwise, it is judged as a spoof. There is always a trade-off between FAR and FRR depending on a threshold value. A value at which the FAR is equal to the FRR of a given data determines the EER of the data for given model.

B. Inclusion or exclusion of MFM Activation

The LCNN model using MFM activation is proved to be effective through ASVspoof competitions in 2017 and 2019 [8], [12]. We experimented with MFM activation in

TABLE I

DEVELOPMENT EER (%) AND TEST EER (%) ON THE ASVSPOOF2019 LA AND PA DATASET. THE BASELINE MODEL INDICATES THE PARALLEL DDWS, SEQUENTIAL DDWS, AND BC-RESMAX MODEL WITHOUT ANY MFM ACTIVATIONS IN f_1 AND f_2. THE MARKS ◯ AND × DENOTE WHETHER THE MFM ACTIVATION IS INCLUDED OR NOT IN THE FUNCTIONS f_1 AND f_2, RESPECTIVELY.

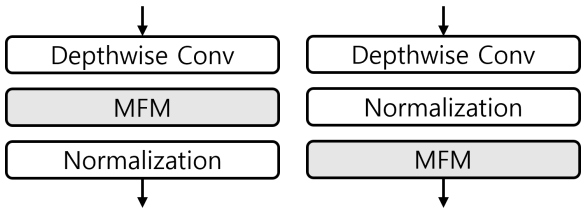
LA

Model	f_2	f_1	MFM position	Parallel DDWS		Sequential DDWS		BC-ResMax	
				EER (Dev)	EER (Eval)	EER (Dev)	EER (Eval)	EER (Dev)	EER (Eval)
baseline	×	×	-	**0.39**	2.63	**0.40**	**2.08**	0.76	2.65
Test1	◯	×	(a)	0.51	4.26	0.82	3.49	**0.67**	**2.59**
Test2	◯	×	(b)	0.44	**2.02**	0.81	2.93	**0.67**	2.85
Test3	×	◯	(a)	0.48	2.96	0.57	2.87	1.06	3.22
Test4	×	◯	(b)	0.69	4.31	0.83	4.05	1.30	3.50
Test5	◯	◯	(a)	0.51	2.22	0.63	3.31	1.17	2.79
Test6	◯	◯	(b)	0.52	2.68	0.54	3.30	1.01	3.86

PA

Model	f_2	f_1	MFM position	Parallel DDWS		Sequential DDWS		BC-ResMax	
				EER (Dev)	EER (Eval)	EER (Dev)	EER (Eval)	EER (Dev)	EER (Eval)
baseline	×	×	-	0.24	**0.47**	**0.34**	**0.63**	0.35	0.73
Test1	◯	×	(a)	0.34	0.56	0.55	0.79	0.29	0.54
Test2	◯	×	(b)	0.29	0.56	0.52	0.96	**0.21**	**0.49**
Test3	×	◯	(a)	0.43	0.73	0.72	0.94	0.44	0.68
Test4	×	◯	(b)	0.37	0.65	0.76	1.20	0.45	0.67
Test5	◯	◯	(a)	0.23	0.59	0.53	0.77	0.32	0.55
Test6	◯	◯	(b)	**0.19**	0.60	0.54	0.93	0.28	0.72

our proposed block designs. In each block, two depthwise convolutions are applied through the functions f_1 and f_2. The function f_1 contains a temporal depthwise convolution followed by normalization and an activation function. The function f_2 contains a frequency depthwise convolution followed by normalization and an activation function. Three types of experiments were conducted in which MFM activation is used only in f_1 or only in f_2, or both. In addition, the position of the MFM layer was tested by comparing the two cases; before and after normalization, as shown in Figure 6 (a) and (b). By evaluating and comparing a total of 6 cases, the best-performing model was proposed.



(a) Before Normalization (b) After Normalization

Fig. 6. MFM position.

Table I shows the performance when MFM layer is used as the activation of each depthwise separable convolution layer. The marks ◯ and × in columns f_1 and f_2 indicate whether MFM activation is included or excluded in the corresponding functions of a model. For Parallel DDWS, the conditions for achieving the best performance change every time, making it difficult to determine the best case. Thus, we decide not to use MFM with the Parallel DDWS model. The Sequential DDWS model without MFM achieved the best performance for all development and evaluation sets in LA and PA data. For the BC-ResMax model, MFM activation in f_2 achieved the best-performing results in all four cases (development and evaluation sets in LA and PA data). In three of the four cases, the MFM performed better when it came after the Normalization layer. Thus, our default BC-ResMax model has MFM activation in f_2 as shown in Figure 4.

C. Experimental Results

In Table II, we compared the performance of our proposed model with the top 5 performance, including multi-model and single model, for EER. The EER results are compared against the top five models from each dataset; models are sorted based on the EER in descending order. #Mo describes the number of models used in an ensemble system. Two top-performing single model systems are also shown at the end of each table in *italic*. #Params represent the number of parameters contained in the models. Results that are not public are denoted as hyphens. As shown in the table, most of our proposed models outperform other single models based on EER for both LA and PA evaluation sets. However, the T28 model performed better than the Sequential DDWS model in the PA data.

In the LA set, our proposed Sequential DDWS, Parallel DDWS, and BC-ResMax models rank 3∼5th among all solutions. Sequential DDWS model was the best with an EER of 2.08%, followed by BC-ResMax with 2.59% and Parallel DDWS with 2.63%. In the PA set, two models excluding Sequential DDWS performed second place among the total solutions, and the Sequential DDWS model ranked 6th with an EER of 0.63%. Parallel DDWS model was the best with 0.47%, and the BC-ResMax model had the next best performance with 0.49%.

The top five models with excellent performance achieved high performance using an ensemble solution using multiple

TABLE II

OUR MODELS PERFORMANCE ON THE ASVSPOOF 2019 DEVELOPMENT SETS AND EVALUATION SETS. MODELS ARE SORTED BASED ON THE EER IN A DESCENDING ORDER. OUR PROPOSED METHODS ARE IN **BOLD**. SYSTEMS THAT USE A SINGLE MODEL ARE IN *italic*. #MO DENOTE THE NUMBER OF MODELS USED IN AN ENSEMBLE SYSTEM. #PARAMS REPRESENTS THE NUMBER OF PARAMETERS USED IN THE MODELS.

LA

#	*Model*	*EER (Dev)*	*EER (Eval)*	*#Mo*	*# Params*
1	T05	-	0.22	-	-
2	T45 [12]	0.00	1.86	5	1484K
3	***Sequential DDWS***	0.40	2.08	1	28K
4	***BC-ResMax***	0.67	2.59	1	29K
5	***Parallel DDWS***	0.39	2.63	1	45K
6	T60 [28]	0.00	2.64	4	-
7	T24	-	3.45	-	-
8	T50	0.90	3.56	-	-
	T45 (FFT-LCNN)	0.04	4.53	1	371K
	T45 (LFCC-LCNN)	0.16	5.06	1	371K

PA

#	*Model*	*EER (Dev)*	*EER (Eval)*	*#Mo*	*# Params*
1	T28	-	0.39	-	-
2	***Parallel DDWS***	0.24	0.47	1	45K
3	***BC-ResMax***	0.21	0.49	1	29K
4	T45 [12]	0.02	0.54	3	1113K
5	T44 [29]	0.13	0.59	5	5811K
6	***Sequential DDWS***	0.34	0.63	1	28K
7	T10 [30]	0.24	0.66	6	1330K
8	T24	-	0.77	-	-
	T28	-	0.50	1	-
	T45 (CQT-LCNN)	0.80	1.23	1	371K
	T44 (logspec-SENet)	0.58	1.29	1	1344K

models, while our proposed models all used a single model. In terms of the number of model parameters, it uses a remarkably small number of parameters compared to other excellent models. It achieves the purpose of reducing model complexity and reducing the amount of computation. Also, when compared to other single models, the performance is excellent, and the number of parameters is much smaller, so the evaluation results in LA and PA sets show competitively high performance.

IV. DISCUSSION

A. Voice spoofing detection in noisy environment

Recently, the Audio Deep Synthesis Detection challenge (ADD) 2022 was hosted as a signal processing grand challenge in the International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2022. Track 1 is to build a model that detects fake audio in noisy environments such as background music effects and real-world noises. The fake audio used in the ADD competition was generated by speech synthesis and voice conversion, similar to the LA scenario from ASVspoof 2019. We participated in the ADD competition, experimenting with our proposed DDWS and BC-ResMax with other well-performing models. Our ensemble system achieved 23.8% of EER, ranking 3rd among all 42 submitted systems. Our ensemble model comprises five systems, Sequential DDWS, BC-ResMax, LCNN [12], ResMax [31], and one more manually derived model. This result means that our proposed DDWS and BC-ResMax models not only have excellent performance but also work well in noisy environments.

B. Computational cost with temporal and frequency depthwise convolution design

Depthwise separable convolution significantly reduces the computational cost compared to the existing convolution [20], [24]. Using temporal and frequency depthwise convolution can further reduce the cost of the depthwise convolution part.

Assume that we have a feature map of the size $H \times W \times C_1$, and we want to find the computational cost when a convolutional layer with C_2 number of filters of the kernel size $k_1 \times k_2$. We consider the case when zero-padding is applied so that the spatial dimension remains the same. For a standard convolution, the computational cost will be $k_1 \cdot k_2 \cdot C_1 \cdot H \cdot W \cdot C_2$.

For a depthwise separable convolution, the cost of a depthwise convolution is $k_1 \cdot k_2 \cdot C_1 \cdot H \cdot W$ and the cost of a pointwise convolution is $1 \cdot 1 \cdot C_1 \cdot H \cdot W \cdot C_2$, so that the total cost will be $k_1 \cdot k_2 \cdot C_1 \cdot H \cdot W + C_1 \cdot H \cdot W \cdot C_2$.

For our DDWS convolution, the computational cost of frequency depthwise convolution is $k_1 \cdot 1 \cdot C_1 \cdot H \cdot W$, and the cost of temporal depthwise convolution is $1 \cdot k_2 \cdot C_1 \cdot H \cdot W$. Including a pointwise convolution, the total cost of DDWS convolution will be $(k_1 + k_2) \cdot C_1 \cdot H \cdot W + C_1 \cdot H \cdot W \cdot C_2$.

Therefore, if we use a depthwise separable convolution or our DDWS convolution, we get a reduction in computation of

$$\frac{1}{C_2} + \frac{1}{k_1 \cdot k_2} \quad or \quad \frac{1}{k_2 \cdot C_2} + \frac{1}{k_1 \cdot C_2} + \frac{1}{k_1 \cdot k_2},$$

compared to standard convolutions, respectively. In our models, we set $k_1 = k_2 = 3$ which reduces about 9 times less computation than standard convolutions.

V. CONCLUSION

Most of the spectrogram feature-based voice spoofing detection systems are not considering frequency information very well. Also, solutions should be designed to consider real-world model complexity and detection latency requirements. Our parallel DDWS, sequential DDWS, and BC-ResMax models are designed to process temporal and frequency-wise features separately by considering temporal depthwise convolution and frequency depthwise convolution. Moreover, the depthwise convolution design itself is very light, significantly decreasing the number of parameters used in the models. Our parallel DDWS, sequential DDWS, and BC-ResMax models used only a single deep learning model with far fewer model parameters. Each of these models achieved spoofing attack detection EER of 2.63%, 2.08% and 2.59% in the LA set, and 0.63%, 0.47% and 0.49% in the PA set, achieving comparable performance with other top ensemble systems from the 2019 ASVspoof competition. Furthermore, these models used only 45K, 28K, and 29K parameters, respectively. It is a far fewer number of parameters than existing models.

ACKNOWLEDGMENT

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).

REFERENCES

[1] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurou, "Tacotron: Towards end-to-end speech synthesis," in *Proc. Interspeech 2017*. Stockholm: ISCA, 2017, pp. 4006–4010.

[2] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," in *6th International Conference on Learning Representations*. Vancouver: ICLR, 2018.

[3] 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.

[4] 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.

[5] 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.

[6] 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. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2249

[7] J. Yamagishi, X. Wang, M. Todisco, M. Sahidullah, J. Patino, A. Nautsch, X. Liu, K. A. Lee, T. Kinnunen, N. Evans, and H. Delgado, "ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection," in *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, 2021, pp. 47–54.

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

[9] W. Cai, H. Wu, D. Cai, and M. Li, "The DKU Replay Detection System for the ASVspoof 2019 Challenge: On Data Augmentation, Feature Representation, Classification, and Fusion," in *Proc. Interspeech 2019*, 2019, pp. 1023–1027.

[10] 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.

[11] J. Williams and J. Rownicka, "Speech Replay Detection with x-Vector Attack Embeddings and Spectral Features," in *Proc. Interspeech 2019*, 2019, pp. 1053–1057.

[12] 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. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1768

[15] 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.

[13] A. Tomilov, A. Svishchev, M. Volkova, A. Chirkovskiy, A. Kondratev, and G. Lavrentyeva, "STC Antispoofing Systems for the ASVspoof2021 Challenge," in *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, 2021, pp. 61–67.

[14] J. Pons, T. Lidy, and X. Serra, "Experimenting with musically motivated convolutional neural networks," in *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2016, pp. 1–6.

[16] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing acoustic scene classification models with CNN variants," DCASE2020 Challenge, Tech. Rep., June 2020.

[17] 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.

[18] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The receptive field as a regularizer in deep convolutional neural networks for acoustic scene classification," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[19] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," *arXiv preprint arXiv:2106.04140*, 2021.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1800–1807.

[25] S. Chang, H. Park, J. Cho, H. Park, S. Yun, and K. Hwang, "Subspectral normalization for neural audio data processing," 2021.

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

[27] 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.

[28] 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. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2505

[29] 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. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1794

[30] W. Cai, H. Wu, D. Cai, and M. Li, "The DKU Replay Detection System for the ASVspoof 2019 Challenge: On Data Augmentation, Feature Representation, Classification, and Fusion," in *Proc. Interspeech 2019*, 2019, pp. 1023–1027. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1230

[31] 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.