WavLM and Omni-Scale CNNs: Enhancing Boundary Detection in Partially Spoofed Audio

Menghan Li* and Zhihua Huang†
* School of Computer Science and Technology, Xinjiang University, Urumqi, China
E-mail: 1131687972@qq.com Tel/Fax: +86-15599694888
† Key Laboratory of Signal Detection and Processing in Xinjiang, Umumgi, China
E-mail:zhhuang@xju.edu.cn Tel/Fax: +86-18699179000

Abstract—**Partially spoofed/fake audio, in which segments of utterances are replaced with synthetic or natural audio clips, has emerged as a new form of deep audio forgery, posing potential severe threats to societal security. To address this issue, we employ a deep learning-based frame-level detection system, introducing a frame-level detection approach. We explore the effectiveness of WavLM in waveform boundary detection, utilizing WavLM as a feature extractor for original audio samples. Acoustic features and frame-level embeddings are concatenated, with an OS block embedded within the frame-level feature extraction process, in conjunction with CNN-1D to form the ResNet-OS network. This system can detect partially deceived audio and pinpoint the manipulated segments, effectively integrating multi-scale convolution to consider the interplay between local and global information in time series classification tasks. Experimental results show that this approach offers substantial generalization capabilities and robustness compared to traditional frame-level detection techniques.**

I. Introduction

Deepfake audio refers to sounds that are modified or created using deep learning technologies, aimed at deceiving both humans and machines. These technologies, particularly text-to-speech (TTS) and voice conversion (VC), have significantly enhanced the realism of synthetic voices, making them nearly indistinguishable from natural speech[1]–[3]. In this context, high-fidelity synthesis/conversion systems inevitably provide criminals with the means to commit fraud by impersonating others. Therefore, the detection of audio forgeries is linked to societal security, privacy protection, and property safety, making it crucial to effectively detect deceptive speech to mitigate the potential threats posed by false information in audio content.

In response to the threats posed by audio spoofing attacks, the ASVspoof Challenge is held biennially to explore defensive strategies against various types of spoofing attacks, including synthetic speech, voice conversion, replay, and impersonation[4]. Nevertheless, a significant scenario has been overlooked in most datasets and challenges where a bonafide speech utterance is contaminated by synthesized speech segments, leading to partial spoofing (PS).Attackers can use PS technology to modify key words in a sentence, such as time, place, and characters, thereby changing the semantic meaning of the sentence. This type of modification is low-cost and easy to perform. Therefore, defending against this PS scenario presents a significant challenge for defenders.

In recent years, significant advancements have been made in the realm of Audio Deepfake Detection (ADD) concerning PS scenarios. Yi et al.[5]developed a dataset focused on altering several key words in semi-authentic audio, while Zhang et al.[6]designed the ”PartialSpoof” voice database specifically for PS scenarios. These databases signify the commencement of PS scenario research within ADD tasks. Researchers have explored large-scale self-supervised pre-training models[7], [8],which have demonstrated superior performance over traditional acoustic features such as MFCC and LFCC. Wu et al.[9] have developed a boundary detection system named ”Fake Span Discovery,” which is capable of identifying splicing boundaries. Additionally, Track 2 of ADD 2023 emphasizes the importance of accurately locating manipulated regions, making the research into the precise localization of spliced segments a more challenging and critical task. The performance of the boundary detection system still requires further enhancement.

This paper introduces a novel frame-level boundary detection system designed to identify partially spoofed audio. Unlike previous approaches that focused on utterance-level detection, our system exploits discontinuities between audio segments to accurately detect connection boundaries at the frame level. We employ WavLM as the feature extractor, coupled with the ResNet-OS architecture and a Transformer-based frame-level classifier for boundary detection. The model was trained using the HAD dataset and evaluated across multiple test sets. The results demonstrate that our system surpasses existing partial fake audio detection systems with boundary detection capabilities. It achieved an Equal Error Rate (EER) of 0.06% on the HAD validation set and 0.023% on the HAD test set, indicating its state-of-the-art performance in detecting forged regions within audio streams.

II. METHOD

In this study, we focus on audio signals containing forged segments, with the objective to identify frames that contain discontinuous information. Given the acoustic feature input $X = (x_1, x_2, \ldots, x_T) \in \mathbb{R}^{D\times T}$, where D represents the dimension of features and T represents the number of frames, this task is defined as a frame-level binary classification problem. The classification labels $y = (y_{1}, y_{2}, ..., y_{T,}) \in \{0, 1\}^T$.

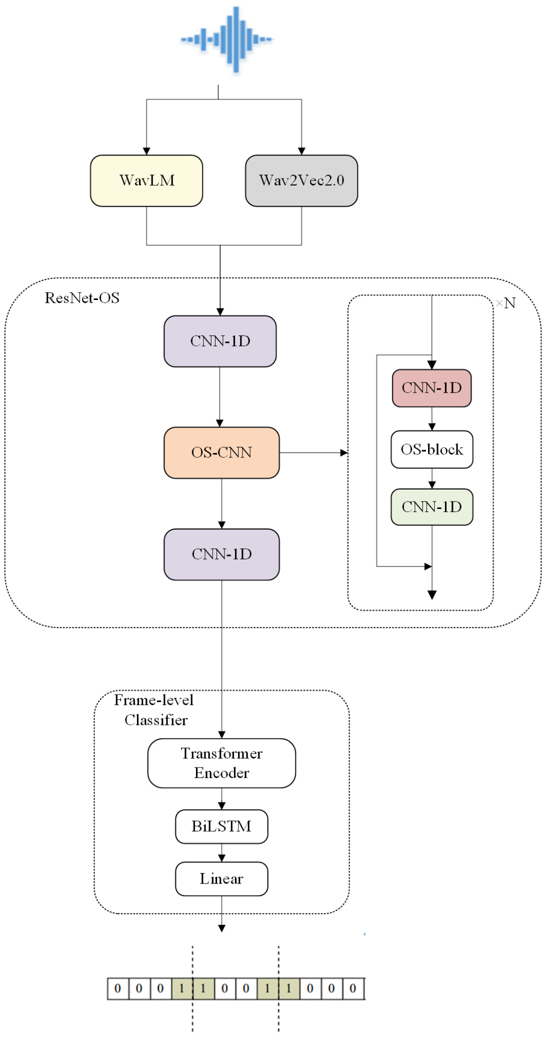

Fig. 1. The architecture of our proposed model

Our system is capable of identifying specific frames that contain discontinuities. If a frame is located at the boundary between forged and authentic audio, it is labeled as 1; otherwise, it is labeled as 0. To enhance the robustness of the system, frames close to the boundary are also marked as 1. This approach not only detects forged segments within the audio signal but also accurately pinpoints their locations.

A. PROPOSED MODEL

Figure 1 depicts the architecture of our proposed model. We leverage the pretrained WavLM model[10] for feature extraction from the original audio. Subsequently, we utilize ResNet-OS to further extract frame-level embeddings. These acoustic features and frame-level embeddings are then concatenated and input into a transformer-based frame-level classifier, which provides the probability of each frame being a boundary.

B. Feature extraction

Wav2Vec[11]and WavLM[10]are two state-of-the-art self-supervised learning models that are primarily used for processing speech data. They have demonstrated exceptional performance in numerous downstream tasks, including automatic speech recognition (ASR)[12]. Consequently, we also employ these unsupervised models as feature extractors for original audio samples. We set the frame count T for both Wav2Vec and WavLM at 64, with an output feature frame rate of 20 milliseconds, culminating in a total feature dimension of 768.

C. Frame-level embedding extraction

After the feature extractor, we employ the ResNet-OS network to obtain frame-level embeddings, as illustrated in Figure 1. The ResNet-OS architecture includes two CNN-1D layers with N residual blocks sandwiched between them. Each residual block consists of two CNN-1D layers and an OS block, and features a residual connection that runs from the input to the output.

ResNet-Os is the core architecture of the model, consisting of N residual blocks, each configured as an OS-CNN. OS-CNN (Omni-Scale Convolutional Neural Network) is an architecture specifically designed for time series classification, based on the traditional one-dimensional convolutional neural network (1D-CNN). This architecture incorporates OS block[13], enabling the model to effectively learn and extract features across various scales. Consequently, it excels in handling time series data characterized by complex patterns and dynamic changes. ResNet-OS includes two CNN-1D layers, between which N OS-CNN blocks are interspersed. Each OS-CNN block comprises two CNN-1D layers, with an OS block embedded between them. Moreover, the structure features a residual connection running from the input to the output to enhance learning capabilities and efficiency in information transfer. The architecture of the OS block, depicted in Figure 2, is a three-layer, multi-core convolutional network structure, where each convolutional layer employs multiple kernels and performs same-padding convolution operations. For configuring the kernel sizes, we use $p^{(i)}$ to represent the set of kernel sizes for the i-th layer:

$$P^{(i)} = \begin{cases} \{1, 2, 3, 5, \ldots, p^k\} & i \in \{1, 2\} \\ \{1, 2\} & i = 3 \end{cases} \tag{1}$$

This set can include various sizes, allowing each layer to capture features at different scales. Typically, smaller kernels are capable of capturing more detailed features, while larger kernels can capture broader features.The core concept of the OS block (Omni-Scale Block) revolves around how to cover all possible receptive field (RF) sizes by selecting convolutional kernels of different sizes. The receptive field size, defined as the size of the region in the input that produces the feature, has always been a crucial factor affecting the performance of one-dimensional convolutional neural networks (1D-CNN) in time series classification tasks. The set of receptive field sizes, S, represents the collective receptive field sizes of all paths. The calculation of the receptive field set S can be described as follows:

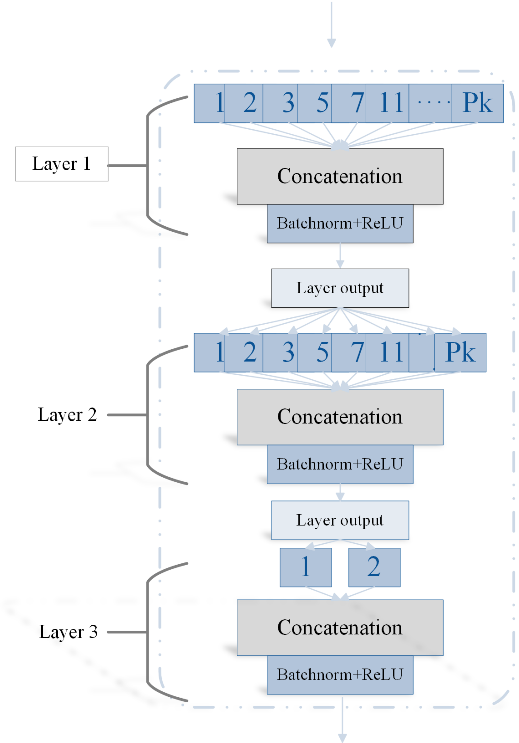

Fig. 2. Diagram of the OS block.

$$S = \left\{ p^{(1)} + p^{(2)} + p^{(3)} - 2 \mid p^{(i)} \in \mathbb{P}^{(i)} \right\} \tag{2}$$

Based on Goldbach's conjecture, which posits that every even number can be expressed as the sum of two primes, we can expand the receptive field set:

$$S = \left\{ e + p^{(3)} - 2 \mid p^{(3)} \in \mathbb{P}^{(3)}, e \in \mathbb{E} \right\} \tag{3}$$

where $\mathbb{E}$ is the set of all even receptive fields generated by the first two layers. Combining Equation (3) and Equation (1), we get:

$$S = \{e \mid e \in \mathbb{E}\} \cup \{e - 1 \mid e \in \mathbb{E}\} \equiv \mathbb{N}^+ \tag{4}$$

$\mathbb{N}^+$ represents the set of all positive integers, indicating that through appropriate selection of convolutional kernels, it is possible to cover all receptive field (RF) sizes from 1 to the maximum possible size. Therefore, by properly choosing p^k, we can cover any range of integer receptive field sizes, thereby extracting more effective frame-level features.

D. Frame-level classifier

To capture long-range global context within frames, we utilize several Transformer encoders. Then, we further model the sequence embeddings from the Transformer encoders using BiLSTM. Finally, a fully connected layer predicts the boundary probability for each frame.

III. EXPERIMENT

A. Data Preparation

Our training data comes from the Half-truth Audio Detection (HAD) dataset[5]. This type of audio only modifies a few words in the original speech, posing a serious threat to audio verification systems, as these small-scale modifications are often difficult to detect. Table I provides statistics on the datasets used in our experiments. As shown in Table I, the HAD-train dataset contains 26,554 genuine utterances and 26,554 fake utterances, while the HAD-dev dataset includes 8,914 genuine utterances and 8,914 fake utterances. All fake utterances from the HAD-train and HAD-dev datasets are synthesized using mainstream speech synthesis technologies. The HAD-test dataset consists of 9,072 labeled utterances.

TABLE I
THE STATISTICS OF DATASETS (#UTTERANCES)

Name	Bona fide	Fake	ALL
HAD-train	26554	26554	53108
HAD-dev	8914	8914	17824
HAD-test	-	-	9072

Half-truth Audio Detection (HAD) dataset is based on AISHELL-3 corpus[14]. It is publicly available2 and under theApache license 2.0 . AISHELL-3 is a multi-speaker Mandarin speech corpus for training text-to-speech (TTS) models. We inserted audio clips into authentic discourse according to the following strategy:

1. Named entity recognition and lexical annotation using jieba, followed by random replacement or substitution of these keywords with antonyms in order to change the semantics and emotional expression of the original utterance.

2. The (GST)-based Tacotron[15], [16] system is used to generate audio corresponding to the edited text, which is then processed by the neural vocoder LPCNet to enhance the naturalness and emotional expression of the synthesized audio and to ensure that its sound quality is consistent with the original recording.

3. The precisely processed synthesized audio is inserted into a specific position of the original audio, and the volume and sound quality are adjusted to ensure a seamless auditory integration and consistency between the synthesized part and the original audio.

In terms of model training, we combine the real discourse in the HAD-train dataset with part of the fake dataset as the final training data. Based on the HAD-test set, it is evaluated by combining it with the HAD-dev set as an adaptation dataset.

B. Parameter settings

ResNet-OS consists of 12 OS-CNN blocks, each containing a CNN-1D layer without bias, with a kernel size of 1 and both input and output sizes set to 512. The first CNN-1D operates without bias at a kernel size of 5, with an input size of 768 and an output size of 512. The final CNN-1D has a kernel size of 1, input size of 512, and outputs frame-level embeddings with an embedding size set to 128.

For the frame-level classifier, the Transformer encoder includes four attention heads and two encoder layers. The size of the feed-forward network (FFN) is 1024. The BiLSTM contains 128 hidden neurons and is followed by a ReLU activation. Finally, a 256-dimensional fully connected layer predicts the probability of each frame.

C. Training Process

During the training process, genuine and fake utterances are considered identical since there is no boundary between them. We randomly draw an audio sample from the partially fake dataset and genuine data from HAD-train. The probability of selecting a positive sample is set at 0.5 to ensure data balance. Each audio is cut to a fixed length, for instance, 0.64 seconds, 1.28 seconds, or 2.56 seconds. We use the MUSAN[29] and RIRs[30] corpora for online data augmentation. For positive samples, we set the label as a zero vector. For negative samples, the boundary label between the genuine audio clip and the fake audio clip is set to 1, with all others set to zero. In addition, we also set the labels near the boundaries to 1.

In our experiments, the model is trained 100 epochs using binary cross-entropy loss and the Adam optimizer. The batch size is set at 64, and the learning rate is set at 10^{-4}. During training, we evaluate the equal error rate (EER) on the adaptation set and the test set. For the model based on wav2vec, we take the average of the five models with the lowest EER for inference and evaluation. However, averaging models based on wavlm resulted in decreased performance. Therefore, for the wavlm-based model, we only consider the model with the best performance for evaluation.

IV. RESULTS AND DISCUSSION

Our experiments evaluated two systems: one trained with the Wav2Vec feature extractor and the other with the WavLM feature extractor. The results are shown in Table II ,indicate that an audio length of 1.28 seconds offers the best performance for tasks using WavLM and Wav2Vec. This may be because this duration is sufficient to capture adequate contextual information without introducing noise or unnecessary details due to excessive length. The model based on Wav2Vec features achieves an Equal Error Rate (EER) of 0.08% on the HAD-dev set and 0.067% on the HAD-test set at 1.28 seconds. For the WavLM model at the same duration, the EER on HAD-dev is 0.06%, and on HAD-test it is 0.023%, with performance degradation observed at audio lengths of 0.64 seconds and 2.56 seconds. These results suggest that the model trained with WavLM features exhibits superior generalization in this task compared to Wav2Vec.

TABLE II
SYSTEM PERFORMANCES REGARDING VARIOUS SEGMENT L, REPORTED IN EER.

Feature	Test Sets	Wav length l (s)		
		0.64s	1.28s	2.56s
Wav2Vec (w/o OS block)	HAD-dev HAD-test	0.35% 0.20%	**0.08%** **0.067%**	0.42% 0.18%
WavLM	HAD-dev HAD-test	0.19% 0.12%	**0.06%** **0.023%**	0.28% 0.10%

In this study, we set the waveform length l to 1.28 seconds for models based on Wav2Vec and WavLM, and conducted a series of ablation studies to evaluate the performance impact of specific network components. As shown in Table III , we removed certain components from the proposed network and reported the corresponding performance. Specifically, the ”w/o ResNet-OS” model indicates the removal of the ResNet-OS component used for frame-level embedding extraction. For the Wav2Vec model, the equal error rate (EER) increased to 0.16% on the HAD-dev dataset and to 0.15% on the HAD-test dataset after removing ResNet-OS, demonstrating its critical role in model performance. Similarly, for the WavLM model, removing ResNet-OS resulted in an EER of 0.10% on HAD-dev and 0.05% on HAD-test. Removing the OS block lead to an EER of 0.073% on HAD-dev and 0.035% on HAD-test for the WavLM model, proving that the OS block also plays a significant role in enhancing model performance.

Overall, the WavLM model showed a better error rate performance compared to Wav2Vec, especially evident in the test set. These findings emphasize the critical role of the OS block and ResNet-OS structure in enhancing the accuracy of partial fake audio detection tasks, and removing these key components significantly reduces model performance.

TABLE III
PERFORMANCES (EER) REGARDING VARIOUS FEATURE EXTRACTOR AND ARCHITECTURE DESIGNS, W/O MEANS WITHOUT.

Model	HAD-dev	HAD-test
Wav2Vec	0.08%	0.067%
w/o ResNet-OS	0.16%	0.15%
WavLM	**0.06%**	**0.023%**
w/o ResNet-OS	0.10%	0.05%
w/o OS block	0.073%	0.035%

V. CONCLUSIONS

We developed a frame-level partial fake audio detection method that not only provides binary judgments at the utterance level for partially spoofed audio but also predicts the precise insertion or replacement locations of fake clips. By conducting ablation studies on our model components and evaluating various acoustic features, including Wav2Vec and WavLM, our experimental results demonstrate that this system outperforms existing boundary detection systems in detecting the boundaries of partially spoofed audio.

REFERENCES

[1] A. van den Oord, S. Dieleman, H. Zen, *et al.*, “Wavenet: A generative model for raw audio,” in *9th ISCA Speech Synthesis Workshop*, 2016, pp. 125–125.

[2] J. Shen, R. Pang, R. J. Weiss, *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[3] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 17 022–17 033.

[4] M. R. Kamble, H. B. Sailor, H. A. Patil, and H. Li, "Advances in anti-spoofing: From the perspective of asvspoof challenges," *APSIPA*, vol. 9, 2020.

[5] J. Yi, Y. Bai, J. Tao, *et al.*, "Half-truth: A partially fake audio detection dataset," in *Proceedings of Interspeech*, 2023, pp. 1654–1658.

[6] L. Zhang, X. Wang, E. Cooper, J. Yamagishi, J. Patino, and N. Evans, "An initial investigation for detecting partially spoofed audio," in *Proceedings of Interspeech*, 2021, pp. 4264–4268.

[7] J. Martín-Doñas and A. Álvarez, "The vicomtech audio deepfake detection system based on wav2vec2 for the 2022 add challenge," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022, pp. 9241–9245.

[8] Z. Cai, W. Wang, and M. Li, "Waveform boundary detection for partially spoofed audio," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023, pp. 1–5.

[9] H. Wu, H.-C. Kuo, N. Zheng, *et al.*, "Partially fake audio detection by self-attention-based fake span discovery," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022, pp. 9236–9240.

[10] S. Chen, C. Wang, Z. Chen, *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[11] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 449–12 460, 2020.

[12] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli, "Unsupervised speech recognition," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 826–27 839, 2021.

[13] W. Tang, G. Long, L. Liu, T. Zhou, M. Blumenstein, and J. Jiang, "Omni-scale cnns: A simple and effective kernel size configuration for time series classification," 2022.

[14] Y. Shi, H. Bu, X. Xu, S. Zhang, and M. Li, *Aishell-3: A multispeaker mandarin tts corpus and the baselines*, arXiv preprint arXiv:2010.11567, 2020.

[15] R. J. Skerry-Ryan, E. Battenberg, X. Ying, Y. Wang, and R. A. Saurous, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[16] Y. Wang, D. Stanton, and Y. Zhang, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.