Detecting Spoofed Noisy Speeches via Activation-Based Residual Blocks for Embedded Systems

Jinyu Zhan, *Member, IEEE*, Suidi Peng, Wei Jiang, *Senior Member, IEEE*, Xiang Wang, and Jiarui Liu

***Abstract*—Spoofed noisy speeches seriously threaten the speech-based embedded systems, such as smartphones and intelligent assistants. Consequently, we present an anti-spoofing detection model with activation-based residual blocks to identify spoofed noisy speeches with the requirements of high accuracy and low time overhead. Through theoretic analysis of noise propagation on shortcut connections of traditional residual blocks, we observe that different activation functions can help reducing the influence of noise under certain situations. Then, we propose a feature-aware activation function to weaken the influence of noise and enhance the anti-spoofing features on shortcut connections, in which a fine-grained processing is designed to remove noise and strengthen significant features. We also propose a variance-increasing-based optimization algorithm to find the optimal hyperparameters of the feature-aware activation function. Benchmark-based experiments demonstrate that the proposed method can reduce the average equal error rate of anti-spoofing detection from 21.72% to 4.51% and improve the accuracy by up to 37.06% and save up to 91.26% of time overhead on Jetson AGX Xavier compared with ten state-of-the-art methods.**

***Index Terms*—Anti-spoofing detection, embedded speech recognition, feature-aware activation, residual blocks, spoofed noisy speeches.**

I. Introduction

HUMAN voices have been widely used in biometric authentication, facilitating the replacement of traditional passwords in human–machine interaction embedded systems, such as smartphones and intelligent assistants. Unfortunately, speech-based authentication systems are vulnerable to malicious spoofing attacks [1], such as voice conversion [2] and text-to-speech attacks [3], [4], [5]. With the development of deep learning techniques, speech synthesis methods based on deep neural networks, recurrent neural networks, and sequence-to-sequence networks have been proposed to improve the performance of spoofed speeches [6]. Moreover, neural vocoders, such as WaveNet [7] and parallel WaveGAN [8], are further applied to voice conversion and text-to-speech, increasing the difficulty of detecting spoofed speeches. Unfortunately, pure-speech-oriented anti-spoofing detection methods will become ineffective, especially when attackers introduce background sounds, such as additive noise and reverberation into spoofed speeches [9], [10]. Therefore, spoofed noisy speeches have emerged as a great threat to speech-based authentication systems.

Many researchers have made efforts to deal with noisy speeches in multiple fields. Lv et al. [11] constructed a multitask learning framework for both denoising and keyword spotting to discriminate keywords in noisy environments. Martel et al. [12] used progressive learning to perform audio-visual speech separation in noisy environments. Based on Whisper [13], whisper-AT [14] recognized noisy speeches and cost less than 1% extra computational overheads. Kim et al. [15] proposed extended U-Net to optimize the structural limitations of U-Net [16] for speaker verification tasks in noisy environments. Unfortunately, studies [17], [18], [19], [20], [21], [22], [23], [24], [25] on anti-spoofing detection focused on pure speeches, though the ASVspoof challenge series [26], [27], [28], [29] have provided the datasets with relevant audio and anti-spoofing resources. These methods have a significant performance degradation in detecting spoofed noisy speeches. An intuitive approach is to combine the denoising methods with anti-spoofing detection methods to detect spoofed noisy speeches. Specifically, conventional anti-spoofing detection methods can be directly leveraged after removing the noises from speeches. However, denoising methods might modify or remove certain anti-spoofing features of noisy speeches, leading to extremely low detection accuracy. Furthermore, inputting spoofed noisy speeches separately into denoising methods and anti-spoofing detection methods will definitely result in high time overheads, hindering the deployment in embedded systems. Thus, how to detect spoofed noisy speeches with low time overhead and high accuracy is a challenging work.

As a supplementary work, we propose an anti-spoofing detection model with activation-based residual blocks to identify spoofed noisy speeches. We formulate the noise propagation model and evaluate the impact on the outputs of the traditional residual blocks. By analyzing the influence of different activation function on shortcut connection of residual blocks, we consider to improve the residual blocks with a fine-grained activation function to deal with spoofed noisy

Manuscript received 29 July 2024; accepted 29 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62072076 and Grant 62372087. This article was presented at the International Conference on Hardware/Software Codesign and System Synthesis (CODES + ISSS) 2024 and appeared as part of the ESWEEK-TCAD Special Issue. This article was recommended by Associate Editor S. Dailey. *(Corresponding author: Jinyu Zhan.)*

The authors are with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China (e-mail: zhanjy@uestc.edu.cn).

Digital Object Identifier 10.1109/TCAD.2024.3437331

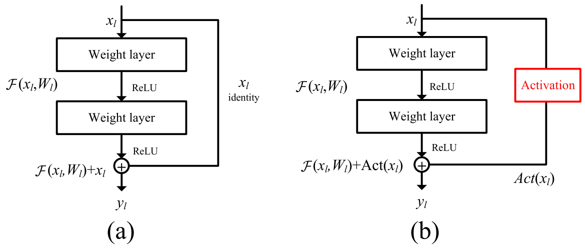

Fig. 1. (a) Traditional residual block versus (b) improved residual block.

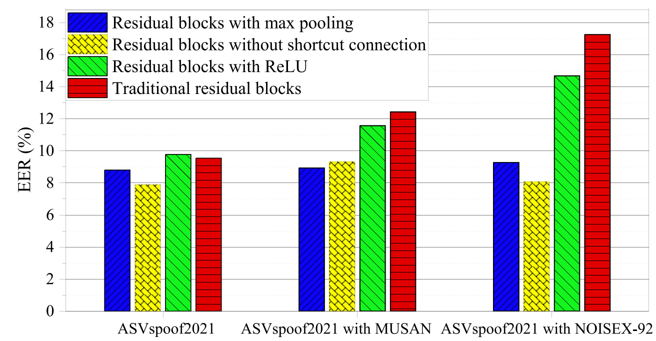

Fig. 2. EER comparison of traditional and improved residual blocks on ASVspoof2021 and ASVspoof2021 with MUSAN, and ASVspoof2021 with NOISEX-92.

speeches, in which the inputs are partitioned into significant, uncertain, or noisy features by comparing with their surrounding values. We also devise a variance-increasing-based optimization (VIO) algorithm to find the optimal hyperparameters of the feature-aware activation function. The main contributions are listed as follows.

1) We propose an anti-spoofing detection method based on activation-based residual blocks for the speech-based authentication embedded systems, which can identify the spoofed noisy speeches with high accuracy and low time overhead.
2) We design a feature-aware activation function on shortcut connection of the residual block, which is a piecewise function to enhance anti-spoofing features and suppress noise.
3) We present a VIO algorithm to obtain the optimal hyperparameters of the feature-aware activation function.
4) Benchmark-based experiments are conducted to evaluate the efficiency of the proposed method on datasets of ASVspoof 2021 and NOISEX-92 corpus. In specific, the proposed method achieves quick and accurate anti-spoofing detection, which is very suitable for embedded devices.

II. Motivation

Existing anti-spoofing detection methods usually utilize residual blocks to construct neural networks. In traditional residual networks, the residual block is generally constructed as in Fig. 1(a), i.e., $y_l = \mathcal{F}(x_l, W_l) + x_l$, where x_l and y_l are the input and output of the lth residual block, and W_l is the weights and biases associated with the lth residual block. $\mathcal{F}$ is the function of weight layers which include two $[3 \times 3]$ convolutions, ReLU activation, and Batch Normalization. He et al. [30] found that x_l achieves the lowest training loss and fastest error reduction among all variants, whereas shortcut connections of scaling, gating, and 1×1 convolutions all lead to higher training losses and errors.

During detecting noisy speeches, noise as a part of the speeches directly affects the outputs through the shortcut connection, since both $\mathcal{F}(x_l, W_l)$ and x_l have effect on y_l. The features of noise can directly disturb the anti-spoofing detection, leading to a decrease in accuracy. Therefore, we explore an activation function based on traditional residual block, as shown in Fig. 1(b), to enhance the anti-spoofing features and suppress noise on the shortcut connection. The improved residual block can be expressed as

$$y_l = \mathcal{F}(x_l, W_l) + \mathrm{Act}(x_l) \tag{1}$$

where Act($\cdot$) denotes the activation function.

We try to evaluate the impact of different activation functions on shortcut connections. The experiments are conducted on both ASVspoof2021 dataset and ASVspoof2021 dataset with noise (including NOISEX-92 [31] and MUSAN [32]) at the SNR of 10 dB (decibels). First, we add a max-pooling function on the shortcut connection to simulate enhancing the anti-spoofing features. Second, we remove the shortcut connection to simulate suppressing noise, replacing the residual block with the plain block. Third, we use a ReLU activation function on the shortcut connection to simulate restricting the negative values. The experimental results of equal error rate (EER) are shown in Fig. 2. We can observe that the EERs of three improved residual blocks are lower than that of the traditional residual block on ASVspoof2021 dataset with noise from NOISEX-92. The EERs of residual blocks with max pooling, without shortcut connections, and with ReLU are only 9.00%, 8.06%, and 14.68%, whereas that of the traditional residual block is 17.25%. On ASVspoof2021 dataset with MUSAN, the EER of the traditional residual blocks is 12.43% while those of residual blocks with max pooling, without shortcut connections, and with ReLU are 8.93%, 9.31%, and 11.57%. The experimental results suggest that both enhancing anti-spoofing features and suppressing noise on the shortcut connection can improve the anti-spoofing detection performance of noisy speeches.

Therefore, it is promising to propose an improved residual block with a fine-grained activation function to suppress noise and enhance anti-spoofing features. To this end, we need to address the following two problems.

1) How to model the propagation of noise and evaluate its impact on the outputs of the traditional residual block?
2) How to design an activation function to weaken the influence of noise and enhance the anti-spoofing features?

III. Noise Propagation Formulation

To address the first problem, this section formulates noise propagation in one residual block. Then, we analyze three possible methods to improve the robustness of a traditional residual block, assuming that noise is smaller than pure speeches and will not change pure values dramatically.

A. Noise Propagation Formulation in One Residual Block

For each residual block as shown in Fig. 1(a), its operations are mainly composed of correlation calculation and identity mapping in the shortcut connection. The layer calculation model in a residual block can be defined as follows. Given $\hat{x}_l$ as the pure input of the lth residual block, the pure output $\hat{y}_l$ of the lth block can be formulated as

$$\hat{y}_l = W_l \odot \hat{x}_l + \hat{x}_l \tag{2}$$

where $\odot$ is the Hadamard (or elementwise) product. W_l is the weight matrix calculated by the lth weight layers which includes two convolutions and ReLU activation. Therefore, W_l is no less than zero.

Assuming the noisy input of the lth residual block as x_l, the noisy output y_l of the lth residual block can be formulated as

$$y_l = W_l \odot x_l + x_l. \tag{3}$$

We use $x_l = \hat{x}_l + \epsilon_l$ to replace $\hat{x}_l$, where ϵ_l is the perturbed value caused by the noise. Then, the noisy output y_l of the lth residual block can be expressed as

$$y_l = W_l \odot \hat{x}_l + W_l \odot \epsilon_l + \hat{x}_l + \epsilon_l. \tag{4}$$

θ_l is used to denote the difference between the pure output and noisy output of the lth residual block. According to (2) and (4), θ_l can be formulated as

$$\theta_l = |y_l - \hat{y}_l| = |W_l \odot \epsilon_l + \epsilon_l| = |(W_l + 1) \odot \epsilon_l|. \tag{5}$$

We can observe that θ_l grows as W_l increases in traditional residual blocks, making noise perturbation more influential. Now we put an $f(\cdot)$ activation function on the shortcut connection, then, (5) will be converted into

$$\theta_l^f = |y_l - \hat{y}_l| = |W_l \odot \epsilon_l + f\left(\hat{x}_l + \epsilon_l\right) - f\left(\hat{x}_l\right)|. \tag{6}$$

Regarding to (6), we analyze the change of θ_l to lower the influence of noise by two cases of the function $f(\cdot)$.

Case 1: Keeping the noisy input x_l unchanged, function $f(\cdot)$ increases the ratio of pure feature $\hat{x}_l$ to noise ϵ_l.

Case 2: Keeping the ratio of $\hat{x}_l$ to ϵ_l unchanged, function $f(\cdot)$ decreases both the noisy input x_l and pure input $\hat{x}_l$ in the shortcut connection.

In case 1, $f(\hat{x}_l + \epsilon_l)$ approximately equals to $f(\hat{x}_l)$ in the limit situation, and θ_l^f will be converted into $|W_l \odot \epsilon_l|$, which is definitely less than θ_l. Thus, in case 1, function $f(\cdot)$ reduces the noise impact by increasing pure features. In case 2, both $f(x_l)$ and $f(\hat{x}_l)$ are equal to zero in the limit situation and θ_l^f will be converted into $|W_l \odot \epsilon_l|$, which is also less than θ_l. Function $f(\cdot)$ removes pure features in exchange for blocking the noise. When the ratio of pure features $\hat{x}_l$ is high, function $f(\cdot)$ in case 1 should be applied. On the contrary, when the noise in x_l is large, function $f(\cdot)$ in case 2 needs to be used. Accordingly, a clear boundary between enhancing and suppressing is needed.

The following sections will discuss the satisfaction of three activation methods for either of the two cases, which could be helpful for noise processing.

B. Feasibility of Replacing Residual Blocks With Plain Blocks

Removing the shortcut connection, a traditional residual block is turned into a plain block, which prevents all information from propagating through the shortcut connection. We use θ_l^P to denote the difference between the pure output and noisy output of the lth plain block, which can be formulated as

$$\theta_l^P = |y_l^p - \hat{y}_l^p| = |W_l \odot \epsilon_l|. \tag{7}$$

θ_l^P is definitely less than θ_l since W_l is positive. Therefore, replacing residual blocks with plain blocks is feasible to lower the influence of noise, which satisfies case 2.

C. Feasibility of ReLU Activation

Now, we put a ReLU activation function on the shortcut connection of a traditional residual block to discard negative outputs. θ_l^R is defined to denote the difference between the pure output and noisy output of the lth residual block with ReLU, which can be formulated as

$$\theta_l^R = \left|y_l^R - \hat{y}_l^R\right| = \left|W_l \odot \epsilon_l + \sigma\left(\hat{x}_l + \epsilon_l\right) - \sigma\left(\hat{x}_l\right)\right| \tag{8}$$

where $\sigma(\cdot)$ is the ReLU activation function.

We can compare θ_l^R with θ_l under four situations as follows.

1) *$\hat{x}_l + \epsilon_l \geq 0$, $\hat{x}_l \geq 0$:* In this situation, $\sigma(\hat{x}_l + \epsilon_l) - \sigma(\hat{x}_l) = \hat{x}_l + \epsilon_l - \hat{x}_l = \epsilon_l$. Thus, $\theta_l^R = |W_l \odot \epsilon_l + \epsilon_l|$, meaning that $\theta_l^R = \theta_l$.
2) *$\hat{x}_l + \epsilon_l \geq 0$, $\hat{x}_l < 0$:* In this situation, $\sigma(\hat{x}_l + \epsilon_l) - \sigma(\hat{x}_l) = \hat{x}_l + \epsilon_l$. Since $0 < -\hat{x}_l \leq \epsilon_l$, $0 \leq \hat{x}_l + \epsilon_l < \epsilon_l$. Then, $\theta_l^R = |W_l \odot \epsilon_l + \hat{x}_l + \epsilon_l|$ and $\theta_l = |W_l \odot \epsilon_l + \epsilon_l|$. Thus, $\theta_l^R < \theta_l$.
3) *$\hat{x}_l + \epsilon_l < 0$, $\hat{x}_l \geq 0$:* In this situation, $\sigma(\hat{x}_l + \epsilon_l) - \sigma(\hat{x}_l) = -\hat{x}_l$. $\theta_l^R = |W_l \odot \epsilon_l - \hat{x}_l|$, whereas $\theta_l = |W_l \odot \epsilon_l + \epsilon_l|$. Since $\hat{x}_l + \epsilon_l < 0$ and $\hat{x}_l \geq 0$, $\epsilon_l < -\hat{x}_l \leq 0$. Thus, $\theta_l^R < \theta_l$.
4) *$\hat{x}_l + \epsilon_l < 0$, $\hat{x}_l < 0$:* In this situation, $\sigma(\hat{x}_l + \epsilon_l) - \sigma(\hat{x}_l) = 0$. Thus, $\theta_l^R = |W_l \odot \epsilon_l|$, meaning that $\theta_l^R \leq \theta_l$.

To sum up, $\theta_l^R \leq \theta_l$ in four situations. ReLU activation used in the shortcut connection can reduce the influence of noise. It remains the positive values as useful features and removes negative values as noise, which has a boundary and partly satisfies case 2.

D. Feasibility of Max Pooling

We put a max-pooling function on the shortcut connection of a traditional residual block. Max pooling increases every element to the maximum value of the current receptive field.

Noise ϵ is introduced to any pure input $\hat{x}_{i,j}^S$ in a region S which contains the same values after max pooling with the shape of $[h \times w]$, i.e., $x_{i,j}^S = \hat{x}_{i,j}^S \pm |\epsilon|$. We use $r_{i,j} = (|\epsilon|/|\hat{x}_{i,j}^S|)$ to denote the ratio of noise ϵ to pure speech $\hat{x}_{i,j}^S$. Therefore, we define $D(i,j)$ to denote the difference between the pure speech and noise in the shortcut connection of traditional residual block, which can be expressed as

$$D(i,j) = \left|\hat{x}_{i,j}^S\right| - \left|r_{i,j} \cdot \hat{x}_{i,j}^S\right|. \tag{9}$$

Similarly, we can obtain the difference $D^M(i,j)$ between the pure speech and noise in shortcut connection of the residual block with max pooling, which is expressed as

$$D^M(i,j) = |\hat{x}^S_{b,c}| - |r_{i,j} \cdot \hat{x}^S_{b,c}| \tag{10}$$

where $x^S_{b,c}$ is the maximum of the noisy speech in S. (b,c) is the position of $x^S_{b,c}$ in S. $\hat{x}^S_{b,c}$ is the corresponding pure speech of $x^S_{b,c}$.

Then, we compute the average difference in S, which can be formulated as

$$\begin{aligned}\Delta\bar{D} &= \frac{1}{wh} \cdot \sum_{(i,j)\in S} \left(D^M(i,j) - D(i,j)\right) \\ &= \frac{1}{wh} \cdot \sum_{(i,j)\in S} \left(1 - r_{i,j}\right)\left(\left|\hat{x}^S_{b,c}\right| - \left|\hat{x}^S_{i,j}\right|\right).\end{aligned} \tag{11}$$

Therefore, $\Delta\bar{D} \geq 0$ demonstrates that max pooling increases the ratio of pure speech $\hat{x}^S_{i,j}$ to noise ϵ and lowers the influence of noise, which satisfies case 1. Otherwise, $\Delta\bar{D} < 0$ demonstrates that max pooling decreases the ratio of pure speech $\hat{x}^S_{i,j}$ to noise ϵ, which dissatisfies case 1. Then, we analyze $\Delta\bar{D}$ under three situations as follows.

1) *All $\hat{x}^S_{i,j} \geq 0$:* In this situation, $\Delta\bar{D} = (1/wh)\cdot\sum_{(i,j)\in S}(1-r_{i,j})(\hat{x}^S_{b,c} - \hat{x}^S_{i,j})$. In most cases, noise does not change the position of the maximum element, i.e., $\hat{x}^S_{b,c}$ is the maximum element of the pure speech in S. Then, $\hat{x}^S_{b,c} - \hat{x}^S_{i,j} \geq 0$. Thus, $\Delta\bar{D} \geq 0$, meaning that max pooling lowers the influence of noise. If noise changes the position of the maximum element (in rare cases), we can sort the elements of S from the smallest to the largest and store them into a vector V. Assuming $\hat{x}^S_{b,c}$ is the qth element in the vector V (i.e., $\hat{x}^S_{b,c} = \hat{x}^V_q$), $\Delta\bar{D}$ in (11) can be converted into

$$\begin{aligned}\Delta\bar{D} = \frac{1}{wh} \cdot &\left[\sum_{m=1}^{q-1}\left(1 - \frac{|\epsilon|}{\hat{x}^V_m}\right)\left(\hat{x}^V_q - \hat{x}^V_m\right)\right. \\ &\left. - \sum_{n=q}^{wh}\left(1 - \frac{|\epsilon|}{\hat{x}^V_n}\right)\left(\hat{x}^V_n - \hat{x}^V_q\right)\right]\end{aligned} \tag{12}$$

where $\hat{x}^V_m \leq \hat{x}^V_q < \hat{x}^V_n$. The minimal $\Delta\bar{D}$ occurs when $q = 1$. Then, (12) is converted into

$$\Delta\bar{D} = -\frac{1}{wh} \cdot \sum_{n=1}^{wh}\left(1 - \frac{|\epsilon|}{\hat{x}^V_n}\right)\left(\hat{x}^V_n - \hat{x}^V_1\right). \tag{13}$$

At this time, since $x^V_1 = x^S_{b,c}$ is the maximum element of noisy speech and $\hat{x}^V_1 = \hat{x}^S_{b,c}$ is the minimum element of pure speech, $\hat{x}^V_2 - |\epsilon| < \cdots < \hat{x}^V_{wh} - |\epsilon| < \hat{x}^V_1 + |\epsilon|$. Accordingly, the range of $(\hat{x}^V_n - \hat{x}^V_1)$ in (13) is less than $2|\epsilon|$. Consequently, $\Delta\bar{D}$ is more than $-2|\epsilon|$, which is a negative value approaching 0. Therefore, although max pooling shows a little weakness, it can be ignored.

2) *All $\hat{x}^S_{i,j} < 0$:* In this situation, $\Delta\bar{D} = (1/wh)\cdot\sum_{(i,j)\in S}(1-r_{i,j})(\hat{x}^S_{i,j} - \hat{x}^S_{b,c})$. Obviously, $\Delta\bar{D}$ under all $\hat{x}^S_{i,j} < 0$ is completely opposite to that under all $\hat{x}^S_{i,j} \geq 0$, meaning that max pooling dissatisfies case 1 and does not lower the influence of noise.

3) *Some $\hat{x}^S_{i,j} \geq 0$ Whereas Others $\hat{x}^S_{i,j} < 0$:* In this situation, we can sort the elements of S from the smallest to the largest and store them into a vector V. Assuming $\hat{x}^V_t < 0$ and $\hat{x}^V_{t+1} \geq 0$, $\Delta\bar{D}$ can be expressed as

$$\begin{aligned}\Delta\bar{D} = \frac{1}{wh}&\left[\sum_{m=t+1}^{wh}\left(1 - \frac{|\epsilon|}{\hat{x}^V_m}\right)\left(\hat{x}^V_q - \hat{x}^V_m\right)\right. \\ &\left. + \sum_{n=1}^{t}\left(1 - \frac{|\epsilon|}{|\hat{x}^V_n|}\right)\left(\hat{x}^V_q - |\hat{x}^V_n|\right)\right].\end{aligned} \tag{14}$$

According to (14), it can be observed that $\sum_{m=t+1}^{wh}(1 - [|\epsilon|/\hat{x}^V_m])(\hat{x}^V_q - \hat{x}^V_m)$ is the same as $\Delta\bar{D}$ under all $\hat{x}^S_{i,j} \geq 0$. Moreover, when $|\hat{x}^V_n| \leq \hat{x}^V_q$, $\sum_{n=1}^{t}(1 - [|\epsilon|/|\hat{x}^V_n|])(\hat{x}^V_q - |\hat{x}^V_n|)$ is also the same as $\Delta\bar{D}$ under all $\hat{x}^S_{i,j} \geq 0$. On the contrary, when $|\hat{x}^V_n| > \hat{x}^V_q$, $\sum_{n=1}^{t}(1 - [|\epsilon|/|\hat{x}^V_n|])(\hat{x}^V_q - |\hat{x}^V_n|)$ is the same as $\Delta\bar{D}$ under all $\hat{x}^S_{i,j} < 0$.

In summary, when inputs are positive values, max pooling can lower the influence of noise by increasing the ratio of pure speeches, which satisfies case 1.

IV. Constructing Residual Networks With Feature-Aware Activation

To address the second problem mentioned in the Motivation (Section II), we design a feature-aware activation function for residual blocks to detect spoofed noisy speeches. Then, we construct a network model via improved residual blocks.

A. Design of Feature-Aware Activation

To detect spoofed noisy speeches, we make efforts to reduce the noise impact through the shortcut connection of residual blocks by the following four requirements.

- Rq. 1: Significant features are very likely to belong to pure speeches that include anti-spoofing features and should be enhanced. Due to the fact that the outputs of the weight layers cannot be overwhelmed by those of the shortcut connection, significant features cannot be infinitely enhanced.
- Rq. 2: Insignificant features have a high possibility of being noise and should be removed.
- Rq. 3: If an input is unable to determine whether it is an anti-spoofing feature or noise, it should be suppressed.
- Rq. 4: There needs to be a boundary between enhancing significant features and reducing noise.

By satisfying these four requirements, the information from the shortcut connection contains features that are either important, low noise, or even no noise. Even if only a small amount of information can be passed through the shortcut connection, it is significant and useful.

According to the discussions of the plain block, ReLU, and max pooling in Section III, we can conclude that each of them only satisfies some of these four requirements. The plain block satisfies Rq. 2 and Rq. 3, which regards information on the shortcut connection as noisy or uncertain features to remove or suppress. However, the plain block dissatisfies Rq. 1 and Rq. 4. Meanwhile, ReLU activation keeps the positive values as useful features and removes negative values as noise, which satisfies Rq. 2 and Rq. 4, and yet dissatisfies Rq. 1 and Rq. 3. In addition, when inputs are positive values, max pooling regards maximum elements as significant features and enhances them, which satisfies Rq. 1. Nevertheless, max pooling never suppresses or removes noise, which dissatisfies Rq. 2, Rq. 3, and Rq. 4.

Therefore, we define a feature-aware activation function for the shortcut connection of a residual block to suppress noise and enhance anti-spoofing features, which can be expressed as

$$z_l = M_l \odot x_l \tag{15}$$

where M_l is the weights of z_l, which has the same shape as the lth input x_l. We use $M_l^{i,j}$ to demonstrate the enhancement or suppression impact of $x_l^{i,j}$ on $z_l^{i,j}$.

Inspired by the analysis of plain blocks, ReLU and max pooling in Section III. First, we use $R_u^{\max}$ to denote the maximum among the surrounding elements of $x_l^{i,j}$ in a $[p \times p]$ receptive field R_u, which can be expressed as

$$R_u^{\max} = \max_{x_l^{m,n} \in R_u \ \& \ (m,n) \neq (i,j)} x_l^{m,n}. \tag{16}$$

Then, we introduce a significant threshold ST to denote that $x_l^{i,j}$ is more significant than its surrounding elements of its receptive field. We also define a threshold ET to restrict the outputs of the shortcut connection to avoid overwhelming the outputs of the weight layers. Thus, the five rules are designed as follows.

1) *Rule 1:* If $x_l^{i,j}$ is less than zero, $x_l^{i,j}$ should be removed. Then, $z_l^{i,j}$ should be 0. According to the analysis of Section III-D, the negative inputs increase the influence of noise when putting a max pooling on the shortcut connection. Thus, we regard the negative inputs as noise and remove them, similar to ReLU. Accordingly, $M_l^{i,j}$ is 0.
2) *Rule 2:* If $x_l^{i,j}$ is less than $R_u^{\max}$, $x_l^{i,j}$ is considered as noise, which should be removed. Then, $z_l^{i,j}$ should be 0. In this situation, we consider nonmaximum elements as noise, just like max pooling. Therefore, we can interrupt the propagation of $x_l^{i,j}$ directly, like the plain blocks. Then, $M_l^{i,j}$ is 0.
3) *Rule 3:* If $x_l^{i,j}$ is greater than $R_u^{\max}$ and less than ST, $x_l^{i,j}$ is not more significant than its surrounding elements, which means that it is unable to determine whether $x_l^{i,j}$ is anti-spoofing features or noise. Thus, $z_l^{i,j}$ should be suppressed. In this situation, $x_l^{i,j}$ greater than $R_u^{\max}$ may be caused by noise, similar to that max pooling shows a little weakness when noise changes the position of the maximum element. Therefore, to suppress $z_l^{i,j}$, we use an exponential function to make $M_l^{i,j}$ less than 1.
4) *Rule 4:* If $x_l^{i,j}$ is greater than ST and less than ET, $x_l^{i,j}$ is more significant than its surrounding elements, which means $x_l^{i,j}$ is the anti-spoofing feature. Thus, $z_l^{i,j}$ should be enhanced. We believe that significant inputs contain anti-spoofing features. $x_l^{i,j}$ should be enhanced to further mask noise and other insignificant inputs, similar to that max pooling can lower the influence of noise by increasing the ratio of pure speeches. Therefore, we use a convex quadratic function to increase $M_l^{i,j}$ quickly.
5) *Rule 5:* If $x_l^{i,j}$ is greater than ET, $x_l^{i,j}$ should not be infinitely enhanced to avoid $z_l^{i,j}$ of shortcut connection from overwhelming the outputs of weight layers in the residual block. When $x_l^{i,j}$ equals ET, $M_l^{i,j}$ reaches its maximum $M_l^{\max}$. To restrict $z_l^{i,j}$ from increasing, we use an inverse proportional function to control $M_l^{i,j}$.

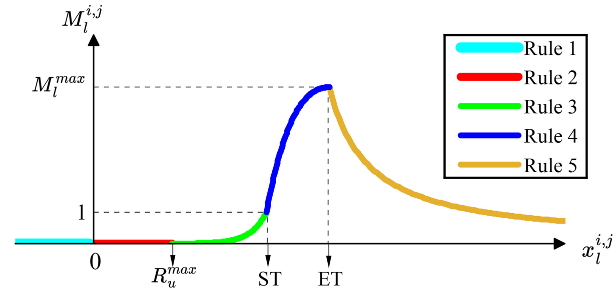

Fig. 3. Function $M_l^{i,j}$.

Therefore, $M_l^{i,j}$ can be expressed as a piecewise function as shown in Fig. 3 to correspond to the five rules, which can be formulated as

$$M_l^{i,j} = \begin{cases} 0, & x_l^{i,j} < 0 \\ 0, & 0 \le x_l^{i,j} < R_u^{\max} \\ e^{\mathrm{cur}\left(x_l^{i,j} - \mathrm{ST}\right)}, & R_u^{\max} \le x_l^{i,j} < \mathrm{ST} \\ \frac{\left(1-M_l^{\max}\right)\left(x_l^{i,j}-\mathrm{ET}\right)^2}{(\mathrm{ST}-\mathrm{ET})^2} + M_l^{\max}, & \mathrm{ST} \le x_l^{i,j} < \mathrm{ET} \\ \frac{M_l^{\max} \cdot \mathrm{ET}}{x_l^{i,j}}, & x_l^{i,j} \ge \mathrm{ET} \end{cases} \tag{17}$$

where cur is used to control the curvature of the exponential function. Thus, we can obtain the activation result z_l in (15) by M_l in (17).

In summary, the fine-grained activation function based on the five rules satisfies the four requirements. Rules 1 and 2 remove negative and nonmaximum elements directly, which satisfy Rq. 2 of removing insignificant features. Meanwhile, Rule 3 suppresses the uncertain inputs, which satisfies Rq. 3. Moreover, Rule 4 enhances the significant inputs, which satisfies Rq. 1. In addition, Rule 5 guarantees that the outputs on the shortcut connection enhancement cannot overwhelm the outputs of weight layers, which satisfies Rq. 1. Finally, these five rules distinguish the inputs into significant features, uncertain features, and noises, to enhance, suppress, and zeroize, respectively, which satisfy Rq. 4.

Consequently, the proposed feature-aware activation on the shortcut connection can make the residual blocks concentrate on the less but more significant information hidden in noisy inputs, instead of passing all noisy information without any limitation like the traditional residual blocks.

B. Searching Hyperparameters for Feature-Aware Activation

There are three hyperparameters in the proposed feature-aware activation function, i.e., ST, ET, and cur. We design a searching algorithm to obtain the three optimal hyperparameters during training.

To focus on the less but more significant information hidden in x_l, our objective can be converted into maximizing the variance of the enhanced outputs and the suppressed outputs in z_l. In addition, to avoid overwhelming the outputs of the weight layers, z_l should be restricted to be less than $\mathcal{F}(x_l)$ (i.e., the outputs of the lth weight layers). Thus, finding the optimal hyperparameters can be transformed as

$$\begin{aligned} &\text{Maximize } \mathcal{L} = \text{variance}(z_l) \\ &\text{Subject to } \sum \mathcal{F}(x_l)^{i,j} > \sum z_l^{i,j} \end{aligned} \tag{18}$$

where $\mathcal{F}(\cdot)$ is the function of the weight layers.

To avoid z_l from overwhelming $\mathcal{F}(x_l)$, we keep the range of z_l the same as $\mathcal{F}(x_l)$. Therefore, $z_l^{\max}$ as the maximum output of the feature-aware activation function can be formulated as

$$z_l^{\max} = \max \mathcal{F}(x_l) = \text{ET} \cdot M_l^{\max}. \tag{19}$$

We propose a VIO algorithm to find the optimal hyperparameters. The augmented Lagrangian algorithm (AUGLAG) is used to generate a population of candidate solutions. The details of VIO algorithm are shown in Algorithm 1. First, the *epoch* and the weight matrix M_l are initialized (line 1). $M_l^{i,j}$ is calculated by (17) after obtaining the max element $R_u^{\max}$ in the receptive field R_u except $x_l^{i,j}$ and generating the population of candidate ST, ET, and cur by AUGLAG (lines 2–11). By the sum and variance of z_l, the objective can be obtained in (18) (lines 12–14). Only if the sum of z_l is less than that of $\mathcal{F}(x_l)$, hyperparameters will be recorded (lines 15–21). After *epoch* reaches the maximal search duration $\text{epoch}_{\max}$, the optimal ST, ET, and cur are returned (lines 23 and 24).

Algorithm 1: VIO Algorithm

Input: the lth input $x_l = [h \times w]$, the lth output of weight layers $\mathcal{F}(x_l)$, the control parameter of the receptive field p, the maximal search duration $epoch_{\max}$
Output: ST, ET, cur
1 Initialize $epoch = 0$ and $M_l = [h \times w]$;
2 **while** $epoch < epoch_{\max}$ **do**
3   **for** $i \leftarrow 0$ **to** $h-1$ **do**
4   **for** $j \leftarrow 0$ **to** $w-1$ **do**
5   Set $x_l \leftarrow x_l$ after ReLU;
6   Obtain the receptive field R_u of $x_l^{i,j}$;
7   Set $R_u^{\max} \leftarrow \max_{x_l^{m,n} \in R_u \ \& \ (m,n) \neq (i,j)} x_l^{m,n}$;
8   Obtain the candidate ST, ET, cur by AUGLAG;
9   Obtain $M_l^{i,j}$ by putting ST, ET, cur, $R_u^{\max}$, and $x_l^{i,j}$ into (17).
10   **end**
11   **end**
12   $z_l = M_l \odot x_l$;
13   set $sum\_z \leftarrow$ sum of z_l and $var\_z \leftarrow$ variance of z_l;
14   set $\mathcal{L} = var\_z$ and $sum\_\mathcal{F}=$ sum of $\mathcal{F}(x_l)$,;
15   **if** $sum\_z > sum\_\mathcal{F}$
16   Clear ST, ET, cur;
17   break;
18   **else**
19   Record $\mathcal{L}$, ST, ET, cur;
20   **end**
21   $epoch = epoch + 1$;
22 **end**
23 Find maximum $\mathcal{L}$ with ST, ET, cur;
24 **Return** ST, ET, cur;

C. Network Model via Improved Residual Blocks

Combining (1) with (15), we can formulate the lth output y_l of the improved residual block as

$$y_l = M_l \odot x_l + \mathcal{F}(x_l, \mathcal{W}_l). \tag{20}$$

Assuming $x_{l+1} \equiv y_l$, recursively $x_{l+1} = y_l = M_l \odot x_l + \mathcal{F}(x_l, \mathcal{W}_l)$ and $x_{l+2} = y_{l+1} = M_{l+1} \odot x_{l+1} + \mathcal{F}(x_{l+1}, \mathcal{W}_{l+1})$, etc., we can formulate the forward propagation process from the lth to Lth improved residual block

$$\begin{aligned} y_L = x_l \odot \prod_{k=l}^{L} M_k + \sum_{i=l}^{L-1} \prod_{k=i+1}^{L} M_k \odot \mathcal{F}(x_i, \mathcal{W}_i) \\ + \mathcal{F}(x_L, \mathcal{W}_L). \end{aligned} \tag{21}$$

From the part $x_l \odot \prod_{k=l}^{L} M_k$ of (21), x_l affects y_L through the feature-aware activation for $L - l + 1$ times. Meanwhile, from the part $\sum_{i=l}^{L-1} \prod_{k=i+1}^{L} M_k \odot \mathcal{F}(x_i, \mathcal{W}_i)$, $\mathcal{F}(x_i, \mathcal{W}_i)$ affects y_L through the feature-aware activation for $L - i$ times. Therefore, with the increase of layers, insignificant features would be removed and significant features, including anti-spoofing features, would be enhanced.

Consequently, we propose an anti-spoofing detection method based on the feature-aware activation function, as shown in Fig. 4.

During the forward propagation, since the traditional loop operation costs a lot of time, we use unfolding and folding operations to reduce the time overheads. First, an input x_l with the shape of $[h \times w]$ is unfolded into $h \cdot w$ vectors with the shape of $[p^2 \times 1]$. $x_l^{i,j}$ is the first element of the $((i-1) \cdot w + j)$th receptive field $R_{(i-1) \cdot w + j}$. Accordingly, $R_{(i-1) \cdot w + j}$ can be expressed as

$$\begin{aligned} R_{(i-1) \cdot w + j} = \Big(& x_l^{i,j}, \ldots, x_l^{i+p-1,j}, \ldots, \\ & x_l^{i,j+p-1}, \ldots, x_l^{i+p-1,j+p-1} \Big)^T. \end{aligned} \tag{22}$$

We can obtain the weight $M_l^{i,j}$ by (17) via comparing $x_l^{i,j}$ with $R_{(i-1) \cdot w + j}^{\max}$, ST, and ET, which are obtained by Algorithm 1. Then, we get z_l (the output of the feature-aware activation function) by (15).

During the backward propagation, we directly use M_l obtained from the forward propagation to activate the input gradient, since taking the partial derivative of x_l is complex and slow. Via the chain rule of backpropagation on (15), we have

$$\frac{\partial \delta}{\partial x_l} = \frac{\partial \delta}{\partial z_l} \frac{\partial z_l}{\partial x_l} = \frac{\partial \delta}{\partial z_l} M_l \tag{23}$$

where δ is the loss function of the network.

By (23), M_l increases the update speed of anti-spoofing features and slows down that of noisy features, resulting in an

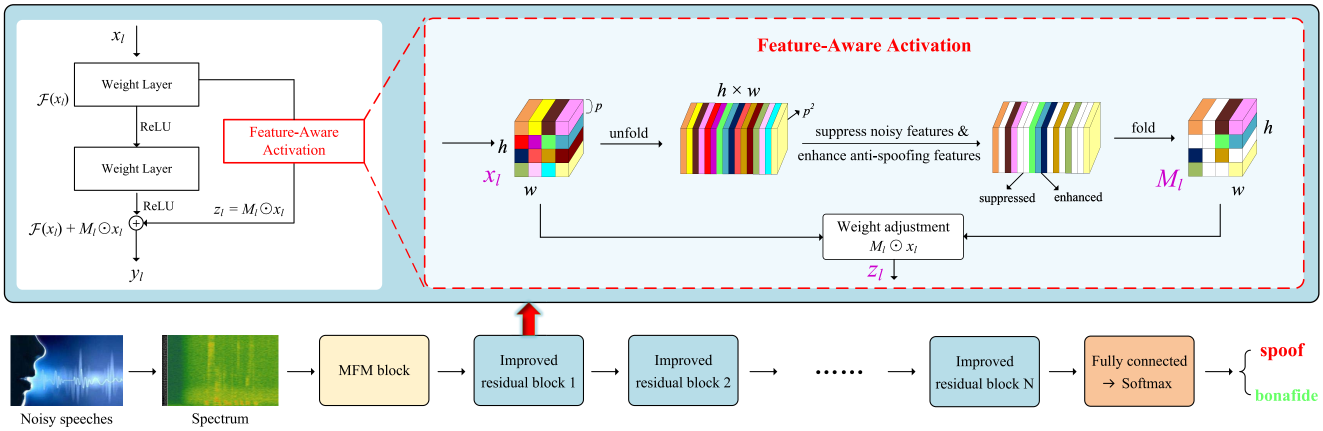

Fig. 4. Anti-spoofing detection design for noisy speeches. Noisy speeches are transformed into spectrums and then inputted into the max feature map (MFM) block to alleviate data redundancy. After passing through the residual blocks, outputs are fed into a fully connected layer with two units that produce classification logits. The logits are finally converted to a probability distribution using a final softmax layer.

Algorithm 2: Parallel-Propagation Algorithm

```
/* Forward propagation */
Input: the lth input x_l = [h × w], the control parameter of the
       receptive field p
Output: the lth output of the feature-aware activation z_l
1  Initialize M_vec;
2  Obtain ST, ET, and cur by Algorithm 1;
3  [R_1, R_2, ..., R_hw] ← unfold x_l with p;
4  Parallel u ← 1 to hw do
5  |   Set R_u^max ← max_{k=1}^{p^2-1} R_u[k];
6  |   Obtain M_u by comparing R_u[0] in (17);
7  |   M_vec append M_u;
8  end
9  M_l ← fold M_vec with p;
10 z_l ← M_l ⊙ x_l;    /* Hadamard product */
11 Return z_l;
/* Backward propagation */
Input: the lth input gradient InGrad_l = [h × w], M_l calculated
       by forward propagation
Output: the l-the output gradient OutGrad_l
12 OutGrad_l ← InGrad_l ⊙ M_l;    /* Hadamard product */
13 Return OutGrad_l;
```

accelerated update rate for the significant features compared with the insignificant ones.

Therefore, we propose a parallel-propagation algorithm to accelerate the forward and backward propagation of the feature-aware activation. The details are given in Algorithm 2. In the forward propagation process, the weight vector M_vec with the shape of $[1 \times hw]$ is initialized to save weights (lines 1). The three hyperparameters are obtained by Algorithm 1 (line 2). $h \cdot w$ receptive fields are got by unfolding x_l with p (line 3). Then, the weights are parallelly calculated by comparing elements in the receptive fields with (17) (lines 4–8). By folding M_vec with p, the weight M_l is obtained, and then the output of the feature-aware activation z_l is returned after a Hadamard product (lines 9–11). In the backward propagation process, M_l is regarded as a constant to activate the gradient directly (lines 12 and 13).

V. Experiments

We conduct experiments to evaluate the efficiency of the proposed method. Some experiments are performed on an Intel Xeon Gold 5218 CPU @ 2.30 GHz and eight NVIDIA GeForce RTX 3090 GPUs, and others are performed on an NVIDIA Jetson AGX Xavier with NVIDIA Volta architecture GPU and Carmel architecture 8-core CPU.

A. Experiment Setup

We choose ASVspoof2021 dataset [29] and NOISEX-92 corpus [31] to evaluate the proposed method. ASVspoof2021 is a widely used spoofed speech dataset, which provides 19 kinds of spoofing attacks, including six spoofing attacks for training and other thirteen spoofing attacks for testing. NOISEX-92 corpus is a widely used noise dataset, which contains common types of noises in the real world.

We randomly add noise (*m109* and *factory1* from NOISEX-92, and *rain* collected from real world) at the SNRs from 5 to 15 dB to ASVspoof2021 training and testing sets to establish T1 and E1 sets. We also randomly add noises (*babble, f16, factory2, white*, and *volvo* from NOISEX-92) at the SNRs from 5 to 15 dB to ASVspoof2021 testing set to establish E2 set. We train our model on T1 and evaluate on E1 and E2.

For comparison, we choose four candidates as follows.

1) *Spec-ResNet [24]:* An anti-spoofing detection method based on traditional residual blocks.
2) *RawNet2 [22]:* One of the four baselines of ASVspoof2021 Challenge.
3) *LFCC-LCNN [25]:* One of the four baselines of ASVspoof2021 Challenge.
4) *RawGAT-ST [23]:* The champion of ASVspoof2021 Challenge.

In addition, we also choose two denoising tools (i.e., Denoiser [33] and PCS [34]), which are used together with the four candidates to evaluate the efficiency of the proposed method.

Since the inputs with the shape of [86 × 9] need to be converted into a single value for final logical classification logits, there are at least six residual blocks in the network

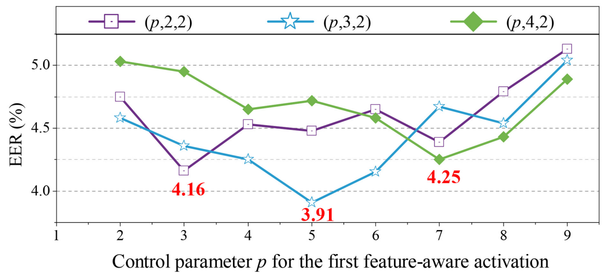

Fig. 5. EER comparison on different receptive fields.

model to reduce the dimensions. Therefore, we only use six residual blocks to realize the fast detection of spoofed noisy speeches.

B. Ablation Experiment

1) Optimal p for Activation-Based Residual Blocks: According to Section IV-A, p controls the size of receptive field of the feature-aware activation function, which determines the ability of noise suppression and feature enhancement on the shortcut connection. With the decrease of p, more inputs will be remained. The improved residual block will degrade into the traditional residual block when p is 1. To obtain the optimal p of feature-aware activation functions on shortcut connections, we evaluate the EER on E1 set at an SNR of 10 dB. Since the shape of inputs is $[86 \times 9]$, the max optional p of the six feature-aware activation functions are 9, 4, 2, 1, 1, and 1, respectively, which means that the last three residual blocks degrade into the traditional residual blocks. Thus, we only test p in the first three residual blocks. Due to the limited selections of p for the second and third layers, we test p of the first layer by providing all combinations of the two layers. The experimental results are shown in Fig. 5. We can observe that the EER is the lowest when the three p are 5, 3, and 2, respectively. Therefore, in the following experiments, we, respectively, set p as 5, 3, 2, 1, 1, and 1 for the six residual blocks.

2) Impact of Parallel-Propagation Algorithm: We evaluate the performance of parallel-propagation algorithm during training and inference. We conduct the experiment on both RTX 3090 and AGX Xavier to test the training and inference time overheads. For comparison, we choose traditional residual blocks and improved residual blocks with *for* loops and *numba* [35]. *numba* is a tool for accelerating *for* loops. The experimental results are shown in Table I. *for* loops cost 6728.54 and 835.78 ms during training and inference on AGX Xavier, which is unsuitable for embedded systems. On RTX 3090, *numba* saves the training and inference time overheads to 14.08 and 12.96 ms due to its acceleration of *for* loops. Our parallel-propagation algorithm can reduce the training and inference time overheads to 9.06 and 7.48 ms, which are only 1.77 and 1.23 ms more than those of traditional residual blocks with identity mapping.

3) Ablation Experiment for Improved Residual Blocks, Traditional Residual Blocks, and Plain Blocks: We conduct

TABLE I
PERFORMANCE OF PARALLEL-PROPAGATION ALGORITHM

	Traditional residual blocks	Improved residual blocks		
Training time (ms)	Identity mapping	*for* loops	*numba*	Ours
RTX 3090	7.29	4821.37	14.08	9.06
AGX Xavier	17.71	6728.54	32.87	22.54
Inference time (ms)	Identity mapping	*for* loops	*numba*	Ours
RTX 3090	6.25	561.29	12.96	7.48
AGX Xavier	15.83	835.78	27.45	19.29

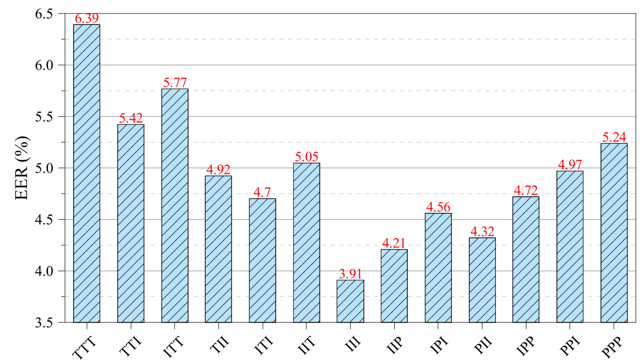

Fig. 6. Ablation experiment on different block combinations.

an ablation experiment to evaluate the performance of our improved residual block compared with the traditional residual block and the plain block. The traditional residual block allows all inputs to be directly sent to the next block whereas the plain block prevents all inputs from being sent to the next block. Since the optimal p of the six residual blocks are 5, 3, 2, 1, 1, and 1, the last three residual blocks degrade into the traditional residual blocks. Thus, we only test different combinations of the first three residual blocks on E1 at an SNR of 10 dB. The experimental results are shown in Fig. 6, where I, T, and P denote the improved residual block, the traditional residual block, and the plain block, respectively. For example, *ITI* denotes that the first, second and third residual blocks are the improved residual block, the traditional residual block and the improved residual block, respectively. We can observe that *III* achieves an EER of 3.91%, which is the lowest among all combinations. *TTT* has the highest EER, which is 6.39%. In addition, replacing the improved residual block with the traditional residual block always performs worse than replacing it with the plain block. For example, EERs of *IIT* and *IIP* are 5.05% and 4.21%, respectively, which shows that *IIT* performs worse than *IIP*. Similar to *IIT* and *IIP*, *ITT* performs worse than *IPP*, and *TTT* performs worse than *PPP*.

According to the experimental results, our improved residual block achieves better performance on detecting spoofed noisy speeches, compared with the traditional residual block and the plain block.

4) Impact of the Feature-Aware Activation on Spec-ResNet: Spec-ResNet is a state-of-the-art model based on traditional residual blocks. We test the performance of the feature-aware activation by replacing traditional residual blocks of Spec-ResNet with improved ones. The optimal p of the six improved residual blocks are 7, 5, 3, 2, 1, and 1, respectively. The

TABLE II
IMPACT OF FEATURE-AWARE ACTIVATION ON SPEC-RESNET

Model	SNR 5		SNR 10		SNR 15		Pure	Average
	E1	E2	E1	E2	E1	E2		
Spec-ResNet	21.23%	23.73%	18.04%	20.42%	16.57%	18.23%	9.68%	18.27%
Spec-ResNet with feature-aware activation	8.01%	9.65%	7.06%	8.12%	6.71%	7.15%	6.55%	7.61%
ours	5.51%	6.87%	3.91%	4.92%	3.16%	4.26%	2.95%	4.51%

TABLE III
EER COMPARISON ON NOISY AND PURE SPEECHES

Model	SNR 5		SNR 10		SNR 15		Pure	Average
	E1	E2	E1	E2	E1	E2		
Spec-ResNet	21.23%	23.73%	18.04%	20.42%	16.57%	18.23%	9.68%	18.27%
RawNet2	29.19%	23.12%	21.53%	19.53%	7.55%	5.69%	4.58%	15.88%
LFCC-LCNN	32.93%	22.95%	23.12%	19.67%	13.18%	15.02%	6.35%	19.03%
RawGat-ST	38.43%	40.27%	20.35%	21.97%	8.36%	8.87%	**1.06%**	19.90%
PCS+RawNet2	14.75%	11.36%	9.24%	7.97%	7.92%	7.11%	6.87%	9.31%
Denoiser+RawNet2	9.274%	10.45%	6.58%	6.676%	5.81%	5.683%	4.89%	7.05%
PCS+LFCC-LCNN	20.13%	24.29%	9.246%	13.82%	12.78%	10.94%	8.17%	14.20%
Denoiser+LFCC-LCNN	29.24%	22.95%	30.31%	19.67%	21.12%	15.02%	13.79%	21.72%
PCS+RawGat-ST	20.22%	22.27%	7.002%	7.80%	5.16%	5.78%	2.43%	10.09%
Denoiser+RawGat-ST	8.45%	8.85%	6.75%	7.19%	4.98%	5.57%	1.75%	6.22%
ours	**5.51%**	**6.87%**	**3.91%**	**4.92%**	**3.16%**	**4.26%**	2.95%	**4.51%**

experimental results are shown in Table II. EER of the original Spec-ResNet on E1 at the SNR of 5 dB is 21.23%, whereas that of Spec-ResNet with feature-aware activation is only 8.01%. The feature-activation can decrease the average EER of Spec-ResNet from 18.27% to 7.61% after replacing traditional residual blocks to improved ones. Moreover, the proposed method can further decrease the average EER from 7.61% to 4.51%. Therefore, the feature-aware activation can improve the detection performance of traditional residual blocks in noisy environments.

C. EER Comparison

In this section, we evaluate the EER of our method on E1 and E2 sets at SNRs from 5 to 15 dB compared with Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST.

The experimental results are summarized in Table III. We can have the following observations. On E1 and E2 at the SNR of 5 dB, EERs of RawNet2, LFCC-LCNN, Spec-ResNet, and RawGAT-ST are significantly high. This indicates that existing anti-spoofing detection methods cannot effectively identify noisy speeches. Combining existing methods with the denoising tools can improve certain performances. Taking E1 at the SNR of 5 dB as an example, Denoiser+RawNet2 decreases EER from 29.19% to 9.274% compared to RawNet2, and Denoiser+RawGAT-ST decreases EER from 38.43% to 8.45% compared with RawGAT-ST. However, EERs of Denoiser+LFCC-LCNN, PCS+LFCC-LCNN, and PCS+RawGAT-ST are still very high, which are 29.24%, 20.13%, and 20.22%, respectively. It means that denoising tools cannot fundamentally solve the problem of detecting spoofed noisy speeches. Moreover, on E1 at the SNR of 15 dB, EERs of LFCC-LCNN and RawNet2 (i.e., 13.18% and 7.55%) are lower than those of Denoiser+LFCC-LCNN and PCS+RawNet2 (i.e., 21.12% and 7.92%). This indicates that the denoising tools might not only remove noises but also destroy the anti-spoofing features, which may bring extra difficulties in detecting speeches in noisy environments. We can have similar observations on E2.

The average EER of our method is only 4.51%, which is the lowest among these 11 methods. The average EERs of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST are 18.27%, 15.88%, 19.03%, 19.90%, 9.31%, 7.05%, 14.20%, 21.72%, 10.09%, and 6.22%, respectively. According to the above experimental results, our method can detect spoofed noisy speeches effectively compared with the other ten methods.

D. t-DCF Comparison

In this section, we evaluate the tandem decision cost function (t-DCF) of the proposed method on E1 and E2 compared with the other ten methods.

The experimental results are shown in Table IV. The average t-DCF of our method is the lowest, which is 0.1204, whereas the average t-DCFs of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST are 0.4624, 0.4301, 0.5842, 0.4410, 0.2459, 0.1641, 0.4001, 0.6548, 0.2725, and 0.1436, respectively.

According to the above experimental results, our method has the best t-DCF performance in detecting spoofed noisy speeches among these 11 methods.

E. Accuracy Comparison

In this section, we evaluate the accuracy of the proposed method on E1 and E2 compared with the other ten methods. Since ASVspoof 2021 dataset consists of more than 90%

TABLE IV
t-DCF Comparison on Noisy and Pure Speeches

Model	SNR 5		SNR 10		SNR 15		Pure	Average
	E1	E2	E1	E2	E1	E2		
Spec-ResNet	0.5357	0.5997	0.4336	0.517	0.4268	0.4504	0.2741	0.4624
RawNet2	0.7049	0.7349	0.6273	0.5221	0.1723	0.1381	0.1111	0.4301
LFCC-LCNN	0.8602	0.7573	0.7349	0.6856	0.3839	0.5090	0.1587	0.5842
RawGat-ST	0.7430	0.8424	0.5630	0.6089	0.1440	0.1523	**0.0334**	0.4410
PCS+RawNet2	0.3729	0.3264	0.2410	0.2096	0.2020	0.1913	0.1781	0.2459
Denoiser+RawNet2	0.2118	0.2488	0.1518	0.1528	0.1342	0.1375	0.1122	0.1641
PCS+LFCC-LCNN	0.6789	0.8353	0.2410	0.3054	0.3343	0.2122	0.1942	0.4001
Denoiser+LFCC-LCNN	0.7286	0.7573	0.7648	0.6856	0.6453	0.5090	0.4932	0.6548
PCS+RawGat-ST	0.5310	0.6215	0.2051	0.22	0.1185	0.1289	0.0788	0.2725
Denoiser+RawGat-ST	0.1883	0.2005	0.1556	0.1582	0.1229	0.1349	0.0453	0.1436
ours	**0.1527**	**0.1715**	**0.1008**	**0.1395**	**0.0865**	**0.1098**	0.0825	**0.1204**

TABLE V
Average Accuracy Comparison on Noisy and Pure Speeches

Model	SNR 5		SNR 10		SNR 15		Pure	Average
	E1	E2	E1	E2	E1	E2		
Spec-ResNet	78.52%	75.22%	82.85%	81.38%	83.33%	81.55%	90.15%	81.85%
RawNet2	70.25%	74.58%	77.91%	80.21%	91.25%	93.35%	94.58%	83.16%
LFCC-LCNN	67.05%	77.21%	76.15%	80.05%	85.15%	84.87%	93.45%	80.56%
RawGat-ST	61.55%	59.66%	79.65%	77.56%	90.85%	91.35%	**98.35%**	79.85%
PCS+RawNet2	85.17%	87.59%	89.37%	91.32%	92.35%	92.85%	93.05%	90.24%
Denoiser+RawNet2	90.15%	88.67%	93.1%	92.89%	93.54%	94.03%	95.13%	92.50%
PCS+LFCC-LCNN	79.45%	73.58%	89.81%	85.67%	87.65%	88.17%	90.66%	84.99%
Denoiser+LFCC-LCNN	70.35%	74.68%	68.67%	80.38%	77.59%	84.68%	86.63%	77.56%
PCS+RawGat-ST	79.88%	76.38%	92.67%	93.15%	94.13%	93.89%	97.62%	89.67%
Denoiser+RawGat-ST	91.54%	91.02%	92.34%	92.67%	94.35%	93.85%	98.04%	93.40%
ours	**94.05%**	**92.15%**	**95.95%**	**95.24%**	**96.72%**	**95.43%**	97.15%	**95.24%**

spoofed speeches, we test the accuracy after balancing the bonafide speeches and the spoofed speeches instead of accuracy on the unbalanced dataset.

The experimental results are shown in Table V. The average accuracy of our method is the highest, which is 95.24%, whereas the average accuracy of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST are 81.85%, 83.16%, 80.56%, 79.85%, 90.24%, 92.50%, 84.99%, 77.56%, 89.67%, and 93.40%, respectively. We can observe that the denoising tools may destroy the anti-spoofing features, which decreases the performance of detection methods. For example, on E1 at the SNR of 15 dB, LFCC-LCNN decreases the accuracy from 85.15% to 77.59% after using Denoiser. When detecting pure speeches, RawGAT-ST decreases the accuracy from 98.35% to 97.62% after using PCS, and from 98.35% to 98.04% after using Denoiser. Moreover, when detecting pure speeches, the accuracy of our method is 1.20%, 0.47%, and 0.89% lower than those of RawGat-ST, PCS+RawGat-ST, and Denoiser+RawGat-ST. This is because the proposed feature-aware activation may remove certain pure anti-spoofing features. However, the accuracy decrease of detecting pure speeches is relatively low with regard to the accuracy improvement of detecting noisy speeches.

According to the experimental results, denoising tools may not always improve the anti-spoofing detection. Our method performs best in detecting spoofed noisy speeches among these 11 methods.

F. Accuracy Comparison on Different Durations of Speech Segments

Embedded systems usually have strict constraints on the response time of tasks. The durations of some speeches are very long, even as long as a few minutes. It may take several minutes to detect the complete speeches, which would dissatisfy the real time requirements of embedded systems. In this section, we evaluate the performance of anti-spoofing detection by different durations of speech segments and obtain the optimal detection duration for quick response tasks of embedded systems. Since most durations of speeches in ASVspoof 2021 are within 5 s, the durations of speech segments are set to increase from 0.25 to 5 s at an interval of 0.25 s. For comparison, we chose Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST as the candidates.

The experimental results on E1 at an SNR of 10 dB are shown in Fig. 7. When speech segment duration is 0.25 s, the accuracy of our method is the highest among these 11 methods, which is 77.85%, whereas the accuracies of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST are 48.83%, 65.85%, 66.87%, 70.24%, 68.49%, 75.78%, 70.89%, 44.29%, 73.25%, and 76.2%, respectively. The accuracies of these 11 methods are very low because the speech segment duration of 0.25 s has few detectable features. When speech segment duration is 1.5 s, our

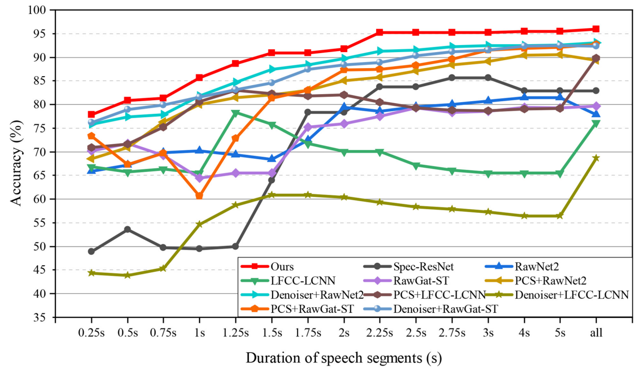

Fig. 7. Accuracy comparison on different durations of speech segments.

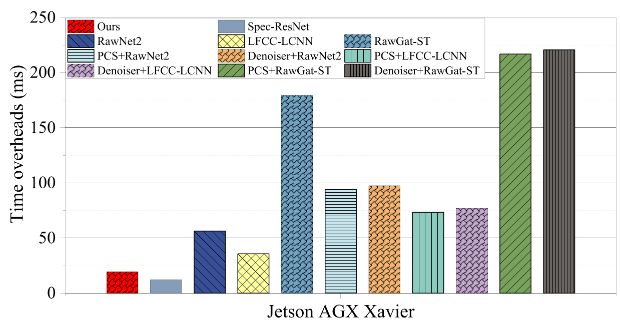

Fig. 8. Time overheads comparison on Jetson AGX Xavier.

method can achieve the accuracy of 90.92%, an acceptable accuracy for embedded systems, which is 26.93%, 22.55%, 15.1%, 25.41%, 8.84%, 3.53%, 8.61%, 30.1%, 9.54%, and 6.39% higher than those of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST, respectively. For the complete speeches, the accuracy of our method is still the highest, achieving 95.95%, which is 13.10%, 18.04%, 19.8%, 16.3%, 6.58%, 2.85%, 6.14%, 27.28%, 3.28%, and 3.61% higher than those of Spec-ResNet, RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST, respectively.

Therefore, our method can achieve an acceptable accuracy in a very short speech segment for embedded systems compared with the other ten methods.

G. Time Overhead Comparison on Embedded Devices

In this section, we evaluate the time overhead of our method on NVIDIA Jetson AGX Xavier. The experimental results on E1 at the SNR of 10 dB are shown in Fig. 8. On Jetson AGX Xavier, the time overhead of Spec-ResNet is the least (12.17 ms) and that of our method is the second least (19.29 ms). However, the average EER of Spec-ResNet is 18.27% whereas that of our method is only 4.51%, as shown in Table III. The time overheads of RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST are 56.38, 35.7, 179.2, 94.05, 97.42, 73.37, 76.75, 216.87, and 220.62 ms, respectively. The time overhead of our method is only 34.21%, 54.03%, 10.76%, 20.51%, 19.80%, 26.29%, 25.13%, 8.89%, and 8.74% of those of RawNet2, LFCC-LCNN, RawGAT-ST, PCS+RawNet2, Denoiser+RawNet2, PCS+LFCC-LCNN, Denoiser+LFCC-LCNN, PCS+RawGAT-ST, and Denoiser+RawGAT-ST.

Therefore, we can conclude that our method can detect spoofed noisy speeches with high accuracy and low time overhead.

VI. Conclusion

In this article, we have made efforts to detect spoofed noisy speeches for embedded systems with high accuracy and low time overhead. We proposed an anti-spoofing detection method with activation-based residual blocks to reduce the influence of noise and enhance the anti-spoofing features. Based on the formulation of the noise propagation model and the analysis of the impact on the outputs of the traditional residual blocks, we presented a feature-aware activation to realize the fine-grained processing of removing noise and strengthen significant features. We also devised a VIO algorithm to find the optimal hyperparameters of the feature-aware activation function. We conducted extensive experiments to evaluate the effectiveness of our approach on datasets of ASVspoof2021 dataset and NOISEX-92 corpus. The experimental results demonstrated the efficiency of our method, which can achieve high accuracy and low time overhead in detecting spoofed noisy speeches compared with the other ten methods.

For the future work, we plan to make efforts on the following studies. First, we will improve the feature-aware activation to extract features from the 3-D inputs. Second, we plan to explore position-aware rules for residual blocks to influence the degrees of enhancement and suppression. Third, we will apply our method to more real-life applications such as speech-based assistants to make it more practical.

References

[1] M. Sahidullah et al., "Integrated spoofing countermeasures and automatic speaker verification: An evaluation on ASVspoof 2015," in *Proc. 17th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2016, pp. 1700–1704.
[2] T. Toda et al., "The voice conversion challenge 2016," in *Proc. 17th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2016, pp. 1632–1636.
[3] A. van den Oord et al., "WaveNet: A generative model for raw audio," in *Proc. 9th ISCA Speech Synth. Workshop (SSW)*, 2016, p. 125.
[4] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system," in *Proc. 9th ISCA Speech Synth. Workshop (SSW)*, 2016, pp. 202–207.
[5] W. Huang, C. Lo, H. Hwang, Y. Tsao, and H. Wang, "Wavenet vocoder and its applications in voice conversion," in *Proc. 30th Annu. Conf. Comput. Linguist. Speech Process. (ROCLING)*, 2018, pp. 96–110.
[6] Y.-Y. Ding, H.-J. Lin, L.-J. Liu, Z.-H. Ling, and Y. Hu, "Robustness of speech spoofing detectors against adversarial post-processing of voice conversion," in *Proc. 22nd IEEE/ACM Trans. Audio, Speech, Lang. Process. (TASLP)*, 2021, pp. 3415–3426.
[7] A. Van Den Oord et al., "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*.
[8] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *Proc. 21th IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2020, pp. 6199–6203.

[9] C. Hanilci, T. Kinnunen, M. Sahidullah, and A. Sizov, "Spoofing detection goes noisy: An analysis of synthetic speech detection in the presence of additive noise," in *Proc. 17th Speech Commun.*, 2016, pp. 83–97.

[10] X. Tian, Z. Wu, X. Xiao, E. S. Chng, and H. Li, "An investigation of spoofing speech detection under additive noise and reverberant conditions," in *Proc. 17th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2016, pp. 1715–1719.

[11] S. Lv, X. Wang, S. Sun, L. Ma, and L. Xie, "DCCRN-KWS: An audio bias based model for noise robust small-footprint keyword spotting," in *Proc. 24th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2023, pp. 929–933.

[12] H. Martel, J. Richter, K. Li, X. Hu, and T. Gerkmann, "Audio-visual speech separation in noisy environments with a lightweight iterative model," in *Proc. 24th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2023, pp. 1673–1677.

[13] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 28492–28518.

[14] Y. Gong, S. Khurana, L. Karlinsky, and J. Glass, "Whisper-AT: Noise-robust automatic speech recognizers are also strong general audio event taggers," in *Proc. 24th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2023, pp. 2798–2802.

[15] J.-H. Kim, J. Heo, H. Jin Shim, and H.-J. Yu, "Extended U-net for speaker verification in noisy environments," in *Proc. 23th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2022, pp. 590–594.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Med. Image Comput. Comput.-Assist. Interv. (MICCAI)*, 2015, pp. 234–241.

[17] L. Zhang, X. Wang, E. Cooper, and J. Yamagishi, "Multi-task learning in utterance-level and segmental-level spoof detection," in *Proc. Automat. Speaker Verification Spoofing Countermeasures Challenge*, 2021, pp. 9–15.

[18] X. Wang, X. Qin, T. Zhu, C. Wang, S. Zhang, and M. Li, "The DKU-CMRI system for the ASVspoof 2021 challenge: Vocoder based replay channel response estimation," in *Proc. Automat. Speaker Verification Spoofing Countermeasures Challenge*, 2021, pp. 16–21.

[19] W. Ge, J. Patino, M. Todisco, and N. Evans, "Raw differentiable architecture search for speech deepfake and spoofing detection," in *Proc. Automat. Speaker Verification Spoofing Countermeasures Challenge*, 2021, pp. 22–28.

[20] R. K. Das, "Known-unknown data augmentation strategies for detection of logical access, physical access and speech deepfake attacks: ASVspoof 2021," in *Proc. Automat. Speaker Verification Spoofing Countermeasures Challenge*, 2021, pp. 29–36.

[21] J. Zhan, Z. Pu, W. Jiang, J. Wu, and Y. Yang, "Detecting spoofed speeches via segment-based word CQCC and average ZCR for embedded systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 3862–3873, Nov. 2022.

[22] H. Tak, J. Patino, M. Todisco, A. Nautsch, N. Evans, and A. Larcher, "End-to-end anti-spoofing with rawnet2," in *Proc. 22th IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2021, pp. 6369–6373.

[23] H. Tak, J. Weon Jung, J. Patino, M. Kamble, M. Todisco, and N. Evans, "End-to-end spectro-temporal graph attention networks for speaker verification anti-spoofing and speech deepfake detection," in *Proc. 22th Automat. Speaker Verification Spoofing Countermeasures Challenge*, 2021, pp. 1–8.

[24] M. Alzantot, Z. Wang, and M. B. Srivastava, "Deep residual neural networks for audio spoofing detection," in *Proc. 20th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2019, pp. 1078–1082.

[25] X. Wang and J. Yamagishi, "A comparative study on recent neural spoofing countermeasures for synthetic speech detection," in *Proc. 22th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2021, pp. 4259–4263.

[26] Z. Wu et al., "ASVspoof 2015: The first automatic speaker verification spoofing and countermeasures challenge," in *Proc. 16th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2015, pp. 2037–2041.

[27] T. Kinnunen et al., "The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection," in *Proc. 18th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2017, pp. 2–6.

[28] M. Todisco et al., "ASVspoof 2019: Future horizons in spoofed and fake audio detection," in *Proc. 20th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2019, pp. 1008–1012.

[29] H. Delgado et al., "ASVspoof 2021: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," 2021, *arXiv:2109.00535*.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. 14th Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 630–645.

[31] "NOISEX-92." 2023. [Online]. Available: https://github.com/speechdnn/Noises/tree/master NoiseX-92

[32] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," 2015, *arXiv:1510.08484*.

[33] A. Défossez, G. Synnaeve, and Y. Adi, "Real time speech enhancement in the waveform domain," in *Proc. 21th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2020, pp. 3291–3295.

[34] R. Chao, C. Yu, S. Wei Fu, X. Lu, and Y. Tsao, "Perceptual contrast stretching on target feature for speech enhancement," in *Proc. 23th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, 2022, pp. 5448–5452.

[35] "Numba 0.59.1." 2024. [Online]. Available: https://doi.org/10.5281/zenodo.10839385