IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 January 2023, accepted 24 February 2023, date of publication 9 March 2023, date of current version 15 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3254880

RESEARCH ARTICLE

Experimental Case Study of Self-Supervised Learning for Voice Spoofing Detection

YERIN LEE[1], NARIN KIM[1], JAEHONG JEONG[2,3], AND IL-YOUP KWAK[1], (Member, IEEE)
[1]Department of Applied Statistics, Chung-Ang University, Seoul 06974, South Korea
[2]Department of Mathematics, Hanyang University, Seoul 04763, South Korea
[3]Research Institute for Natural Sciences, Hanyang University, Seoul 04763, South Korea

Corresponding author: Il-Youp Kwak (ikwak2@cau.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Ministry of Science and ICT under Grant 2020R1C1C1A01013020, Grant RS-2023-00208284 and Grant 2021R1F1A1049185.

ABSTRACT This study aims to improve the performance of voice spoofing attack detection through self-supervised pre-training. Supervised learning needs appropriate input variables and corresponding labels for constructing the machine learning models that are to be applied. It is necessary to secure a large number of labeled datasets to improve the performance of supervised learning processes. However, labeling requires substantial inputs of time and effort. One of the methods for managing this requirement is self-supervised learning, which uses pseudo-labeling without the necessity for substantial human input. This study experimented with contrastive learning, a well-performing self-supervised learning approach, to construct a voice spoofing detection model. We applied MoCo's dynamic dictionary, SimCLR's symmetric loss, and COLA's bilinear similarity in our contrastive learning framework. Our model was trained using VoxCeleb data and voice data extracted from YouTube videos. Our self-supervised model improved the performance of the baseline model from 6.93% to 5.26% for a logical access (LA) scenario and improved the performance of the baseline model from 0.60% to 0.40% for a physical access (PA) scenario. In the case of PA, the best performance was achieved when random crop augmentation was applied, and in the case of LA, the best performance was obtained when random crop and random shifting augmentations were considered.

INDEX TERMS Spoofing detection, self-supervised learning, contrastive learning.

I. INTRODUCTION

In general, the performance of a machine learning model is enhanced by increasing the number of data used in the learning process. This is especially true for deep learning compared to other traditional machine learning models [1]. However, it is often not easy to procure a large amount of labeled data in a supervised learning situation because of labeling costs. Therefore, it is common to encounter semi-supervised learning situations where only some of the data are labeled. A self-supervised learning approach, that learns general feature representation using pseudo-labeling, is widely used to handle semi-supervised learning [2], [3]. Contrastive learning is one of the best-performing self-supervised learning approaches and we refer to [4], [5], [6] as references. In this study, we examined the performance of voice spoofing detection models using the latest contrastive learning frameworks.

The associate editor coordinating the review of this manuscript and approving it for publication was M. Shamim Kaiser.

A. VOICE SPOOFING ATTACK

Voice assistants have been applied for performing various tasks, such as schedule management, home appliance control, and financial transactions, via voice commands [7]. With the rapid increase in voice assistant application market size, interest in speaker recognition technology that allows voice assistants to respond only to registered voices has increased [8]. Speaker recognition is a function that identifies a person from a voice pattern, and there are two aspects: speaker identification and speaker verification. Speaker identification selects a specific speaker by finding the voice most similar to the input voice among multiple candidates registered in the voice assistant's database [9]. It is possible for the speaker identification function to erroneously characterize an unregistered

voice as the voice of one of the registered candidates. Speaker verification uses technology to determine whether the input voice matches the voice registered in the voice assistant and supplements the speaker identification function [10].

Attempts to defeat speaker verification through voice spoofing attacks threaten the security of voice assistants. Voice spoofing attacks try to deceive the speaker verification system by simulating the voice registered in the voice assistant using the following methods: text-to-speech (TTS), voice conversion (VC), and replay spoofing attacks. Figure 1 illustrates the two scenarios in detecting voice spoofing attacks. The logical access (LA) attack detection scenario detects TTS and VC voices, and the physical access (PA) attack detection scenario detects replay spoofing attacks [11].

Deep learning models have been receiving increasing attention in voice spoofing attack detection. For example, the Automatic Speaker Verification Spoofing and Countermeasures (ASVspoof) challenge has been held every two years, starting in 2015, promoting the development of countermeasures against voice spoofing attacks on speaker verification systems [11], [12], [13]. In the 2017 competition, only 5 teams out of the top 10 considered deep learning models, but in the 2019 competition, 8 teams out of the top 10 used deep learning models. Moreover, the top 5 teams in the 2019 challenge all used deep learning models. As a result of the challenge, the constant-Q transform (CQT) feature [14] has been proven to work well in the deep learning model to distinguish voice spoofing attacks, and it has been widely used together with the short-time Fourier transform (STFT). Among the deep learning models, the light convolutional neural network (Light CNN) model [15] ranked first in 2017 and second in 2019. It has been established that the Max Feature Map (MFM) activation function of the model works well in detecting forgery attacks [16], [17]. In addition to Light CNN, various ResNet-based deep learning models have appeared. In particular, with ResMax (for 'Residual network with Max Feature Map') [18], the model showed high levels of performance in the form of equal error rates (EERs) of 2.19% for the LA attack set and 0.37% for the PA attack set, respectively, in the ASVspoof 2019 competition. Reference [18] used a model that considers the MFM activation function in the residual block.

B. SELF-SUPERVISED LEARNING

In supervised learning, all items in the dataset have associated labels, and both input data and labels are critical at the stage of the training process. A large number of labeled data is an essential prerequisite to enhance the performance of a supervised learning procedure. However, the labeling process requires substantial time and human effort. In general, the labeling cost increases linearly with the size of the dataset, while the improvement in model performance has a sublinear relation to the size of the dataset [3]. Various methods, such as transfer learning, semi-supervised learning, and self-supervised learning, have been proposed to reduce the labeling cost via relatively inexpensive unlabeled data.

Self-supervised learning, like unsupervised learning, does not use predetermined labels [2]. Rather, learning occurs through unlabeled data by defining a new pretext task that is used to pre-train a model using an automatically generated pseudo label. The pre-trained network is subsequently transferred to the primary model for the execution of the downstream task. By letting the neural network learn a pretext task using many unlabeled data, the model can improve our understanding of the dataset itself. In this study, we achieved enhanced classification performance by performing pre-training as a pretext task through self-supervised learning and subsequently transferring the results from the learning to the execution of a downstream task.

C. CONTRASTIVE LEARNING

Contrastive learning is a self-supervised learning algorithm that has enjoyed much attention recently [4], [19], [20], [21], [22], [23], [24], showing excellent performance in ImageNet. Existing self-supervised learning methods utilize data augmentation techniques, such as rotation [25], colorization [26], jigsaw [27], and the prediction of relative positions [28]. Self-supervised learning trains a model so that there is a high similarity between an image and its transformation. Therefore, feature representation resulting from the self-supervised training of many image data would capture the traits of a diverse range of essential image features.

Various studies on contrast learning were conducted quite recently, including MoCo [19], [20], SimCLR [21], [22], and BYOL [4]. The main aim of contrast learning in this context is to learn whether two input images are similar or dissimilar. MoCo and SimCLR use similar methods for applying different techniques of data augmentation to a single image. This approach embodies a learning method that generates two deformed images by applying different random augmentation techniques to one image. Through the learning process, the model is trained to assign a higher degree of similarity to the augmented images generated from the same image source and a lower degree of similarity to the augmented images generated from different images.

D. CONTRASTIVE LEARNING FOR AUDIO DATA

The use of contrastive learning has been explored in various recent studies. Contrastive Predictive Coding (CPC) [6] applies a method that predicts the representations of future observations from those of past ones. It learns useful representations in four domains of application: speech, images, natural language, and reinforcement learning. CPC2 [29] is a modified version of CPC based on speech. It also studied the performance of combinations of augmentations. Various studies benchmarked SimCLR and MoCo and applied them to speech datasets [30], [31], [32], [33]. COLA [30] is a self-supervised pre-training approach for learning a general-purpose representation of audio input.

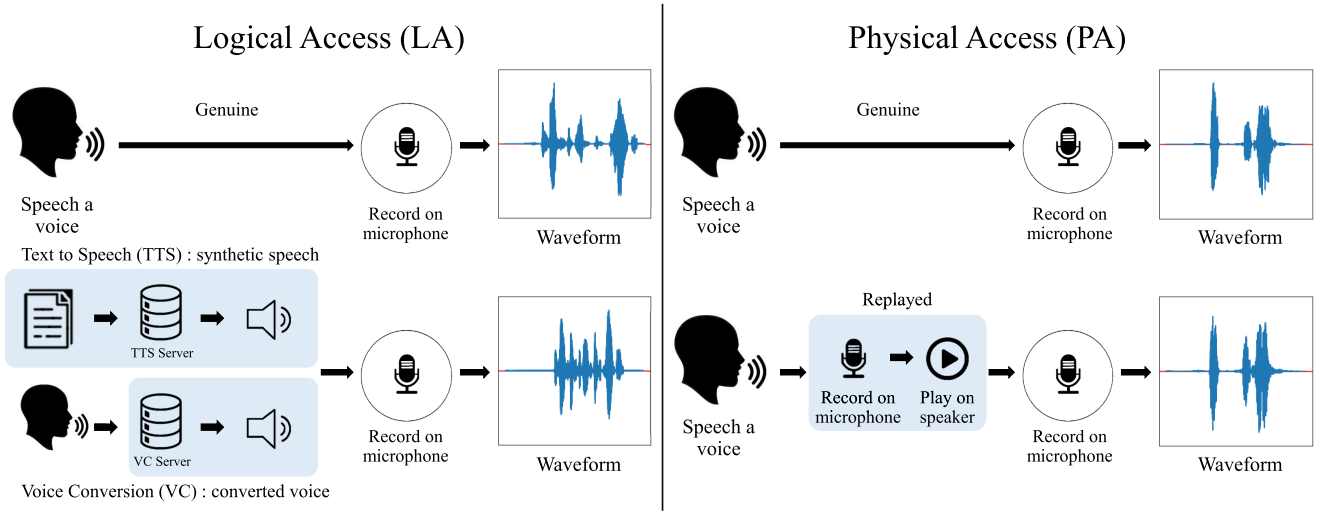


FIGURE 1. Processes involved in detecting voice spoofing attacks, for logical access and physical access, respectively.

After pre-training on the large-scale dataset AudioSet [34], a pre-trained model was tested on nine different tasks ranging from speaker identification to music recognition. CLAR [31] is a modified version of SimCLR applied to audio datasets that include speech data. Speech SimCLR [32] is another modified version of SimCLR using speech data, and it applies augmentation to raw speech and the associated spectrograms. The MoCo speaker embedding system [33] modified MoCo used for speaker verification. Similarly, as in Speech SimCLR, this method applies augmentation to raw speech and the associated spectrograms. The wav2vec 2.0 [35] paper proposes a self-supervised learning method that learns data representations from a large amount of speech data. HuBERT [36] approach uses offline clustering to provide aligned target labels for a prediction loss similar to that of BERT. HuBERT focuses on consistency of unsupervised clustering rather than the quality of assigned labels. By applying the prediction loss over masked regions only, the model learns a combined acoustic and language model over continuous inputs

E. EXPERIMENTAL CASE STUDY OF CONTRASTIVE LEARNING FOR VOICE SPOOFING DETECTION

In this study, we investigated the effect of pre-training based on contrastive learning on voice spoofing attack detection. For training, we used two features, STFT and CQT. We used ResMax models that had displayed good performance levels in the ASVspoof challenge as a base model [18]. The proposed method applies the data augmentation technique and has the following characteristics:

- MoCo's dynamic dictionary is considered to achieve relatively good performance even with small batch sizes. Since SimCLR creates negative pairs in one mini-batch (the size of the negative pairs depends on the size of the mini-batch, N), SimCLR requires a large batch size for efficient performance. MoCo relieves the SimCLR's problem that requires large batch sizes by using a dynamic dictionary to generate negative pairs. Therefore, the proposed model considered Moco's dynamic dictionary to ensure that experiments are feasible with small batch sizes. In our experimental study, the proposed outperformed the baseline model. EERs reduced from 6.93% to 6.23% for the LA scenario and from 0.60% to 0.47% for the PA scenario.
- The symmetric loss of SimCLR and BYOL is considered. The symmetric loss re-calculates loss by changing the original loss's query and key, and the model uses the average of the two resulting losses. In our study, the proposed model showed better performance in EERs. EERs reduced from 6.23% to 6.08% for the LA scenario and from 0.47% to 0.45% for the PA scenario.
- COLA's bilinear similarity is applied to learn weights for each query and key. The performance levels of 6.08% for the LA scenario and 0.45% for the PA scenario achieved in the preceding experiment were further improved from 6.08% to 5.92% for the LA scenario and from 0.45% to 0.40% for the PA scenario.
- Random crop and noise addition are used, which are commonly used augmentation techniques in existing studies of contrast learning with audio data. We experimented with center shifting, dynamic range change, and speed change. The combination of random crop, additive noise, and center shifting improved the performance in the LA scenario from 5.92% to 5.26%.

II. METHODS

A. SimCLR

SimCLR [21] is a framework for contrastive learning based on data augmentation. Say we have N image samples in a given training batch $(x_1, \cdots, x_N)$, two transformed images

$\tilde{x}_{i,1}$ and $\tilde{x}_{i,2}$ are generated from each image x_i, resulting $2N$ transformed images in one batch. In $2N$ number of images $\tilde{x}_{i,1}$ have one positive pair $\tilde{x}_{i,2}$, generated from the same source of image x_i, and other images are negative images.

The images $\tilde{x}_{i,j}, i \in \{1, \cdots, N\}, j \in \{1, 2\}$ are passed through the encoder $f(\cdot)$ to obtain fixed-size image representations $h_{i,j}$. The representations $h_{i,j} = f(\tilde{x}_{i,j})$ are passed through the projection head $g(\cdot)$, which is a nonlinear network, to obtain $z_k = g(f(\tilde{x}_{i,j})), k = 2(i-1)+j$. Here, $f(\cdot)$ is a base encoder, for example, we can use ResNet. A projector header $g(\cdot)$ can be a simple multi-layer perceptron.

The similarity between z_i, and z_j, $i, j \in \{1, \cdots, 2N\}$ is calculated using cosine similarity:

$$s_{i,j} = \frac{z_i^T z_j}{\tau \|z_i\| \|z_j\|},$$

where the quantity τ is a temperature parameter and controls the cosine similarity ranging from -1 to 1.

SimCLR uses a contrastive loss measure based on InfoNCE [6]. For the positive pair of z_i and z_j, the cross entropy can be obtained by taking the negative logarithm of the softmax output:

$$\ell(i, j) = -\log \frac{\exp(s_{i,j})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})}.$$

When the InfoNCE loss is calculated for the same two image pairs, the loss varies depending on the order of the two input images. Therefore, the cross entropy is calculated twice by exchanging z_i and z_j. The average of the two cross entropies produces the symmetric loss for all data pairs:

$$L = \frac{1}{N} \sum_{n=1}^{N} [\ell(2n-1, 2n) + \ell(2n, 2n-1)]. \quad (1)$$

The quantity L is a function of the weights of the encoder and projection heads. The SimCLR model is trained to minimize the symmetric loss L with respect to the weights of the encoder and projection heads.

B. MoCo

SimCLR generates negative pairs within a mini-batch, and a large batch size is required for a high level of performance (because the number of the negative pairs depends on the size N of the mini-batch). However, MoCo [19] obviates the need for a large batch size by using a dynamic dictionary to generate negative images.

The dynamic dictionary is maintained in a queue of constant size (K), greater than the batch size (N). In the dictionary, at the end of each batch, the key encoded by the current batch's momentum encoder is entered into the queue, and the oldest key is output from the queue. Thus, K vectors in the dynamic dictionary become negative pairs in the next batch.

The query and the key are representations created from the given data by transforming an image among the input data. The key works as an anchor that establishes the criterion for a positive or negative assignment of images. The query generated from the same image becomes the positive pair of the key, while the queries generated from different images become the negative pairs of the key.

Starting with one particular image x, two transformed images $\tilde{x}^q$ and $\tilde{x}^k$ are generated by applying different random augmentation techniques. The query q is generated by passing $\tilde{x}^q$ through the encoder $f_q(\cdot)$ and the key k_+ is generated by passing $\tilde{x}^q$ through the momentum encoder $f_k(\cdot)$. The encoder and momentum encoder share the same model architecture such as ResNet, but the weight parameters are different. The key k_+ becomes the only positive pair of q, and the keys $(k_1, k_2, \cdots, k_K)$ stored in the dynamic dictionary constitute the negative pairs of q. That is, one positive pair and K negative pairs are generated from the original image.

The similarity between the query q_i, $i \in \{1, \cdots, N\}$ and the key $k_{i,j}$, $j \in \{+, 1, 2, \cdots, K\}$ is calculated using cosine similarity and the range of similarity is adjusted using the temperature parameter:

$$s_{i,j} = \frac{q_i^T k_{i,j}}{\tau \|q_i\| \|k_{i,j}\|}.$$

The cross entropy is calculated in accordance with the infoNCE framework:

$$L = \frac{1}{N} \sum_{i=1}^{N} -\log \frac{\exp(s_{i,+})}{\sum_{j=1}^{K} \exp(s_{i,j})}.$$

At the end of each batch, the encoder is updated to minimize this cross entropy. The weight parameters of the momentum encoder are updated with a certain momentum from the previous weights of the momentum encoder. The procedure reduces the imbalance between the encoder and the momentum encoder.

C. COLA

COLA [30] is an approach that applies contrast learning to audio data by referring to SimCLR, MoCo, and CURL [37]. COLA uses bilinear similarity to improve the model's performance.

Similar to SimCLR, COLA generates audio features $\tilde{x}_{i,j}$, $i \in \{1, \cdots, N\}, j \in \{1, 2\}$ from transformed audio of the original audio x_i in one batch of size N. Audio features $\tilde{x}_{i,j}$ pass through the encoder $f(\cdot)$ and the projection head $g(\cdot)$, and then generate $z_k = g(f(\tilde{x}_{i,j})), k = 2(i-1)+j$. COLA's bilinear similarity between z_k and $z_l, k, l \in \{1, \cdots, 2N\}$, is calculated as follows:

$$s_{k,l} = z_k^T W z_l.$$

The quantity W is a weight matrix representing the correlation between each element of z_k and z_l. Subsequently, the parameters are updated through the training process.

III. PROPOSED MODEL

The overall process is illustrated in Figure 2. The processing of each batch is divided into four parts: (i) Two audio features are generated by applying different data augmentation

techniques to each audio sample in the batch and generating the audio features by applying Short-Time Fourier Transform (STFT) or a constant-Q-transform (CQT). Let's define them as $\tilde{x}^q$ and $\tilde{x}^+$. And $\tilde{x}^+$ is a positive pair of $\tilde{x}^q$. (ii) One of the two audio features generated from each audio sample passes through the encoder, and the other passes through the momentum encoder to obtain the representation of a fixed size ($f_q(\tilde{x}^q)$, and $f_k(\tilde{x}^+)$). (iii) Representations generated through the encoder and the momentum encoder both pass through the projection head, respectively, and generate queries and keys ($q = g(f_q(\tilde{x}^q))$, and $k_+ = g(f_k(\tilde{x}^+))$). (iv) The contrastive loss is obtained by calculating the similarity between the query and the keys in the dynamic dictionary.

A positive pair is generated by applying two different augmentation techniques to one audio sample in a batch of size N. Additive noise, dynamic range change, center shifting, random crop, and speed change are used as augmentation techniques. The quantities $\tilde{x}^q$ and $\tilde{x}^+$ are 2D spectrogram features generated by STFT or CQT from augmented audio signals.

One feature $\tilde{x}^q$ is passed through the encoder $f_q(\cdot)$, and the other $\tilde{x}^+$ is passed through the momentum encoder $f_k(\cdot)$, to obtain the fixed size representations $h_q = f_q(\tilde{x}^q)$ and $h_k^+ = f_k(\tilde{x}^+)$. The encoder and the momentum encoder have the same model structure as ResMax, but the weight of the momentum encoder is not updated through learning.

The expressions passed through the encoder and the momentum encoder, respectively, and then, subsequently passed through the same projection head $g(\cdot)$ to generate the query $q = g(f_q(\tilde{x}^q))$ and the key $k_+ = g(f_k^+(\tilde{x}^+))$. A study using SimCLRv2 [22] found that deep projection heads improve learning performance. Therefore, we consider a three-layer non-linear network for the projection head in our model. as described in Figure 2. The query q becomes an anchor that forms the criterion for determining positive and negative values, whereas the key k_+ becomes the only positive pair of the query q. We use the dynamic dictionary queue of MoCo to generate negative pairs. The keys $k_1, k_2, \cdots, k_K$ in the dynamic dictionary become negative pairs of the query q. That is, one positive pair and K negative pairs are considered for each audio feature.

The bilinear similarity of COLA is used to calculate the contrastive loss. The bilinear similarity between query $q_i, i \in \{1, \cdots, N\}$ and key $k_{ij}, j \in \{+, 1, 2, \cdots, K\}$ is calculated as follows:

$$s_{i,j} = q_i^T W k_{ij}.$$

The quantity W is a weight matrix representing the correlations among all the elements of q_i and k_{ij}. This matrix is updated through learning.

We use the infoNCE loss [6] to calculate the contrastive loss. We calculated the cross entropy by taking the negative logarithm of the softmax activation as follows:

$$\ell_i = -\log \frac{\exp(q_i^T W k_{i+})}{\sum_{j=1}^{K} \exp(q_i^T W k_{ij})}. \quad (2)$$

We use SimCLR's symmetric loss in our model. Equation (3) is a combination of (1) and (2). By changing the query q_i and the key k_{i+}, the matching loss is re-calculated, and the average of the two losses is obtained:

$$L = \frac{1}{2N} \sum_{i=1}^{N} \left[-\log \frac{\exp(q_i^T W k_{i+})}{\sum_{j=1}^{K} \exp(q_i^T W k_{ij})} - \log \frac{\exp(k_{i+}^T W q_i)}{\sum_{j=1}^{K} \exp(k_{i+}^T W k_{ij})} \right]. \quad (3)$$

At the end of each batch, the $2N$ keys generated from the current batch are input to the dynamic dictionary and the oldest $2N$ keys are out from the dynamic dictionary. The dynamic dictionary maintains a queue of constant size K greater than twice the batch size, $2N$, which is used as a negative pair in the next batch.

The encoder parameters are updated with the aim of minimizing the cross entropy at the end of each batch. Representing the encoder parameter by Θ_q and the momentum encoder parameter by Θ_k, we update the parameter Θ_k as follows:

$$\Theta_k \longleftarrow m\Theta_k + (1-m)\Theta_q,$$

where the quantity $m \in [0, 1)$ is called a momentum coefficient. This reduces the imbalance between the encoder and the momentum encoder.

A. ENCODER AND MOMENTUM ENCODER

ResMax [18] is used as the encoder and momentum encoder. ResMax uses the Max Feautre Map (MFM) layer of Light CNN (LCNN) [15] and the skip connection of ResNet. The MFM function divides the channel into two equal halves and chooses the largest of the two values in the same position. This MFM activation worked well in spoofing detection models, LCNN based models ranked 1st and 2nd in the ASVspoof 2017 and 2019 competitions [16], [17]. A schematic representation of the action of the MFM function is shown in Figure 3(a).

The structure of ResMax block and ResMax model architecture is illustrated in Figure 3(b) and Figure 3(c). The ResMax block adds the original input as a skip connection to one or two convolution layers with MFM activation. The output size and the number of parameters can be reduced by using the 2×2 max pooling layer in the ResMax block. ResMax model architecture consists of 9 ResMax blocks, a global average pooling layer, a dropout layer, and a fully connected layer with a softmax function. The ResMax block is described by four parameters f, k, l, and m. The parameter f represents the number of filters in the convolution layer, and k represents the kernel size (k, k) of the first convolution layer in the ResMax bloc. When $l = 1$, the ResMax block has one more convolution layer followed by MFM activation with $(1, 1)$ kernel, and when $l = 0$, it has only one convolution layer followed by MFM activation. When $m = 1$, the ResMax block has the maximum pooling layer after the skip connection. Both l and m are binary input parameters that have values of 0 or 1.

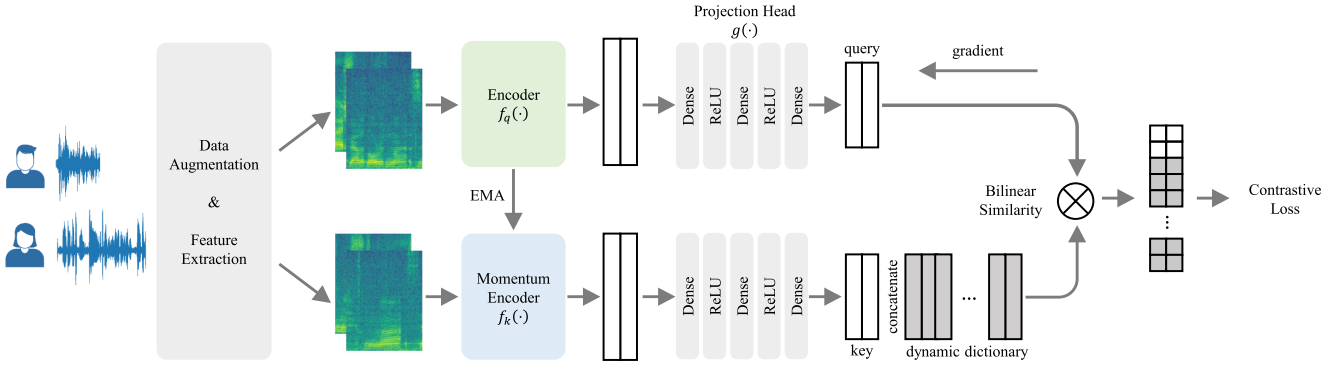


FIGURE 2. Self-supervised learning model architecture.

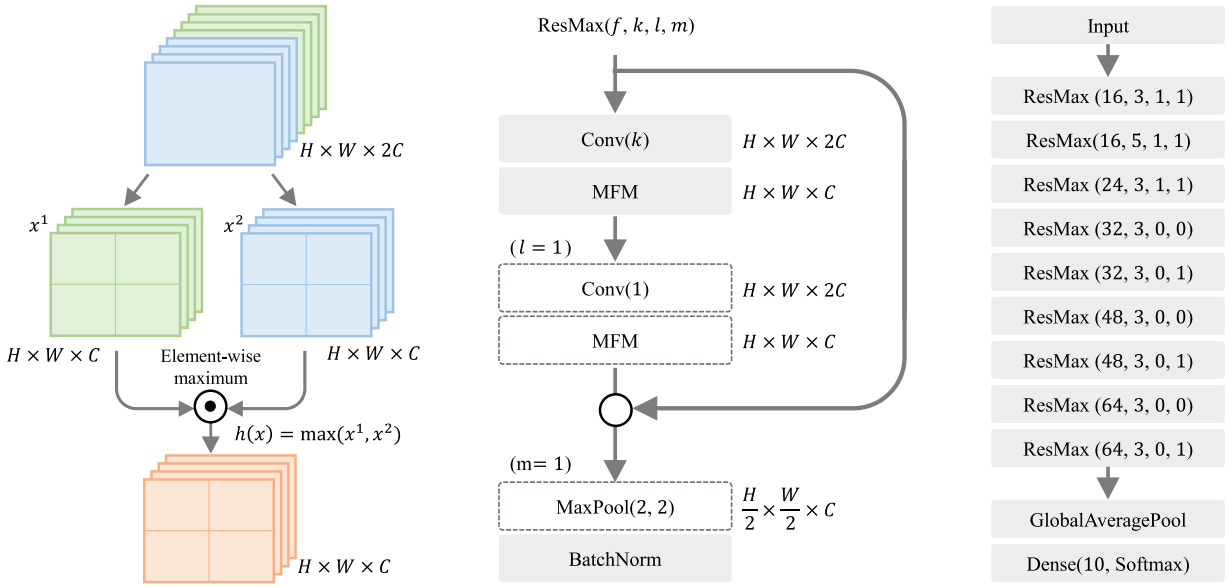


FIGURE 3. Residual network with the Max Feature Map (MFM) function: (a) MFM function, (b) ResMax block, (c) ResMax model architecture.

IV. EXPERIMENTS

A. DATASETS

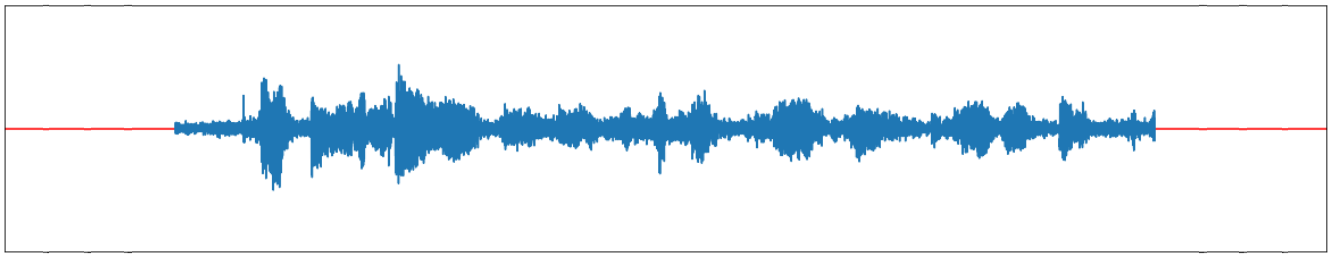
VoxCeleb [38] was used for pre-training. VoxCeleb is an audiovisual dataset that consists of voice clips (each clip being at least 3 seconds long) obtained from interviews with celebrities that have been uploaded to YouTube. The data set includes 153,416 utterances of 1,251 speakers extracted from 22,496 videos.

We used the ASVspoof 2019 dataset to detect voice spoofing attacks [11]. ASVspoof 2019 LA (for Logical Access) is used to detect TTS and VC attacks. ASVspoof 2019 LA consists of training, development, and evaluation datasets of 25,380 (2,580 bonafide, 22,800 spoof), 24,844 (2,548 bonafide, 22,296 spoof), and 71,237 utterances, respectively. ASVspoof 2019 PA (for Physical Access) is used to detect replay spoofing attacks. ASVspoof 2019 PA consists of training, development, and evaluation datasets of 54,000 (5,400 bonafide, 48,600 spoof), 29,700 (5,400 bonafide, 24,300 spoof), and 137,457 utterances, respectively.

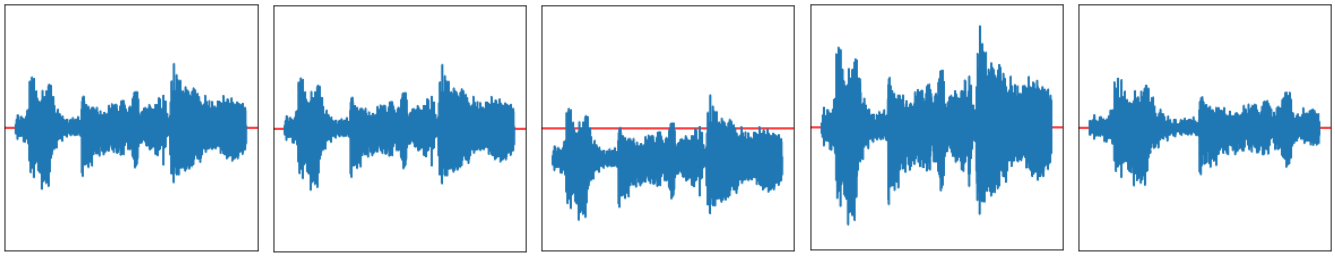
B. DATA AUGMENTATION

A random crop was a widely used option in all our contrastive learning models, and one or two following further augmentation techniques, such as additive noise, center shifting, dynamic range change, and speed change, were additionally considered. The function of each augmentation technique is briefly summarized as follows:

- Random Crop: Randomly cuts the original audio into a second long audio file.
- Additive Noise: Adds white noise with a normal distribution. The maximum amplitude of the added white noise is selected as either 0.001, 0.003, or 0.005 times the maximum amplitude of the original audio.
- Center Shifting: Randomly adds a shifting constant to the original audio. A shifting constant is taken from a uniform distribution with parameter $(-0.5k, 0.5k)$, where k is the maximum amplitude of the original audio.



(a) Original



(b) Random Crop (c) Addictive Noise (d) DRC (e) Volume Change (f) Speed Change

FIGURE 4. Data augmentation: (a) original, (b) random crop, (c) additive noise, (d) center shifting, (e) dynamic range change(DRC), (f) speed change.

- Dynamic Range Change: Randomly amplifies the amplitude of the original audio ranging from 1 to 3 times or ranging from 1 to 6 times.
- Speed Change: Adjusts the speed of the original audio by multiplying the signal with a value ranging from 0.8 to 1.2 or ranging from 0.9 to 1.1.

Figure 4 displays the audio waveforms resulting from the applications of these respective data augmentation techniques. The original audio wavelength is shown in Figure 4(a), the result of a random crop in Figure 4(b), and the results of applying additive noise, a center shifting, a dynamic range change, and a speed change over and above the random crop, in Figures 4(c)-4(f), respectively. These data augmentation techniques are considered for our self-supervised training.

C. IMPLEMENTATION DETAILS

A Short-Time Fourier Transform (STFT) or a constant-Q-transform (CQT) was used as audio features. The frame size of the STFT feature was 320, which corresponds to 20 *ms*, and overlapped by 50%. In addition, a Hanning window was used, with the number of Fast Fourier Transform (FFT) bins set to 256. The generated STFT features were logarithmic and normalized. The CQT feature was set to 12 octaves and 512 hop lengths, as in ResMax [18]. A Hanning window was used with the number of frequency bins set to 120. Logarithms were taken and normalized to the generated CQT features.

In order to create negative pairs, the size of the dynamic dictionary was set to $4,096$ and the encoder and momentum encoder used ResMax; global average pooling generated representations q and k_+ of size 64. The projection head consisted of three fully connected layers with 64 units; the first and second fully connected layers were followed by a Rectified Linear Unit (ReLU) activation function. The last fully connected layer was followed by layer normalization and the Tanh activation functions. The momentum coefficient was set at 0.99.

For the downstream voice spoofing detection, we used only the first 9 seconds of audio files. For audio files shorter than 9 seconds, the connection was repeated until 9 seconds had elapsed. For fine-tuning, the weight up to the first activation layer of the pre-trained projection head was continuously updated as the initial weight.

In both the PA and LA training datasets of ASVspoof 2019, the spoof datasets are about 10 times larger than the bonafide datasets. Thus, we used cost-sensitive learning to solve this class imbalance problem. The bonafide datasets were weighted by a factor of 5 when calculating the loss function.

All models used the Adam optimizer [39] with initial learning rate was 10^{-3}, reduced to 10^{-5} with sigmoidal decay. For pre-training, 200 epochs were trained with a batch size of 128, whereas 30 epochs were trained with a batch size of 16 for downstream tasks.

D. EVALUATION METRIC

The biometric system uses the Equal Error Rate (EER) as an index for evaluating performance. The ASVspoof challenge uses the same evaluation index. The EER is evaluated as the average error rate at the point where the False Acceptance Rate (FAR) and the False Rejection Rate (FRR) are equal. The FAR and FRR are calculated as follows:

$$\text{FAR} = \frac{\text{FP}}{\text{TP} + \text{FP}}, \quad \text{FRR} = \frac{\text{FN}}{\text{TN} + \text{FN}},$$

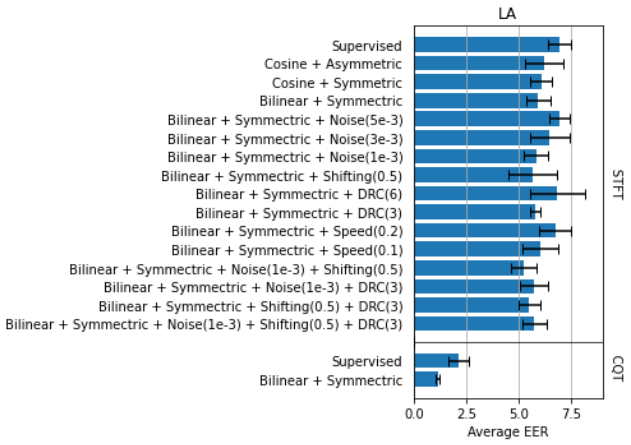
where TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative, respectively. In order to

mitigate bias caused by randomization, all experiments were repeated 10 times, and the mean and standard deviation were calculated.

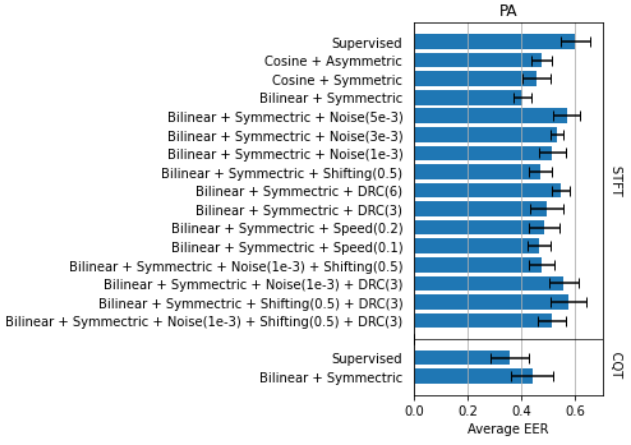
V. RESULTS

A. RESULTS FOR THE STFT FEATURE

A comparison of the classification performance achieved with ASVspoof 2019 data using the STFT feature is displayed in Table 1. The EER of the supervised learning process was 6.93% and 0.60% for the LA and PA, respectively. When pre-training was performed by contrast learning using cosine similarity and unsymmetric loss, the EER for the LA was 6.23%, confirming that the improvement achieved with supervised learning was significant, according to the t-test. (p-value$\leq$ 0.05). Similarly, the EER for the PA was 0.47%, confirming that the improvement was significant. When pre-trained with contrastive learning using cosine similarity and symmetry loss, the EERs of the LA and PA improved to 6.08% and 0.45%, respectively. The EER of the LA was improved to 5.92% when pre-training was performed with contrastive learning using both bilinear similarity and symmetric loss, but the significance of the performance was not confirmed compared to the case when neither bilinear similarity nor symmetric loss was used. On the other hand, the EER of the PA was 0.40%, confirming that the improvement was significant.



(a) LA



(b) PA

FIGURE 5. Barplot of averaged EER values for various augmentation modes, with standard deviation indicated.

When noise less than 10^{-3} times the maximum amplitude was added to the LA with data augmentation techniques, the EER improved to 5.83%. When the center shifting of the sound was adjusted to be small, the EER improved to 5.68%. Even when the amplitude of the sound was amplified by a factor of three, the EER was improved to 5.78%. The EER was 6.95% when the noise was less than 5×10^{-3} times the maximum amplitude and 6.47% when it was less than 3×10^{-3} times the maximum amplitude. These values were larger than before, when noise was less than 10^{-3} times the maximum amplitude, and the EER also increased as noise increased. When the amplitude was amplified by a factor of 6, the EER was 6.85%, which was a poorer performance level than when the amplification factor was 3. Even when the speed was adjusted to 0.2 times or 0.1 times slower or faster, the EER was significantly degraded to 6.74% and 6.03%, respectively. When both additive noise and center shifting were used, the EER was improved to 5.26%. The EER was increased to 5.71% when both additive noise and a dynamic range change were used. Even after both center shifting and a dynamic range change were used, the EER improved to 5.50%. In other words, performance was better when two augmentation strategies were used than when only one augmentation technique was used. On the other hand, when three augmentation techniques, i.e., additive noise, center shifting, and dynamic range change, were added at the same time, the EER was 5.75%. Therefore, in the LA, the performance was optimal when both additive noise and center shifting were used simultaneously as data augmentation techniques. Through a t-test, it was confirmed that the difference in performance compared to when only a random crop was used was significant (p-value$\leq$ 0.05). The data augmentation techniques, which had enhanced the performance in the LA were, also applied to the PA, but resulted in a diminished level of performance in the PA.

B. RESULTS FOR THE CQT FEATURE

A comparison of the classification performance using the CQT feature is displayed in Table 2. Since CQT takes a considerable amount of time to generate features, we only performed experiments with a random crop as an augmentation technique.

The EER of supervised learning was 2.13% and 0.35%, respectively, for the LA and PA, and when pre-trained with contrast learning using both bilinear similarity and symmetry loss, the performance in the LA was enhanced with a significant difference of 1.15%. However, in the PA case, the performance level decreased by 0.44%.

We conducted additional experiments with ASVspoof 2015 and ASVspoof 2017. ASVspoof 2015 deals with the logical

TABLE 1. Values of the Validation EER (%) and the Test EER (%) for the pre-training dataset is VoxCeleb and the downstream dataset ASVspoof 2019 with STFT.

			Augmentation					LA		PA	
	Similarity	Symmetric	Crop	Noise	Shifting	DRC	Speed	Val EER	Test EER	Val EER	Test EER
Supervised	–	–	–	–	–	–	–	0.00±0.00	6.93±0.55	0.33±0.04	0.60±0.05
Fine-tuned	Cosine		∨					0.00±0.00	6.23±0.90	0.30±0.04	0.47±0.03
Fine-tuned	Cosine	∨	∨					0.00±0.00	6.08±0.53	0.27±0.02	0.45±0.05
Fine-tuned	Bilinear	∨	∨					0.00±0.00	5.92±0.57	0.25±0.03	0.40±0.03
Fine-tuned	Bilinear	∨	∨	5e-3				0.00±0.00	6.95±0.48	0.29±0.02	0.56±0.04
Fine-tuned	Bilinear	∨	∨	3e-3				0.00±0.00	6.47±0.93	0.33±0.04	0.53±0.02
Fine-tuned	Bilinear	∨	∨	1e-3				0.00±0.00	5.83±0.56	0.31±0.04	0.51±0.05
Fine-tuned	Bilinear	∨	∨		0.5			0.00±0.00	5.68±0.78	0.26±0.02	0.47±0.04
Fine-tuned	Bilinear	∨	∨			6.0		0.00±0.00	6.85±1.29	0.33±0.04	0.51±0.05
Fine-tuned	Bilinear	∨	∨			3.0		0.00±0.00	5.78±0.22	0.25±0.04	0.49±0.06
Fine-tuned	Bilinear	∨	∨				0.2	0.00±0.00	6.74±0.86	0.30±0.05	0.48±0.05
Fine-tuned	Bilinear	∨	∨				0.1	0.00±0.00	6.03±0.46	0.27±0.03	0.46±0.04
Fine-tuned	Bilinear	∨	∨	1e-3	0.5			0.00±0.00	5.26±0.62	0.28±0.04	0.47±0.04
Fine-tuned	Bilinear	∨	∨	1e-3		3.0		0.00±0.00	5.71±0.67	0.29±0.03	0.55±0.05
Fine-tuned	Bilinear	∨	∨		0.5	3.0		0.00±0.00	5.50±0.51	0.31±0.04	0.57±0.06
Fine-tuned	Bilinear	∨	∨	1e-3	0.5	3.0		0.00±0.00	5.75±0.57	0.33±0.06	0.50±0.05

TABLE 2. Validation EER (%) and Test EER (%) when pre-training dataset is VoxCeleb and downstream dataset is ASVspoof 2019 with CQT.

	ASVspoof2015		ASVspoof2017		ASVspoof2019LA		ASVspoof2019PA	
	Val EER	Test EER	Val EER	Test EER	Val EER	Test EER	Val EER	Test EER
Supervised	0.00±0.00	3.70±0.64	23.05±4.44	29.65±3.36	0.47±0.04	2.13±0.46	0.20±0.05	0.35±0.07
Fine-tuned	0.51±1.14	2.71±0.47	35.77±3.27	42.26±1.92	0.29±0.03	1.09±0.20	0.32±0.05	0.44±0.07

access scenario (similar to ASVspoof 2019 LA set), and ASVspoof 2017 deals with the physical access scenario (similar to ASVspoof 2019 PA set). we used cost sensitive learning to solve this class imbalance problem. The bonafide datasets were weighted by a factor of 5 when calculating the loss function. In the case of ASVspoof 2015, we used cost sensitive learning to solve this class imbalance problem. The bonafide datasets were weighted by a factor of 2 when calculating the loss function. The EER showed performance improvement from 3.70% to 2.71%, while in the case of ASVspoof 2017, the performance was degraded from 29.65% to 42.26%.

VI. DISCUSSION

A. COMPARISON WITH EXISTING RESEARCH

Wang et al. [40] also investigated self-supervised implications on the voice spoofing detection problem. While our research focused on how to pre-train models using self-supervised learning, [40] focused on how to use existing pre-trained networks such as wav2vec 2.0 [35] and HuBert [36] to solve the spoofing detection problem. They experimented with using a convolutional neural network consisting of a bi-directional recurrent layer instead of a non-linear network as a projection head.

Wu et al. [41] proposed the model using Mockingjay [42], a self-supervised learning-based model, to generate a high-level feature for voice spoofing detection model to mitigate the vulnerability to adversarial attacks. In pre-training, they solve the masked-prediction task, randomly masking audio frames and then reconstructing them. Adversarial noise can be prevented by passing the audio feature through a pre-trained model before inputting it into the voice spoofing detection model.

VII. CONCLUSION

In this study, we showed that the performance of audio spoofing attack detection could be enhanced through pre-training based on contrastive learning. The VoxCeleb data set was used for pre-training, and a new contrast learning model was proposed by combining various previously proposed contrastive learning techniques, such as a dynamic dictionary, symmetric loss, and bilinear similarity. In addition, experiments were conducted to determine whether the performance was improved by using contrast learning coupled with five different data augmentation techniques: a random crop, additive noise, center shifting, dynamic range changes, and speed changes. LA and PA data from ASVspoof 2019 were used as voice spoofing attack detection data. As a result, when the STFT feature was used, it was confirmed that there was a significant performance improvement in the LA data when one or two from additive noise, center shifting, and dynamic range change were applied as augmentation techniques in conjunction with a random crop. When center shifting, random crop, and additive noise were used for the contrastive learning model in the LA scenario, the best result of 5.26% EER was obtained. This is 1.67% less EER than the baseline EER of 6.93%. In the PA scenario, the best result of 0.40% EER was obtained with random crop augmentation, 0.20% lower than the baseline of 0.60% EER. To obtain deeper insight into the potential advantages of using contrastive learning and the application of augmentation techniques as explored in this study, further experiments with more complex models, large epochs, and various data augmentation techniques should be conducted.

REFERENCES

[1] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, pp. 1–21, Mar. 2015.

[2] V. R. de Sa, "Learning classification with unlabeled data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 6. San Mateo, CA, USA: Morgan Kaufmann, 1994, pp. 112–119.

[3] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," 2020, *arXiv:2001.08361*.

[4] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent—A new approach to self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 21271–21284.

[5] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 857–876, Jan. 2021.

[6] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[7] E. Buteau and J. Lee, "Hey alexa, why do we use voice assistants? The driving factors of voice assistant technology use," *Commun. Res. Rep.*, vol. 38, no. 5, pp. 336–345, Oct. 2021.

[8] I.-Y. Kwak, J. H. Huh, S. T. Han, I. Kim, and J. Yoon, "Voice presentation attack detection through text-converted voice command analysis," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, May 2019, pp. 1–12.

[9] R. Togneri and D. Pullella, "An overview of speaker identification: Accuracy and robustness issues," *IEEE Circuits Syst. Mag.*, vol. 11, no. 2, pp. 23–61, 2nd Quart., 2011.

[10] G. R. Doddington, "Speaker recognition—Identifying people by their voices," *Proc. IEEE*, vol. 73, no. 11, pp. 1651–1664, Nov. 1985.

[11] 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*, Sep. 2019, pp. 1008–1012.

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

[13] 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*, Aug. 2017, pp. 2–6.

[14] M. Todisco, H. Delgado, and N. Evans, "Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification," *Comput. Speech Lang.*, vol. 45, pp. 516–535, Sep. 2017.

[15] X. Wu, R. He, Z. Sun, and T. Tan, "A light CNN for deep face representation with noisy labels," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2884–2896, Nov. 2018.

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

[17] G. Lavrentyeva, S. Novoselov, A. Tseren, M. Volkova, A. Gorlanov, and A. Kozlov, "STC antispoofing systems for the ASVspoof2019 challenge," in *Proc. Interspeech*, Sep. 2019, pp. 1033–1037.

[18] 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 *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 4837–4844.

[19] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 9729–9738.

[20] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020, *arXiv:2003.04297*.

[21] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[22] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 22243–22255.

[23] X. Chen and K. He, "Exploring simple Siamese representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15750–15758.

[24] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 9912–9924.

[25] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–16.

[26] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 649–666.

[27] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Computer Vision—ECCV 2016*. Cham, Switzerland: Springer, 2016, pp. 69–84.

[28] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1422–1430.

[29] E. Kharitonov, M. Rivière, G. Synnaeve, L. Wolf, P.-E. Mazarè, M. Douze, and E. Dupoux, "Data augmenting contrastive learning of speech representations in the time domain," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Jan. 2021, pp. 215–222.

[30] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 3875–3879.

[31] H. Al-Tahan and Y. Mohsenzadeh, "CLAR: Contrastive learning of auditory representations," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 2530–2538.

[32] D. Jiang, W. Li, M. Cao, W. Zou, and X. Li, "Speech SIMCLR: Combining contrastive and reconstruction objective for self-supervised speech representation learning," 2020, *arXiv:2010.13991*.

[33] W. Xia, C. Zhang, C. Weng, M. Yu, and D. Yu, "Self-supervised text-independent speaker verification using prototypical momentum contrastive learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6723–6727.

[34] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 776–780.

[35] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associate, 2020, pp. 12449–12460.

[36] W.-N. Hsu, B. Bolte, Y.-H.-H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 29, pp. 3451–3460, 2021.

[37] A. Srinivas, M. Laskin, and P. Abbeel, "CURL: Contrastive unsupervised representations for reinforcement learning," 2020, *arXiv:2004.04136*.

[38] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: A large-scale speaker identification dataset," in *Proc. Interspeech*, Aug. 2017, pp. 2616–2620, doi: 10.21437/Interspeech.2017-950.

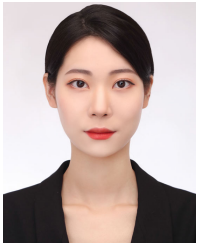
[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[40] X. Wang and J. Yamagishi, "Investigating self-supervised front ends for speech spoofing countermeasures," 2021, *arXiv:2111.07725*.

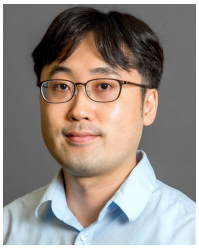
[41] H. Wu, A. T. Liu, and H.-Y. Lee, "Defense for black-box attacks on anti-spoofing models by self-supervised learning," in *Proc. Interspeech*, Oct. 2020, pp. 3780–3784.

[42] A. T. Liu, S.-W. Yang, P.-H. Chi, P.-C. Hsu, and H.-Y. Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 6419–6423.

YERIN LEE received the B.S. degree in mathematics from Incheon National University, Incheon, South Korea, in 2018, and the M.S. degree in statistics from Chung-Ang University, Seoul, South Korea, in August 2021. Since 2022, she has been with Woongjin Thinkbig, Edutech Labs. Her research interest includes sequential data analysis, such as time-series and audio data.



NARIN KIM received the B.S. degree in statistics from Sungshin Women's University, Seoul, South Korea, in 2020. She is currently pursuing the M.S. degree in statistics with Chung-Ang University. Her research interests include machine learning and deep learning models with audio and signal data. Her other interests include computer vision and structured data analysis.



JAEHONG JEONG received the Ph.D. degree in statistics from Texas A&M University, USA, in 2015. From 2015 to 2018, he was a Postdoctoral Fellow with the King Abdullah University of Science and Technology, Saudi Arabia. He was an Assistant Professor of statistics with the University of Maine, USA, from 2018 to 2020. He is currently an Assistant Professor with the Department of Mathematics, Hanyang University, South Korea. His primary research interests include spatio-temporal statistics and non-Gaussian processes for space–time data.

IL-YOUP KWAK (Member, IEEE) was born in Seoul, South Korea, in 1981. He received the B.S. degree in mathematics and the M.S. degree in statistics from Korea University and the Ph.D. degree in statistics from the University of Wisconsin–Madison, Wisconsin, USA, in 2014. He was a Postdoctoral Associate at the University of Minnesota–Twin Cities, from 2014 to 2017. He was with Samsung Research, from 2017 to 2019. Since 2019, he has been an Assistant Professor with the Applied Statistics Department, Chung-Ang University. His research interests include deep learning applied to audio, AI security, and statistical genetics.