IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 January 2023, accepted 21 January 2023, date of publication 25 January 2023, date of current version 31 January 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3239660*

RESEARCH ARTICLE

# WaveBYOL: Self-Supervised Learning for Audio Representation From Raw Waveforms

**SUNGHYUN KIM[1,2] AND YONG-HOON CHOI[1], (Member, IEEE)**
[1]School of Robotics, Kwangwoon University, Seoul 01897, South Korea
[2]NEOWIZ, Seongnam-si, Gyeonggi-do 13487, South Korea

Corresponding author: Yong-Hoon Choi (yhchoi@kw.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) through the Ministry of Science and ICT, under Grant 2021R1F1A1064080; in part by the Korea Institute for Advancement of Technology (KIAT) through the Ministry of Trade, Industry and Energy (MOTIE), under Grant P0017124; and in part by Kwangwoon University, in 2022.

**ABSTRACT** In this paper, we propose the WaveBYOL model, which can learn general-purpose audio representations directly from raw waveforms based on the bootstrap your own latent (BYOL) approach, a Siamese neural network architecture. WaveBYOL does not extract features in a handcrafted manner, and the model learns general-purpose audio representations from raw waveforms by itself. Thus, the model can be easily applied to various downstream tasks. The augmentation layer in the WaveBYOL model is designed to create various views from the time domain of the raw audio waveforms; the encoding layer is designed to learn representations by extracting features from the views, which are augmented audio waveforms. We assess the representations learned by WaveBYOL by conducting experiments with seven audio downstream tasks under both frozen-model evaluation and fine-tuning settings. The accuracy, precision, recall, and F1-score are observed as performance evaluation metrics of the proposed model, and the accuracy score is compared with those of the existing models. In most downstream tasks, WaveBYOL achieves competitive performance compared to that of the recently developed state-of-the-art models such as contrastive learning for audio (COLA), BYOL for audio (BYOL-A), self-supervised audio spectrogram transformer (SSAST), audio representation learning with teacher-student transformer (ATST), and DeLoRes. Our implementation and pretrained models are located on GitHub.

**INDEX TERMS** Self-supervised learning (SSL), audio waveform augmentation, audio representation.

## I. INTRODUCTION

Self-supervised learning is a methodology for learning generalized representations from large datasets without labels. To learn a meaningful representation from a dataset, a pretext task needs to be defined. A pretext task is defined as a task that is not directly useful, but it learns transferable representations from unlabeled datasets, creating a pretrained model. The pretrained model can be applied to various downstream tasks through transfer learning. The downstream task is a stage in which knowledge transfer is performed to address a specific problem.

Recently, a self-supervised learning approach has been successfully used in the computer vision domain. In particular, the Siamese neural network architecture [1] has become a widely used architecture for self-supervised learning. The Siamese architecture consists of two similar networks that share parameters. One of them is typically used as a training target for the other, comparing the representations extracted from two networks. However, the Siamese architecture has a collapsed representation problem in which all output values collapse into constants [2]. Various methodologies to alleviate collapsed representations have been proposed, such as contrastive learning [3]. Contrastive learning is a machine learning (ML) technique used to teach models which data points are similar or different to learn the general features of unlabeled datasets. The goal of contrastive learning is to learn an embedding space in which pairs of similar samples (i.e., positive samples) are kept close to each other while pairs of dissimilar samples (i.e., negative samples) are far apart.

The associate editor coordinating the review of this manuscript and approving it for publication was Aysegul Ucar.

Contrastive learning is a useful approach in self-supervised learning when working with unlabeled data.

SimCLR [4], [5] is a simple framework for contrastive learning of visual representations. Two different data augmentations are applied to the image; an augmented image from the same image is defined as a positive sample, and an augmented image from different images is defined as a negative sample. This framework effectively extracts visual representations with unlabeled images, but the performance of the model has a large deviation depending on the quantity and quality of negative samples. Usually, contrastive learning requires a very large batch size because the larger the number of negative samples is, the more meaningful representations that can be learned. SimCLR uses a large batch size of 8192 to include various negative samples. To stabilize the training process, the authors used the layer-wise adaptive rate scaling (LARS) optimizer [6], which is suitable for large batch sizes. Another way to reduce the batch size is to use a memory bank to avoid uploading negative samples to the batch [7]. It is a method that performs sampling by constructing a dictionary after storing the representations of all data in the memory bank. This includes the problem that a large number of negative samples can be used without being placed in a batch, but the negative samples are not updated. Momentum contrast (MoCo) [8], [9] provides a framework for unsupervised learning of visual representations with dynamic dictionary lookups. Compared the approach of memory banks, queue-based MoCo dictionary allows reuse of representations of mini-batches from the immediately preceding data. The advantage of MoCo over SimCLR is that MoCo separates batch size from negative samples. However, SimCLR requires large batch sizes to obtain enough negative samples, and performance degrades with decreasing batch sizes.

On the other hand, the bootstrap your own latent (BYOL) [10] approach utilizes a strategy to train the model by using only positive samples. BYOL uses two neural networks, an online and a targeted network, that interact and learn from each other. Starting from an augmented view of an image, BYOL trains an online network to predict the representation of the target network for different augmented views of the same image. BYOL addresses the collapsed representation problem by adding a predictive layer to the online network so that the two networks can have slightly different structures. Moreover, since negative samples are not used, it is important to apply effective *data augmentation* techniques to generate different types of views. The target network does not perform backpropagation by itself and adopts an exponential moving average (EMA) strategy so that the weights of the online network can be updated at a certain rate at regular intervals. This strategy also prevents representation collapse while maintaining the weight of the target network. BYOL has achieved the best performance in the computer vision domain.

In the audio domain, recent studies have been conducted to extend the model proposed in the computer vision domain to suit the characteristics of audio input. Recent models to which contrastive learning is applied in the audio domain are contrastive predictive coding (CPC) [11] and contrastive learning for audio (COLA) [12]. CPC is an autoregressive model that uses past audio segments to generate a context vector and learns by comparing future and past representations. COLA is a model extension of SimCLR, a self-supervised pretraining approach for learning general-purpose representations of audio. When training the model, the audio segment extracted from the same audio clip is defined as a positive sample, and the audio segment extracted from different audio clips is defined as a negative sample. Since COLA also has a contrastive learning architecture, the quantity and quality of negative samples have a large effect on model training. Self-supervised audio spectrogram transformer (SSAST) [13], [14] is a transformer-based self-supervised learning model, and the authors proposed a masked-spectrogram patch modeling technique. Decorrelating latent spaces for low-resource audio representation learning (DeLoRes) [15] is a framework consisting of two simple self-supervised pretraining methodologies for learning general-purpose audio representations of speech and sound. Inspired by the Barlow Twins framework [16], the authors used a redundancy reduction-based loss function to make the computed cross-correlation matrix as close to the identity matrix as possible regarding the embeddings of the augmented sample pairs of the same audio segments. DeLoRes also uses the same augmentation module as BYOL for audio (BYOL-A) [17], [18].

BYOL-A [17] is a general-purpose audio representation learning model based on BYOL. Since this model is extended based on BYOL, negative pairs are not used for model training. A single audio segment extracted from an arbitrary position in the audio clip is used as the input to the model. The extracted single audio segment is converted into a log-mel spectrogram and a *view* is created through augmentation. In BYOL-A, mix-up, random resize crop, and normalization block were adopted to create various views. Audio representation learning with teacher-student transformer (ATST) [19] is a transformer-based teacher-student self-supervised learning model. ATST adopts transformer encoder into the baseline teacher-student scheme of BYOL-A [17]. ATST outperforms BYOL-A's convolutional neural network (CNN) encoder in learning the long-term semantic information contained in speech. BYOL-A uses one short segment to create a positive pair, while ATST uses two different long segments. This is better suited for a transformer where the network needs to learn longer time dependencies and match more distinct positive pairs generated from two segments. ATST has achieved state-of-the-art results on various audio classification benchmarks.

The performance of any ML model depends on the features on which the training and testing processes are performed. Hence feature extraction is one of the most vital parts of an ML process [20]. Audio representation models such as COLA, DeLoRes, SSAST, ATST, and BYOL-A convert raw audio waveforms into intermediate representations with

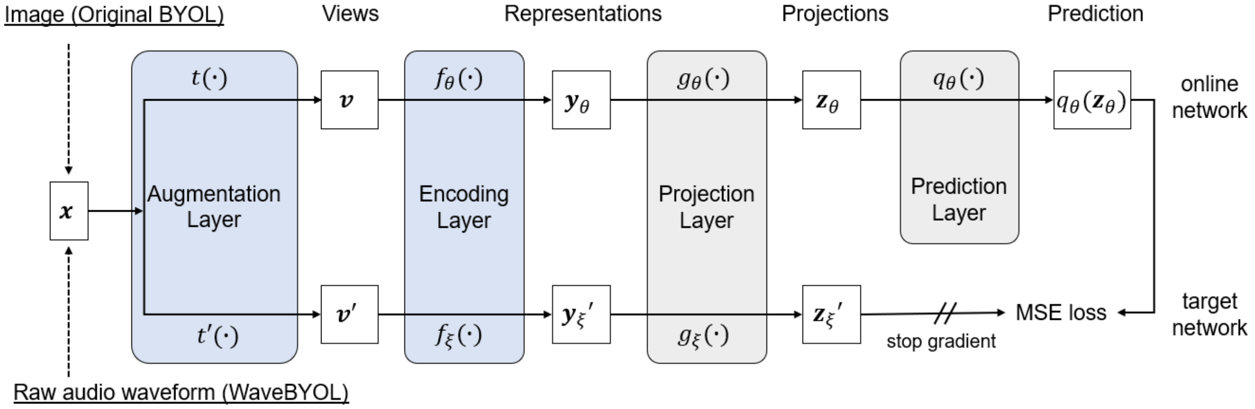

**FIGURE 1.** Original BYOL and WaveBYOL architecture overview.

handcrafted features such as log-mel spectrograms and use them as model inputs. Many studies have converted raw waveforms into spectrograms and applied various augmentation techniques such as random resizing, cropping, and SpecAugment [21]. SpecAugment is an augmentation technique that erases some areas of time and frequency from a spectrogram. However, handcrafted feature extraction may not be optimal for learning general-purpose audio representations [20].

In this paper, we propose a model that can learn representations through various views while directly using raw waveforms as input. The key contributions of this paper are as follows.

- We propose using *raw waveforms* as direct inputs to models learning general-purpose audio representations. Unlike BYOL-A, the proposed model does not use an intermediate representation in which raw waveforms are converted into spectrograms.
- We propose a self-supervised learning model called *WaveBYOL*. We propose an augmentation layer for generating various views from raw waveforms and an encoding layer for learning meaningful representations. Although performance differences are observed depending on the downstream tasks, WaveBYOL generally shows competitive performance compared to the previously proposed models [12], [14], [15], [17], [18], [19].
- Ablation studies are conducted to verify the contribution of each component and their combinations.

The rest of this paper is organized as follows. Section II describes BYOL and the architecture of the proposed model. Section III presents the utilized datasets, training, and performance evaluation. Section IV contains the ablation studies. Finally, Section V concludes the paper.

## II. MODEL DEVELOPMENT

The overall architecture of WaveBYOL proposed in this paper follows the BYOL [10] structure as shown in Figure 1. We expand the BYOL to learn audio representations $\boldsymbol{y}_\theta$ from raw waveforms without the use of negative samples.

### A. BYOL

BYOL [10] consists of an online network and a target network. The online network is defined by a set of weights $\theta$ and comprises three neural network layers: an encoding layer $f_\theta$, a projection layer $g_\theta$, and a prediction layer $q_\theta$. The target network has the same architecture as the online network (but without the prediction layer) and uses a different set of weights $\xi$. Given a set of raw audio waveforms $\mathcal{A}$, for a raw audio waveform $\boldsymbol{x} \sim \mathcal{A}$ sampled uniformly from $\mathcal{A}$, BYOL (as well as WaveBYOL) produces two augmented views $\boldsymbol{v} \triangleq t(\boldsymbol{x})$ and $\boldsymbol{v}' \triangleq t'(\boldsymbol{x})$ applying audio augmentations to $\boldsymbol{x}$. From the first augmented view $\boldsymbol{v}$, the online network outputs a representation $\boldsymbol{y}_\theta \triangleq f_\theta(\boldsymbol{v})$ and a projection $z_\theta \triangleq g_\theta(\boldsymbol{y}_\theta)$. The target network outputs the target representation $\boldsymbol{y}'_\xi \triangleq f_\xi(\boldsymbol{v}')$ and the target projection $z'_\xi \triangleq g_\xi(\boldsymbol{y}'_\xi)$ from the second augmented view $\boldsymbol{v}'$. Then, the model L2-normalizes both $\boldsymbol{q}_\theta(z_\theta)$ and $z'_\xi$ to $\overline{q_\theta}\,(z_\theta) = \boldsymbol{q}_\theta(z_\theta)/\left\|\boldsymbol{q}_\theta(z_\theta)\right\|_2$ and $\overline{z_\xi}' = z'_\xi/\left\|z'_\xi\right\|_2$. Because this prediction layer applies only to online networks, the architecture is asymmetric between online and target pipelines. Finally, the model defines the mean squared error between the L2-normalized predictions $\overline{q_\theta}\,(z_\theta)$ and target predictions $\overline{z_\xi}'$ as

$$\mathcal{L}^{a}_{\theta,\xi} = \left\|\overline{q_\theta}\,(z_\theta) - \overline{z_\xi}'\right\|_2^2 = 2 - \frac{2\boldsymbol{q}_\theta\,(z_\theta)^{\mathrm{T}} z'_\xi}{\left\|\boldsymbol{q}_\theta\,(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}. \quad (1)$$

BYOL symmetrizes the loss $\mathcal{L}^{a}_{\theta,\xi}$ in (1) by separately feeding $\boldsymbol{v}'$ to the online network and $\boldsymbol{v}$ to the target network to compute $\mathcal{L}^{b}_{\theta,\xi}$. At each training step, it performs a stochastic optimization step to minimize $\mathcal{L}^{Total}_{\theta,\xi} = \mathcal{L}^{a}_{\theta,\xi} + \mathcal{L}^{b}_{\theta,\xi}$ with respect to $\theta$ only but not $\xi$. The parameter $\xi$ of the target network is the EMA of the online network parameter $\theta$. Given a target decay rate $\alpha \in [0, 1]$, it performs the following updates after every training step: $\xi \leftarrow \alpha\xi + (1-\alpha)\,\theta$. The target network updates the weights without backpropagation. In practice, the part where the model learns representations is $f_\theta(\cdot)$ of the online network and is used later as a pretrained model for the downstream task.

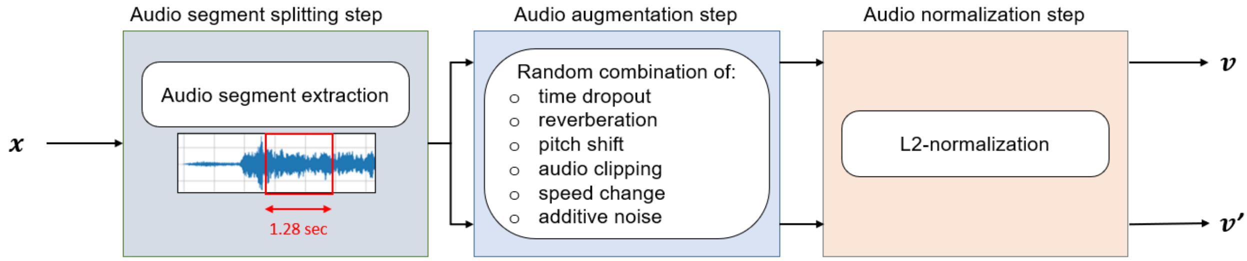

**FIGURE 2.** Augmentation Layer architecture of WaveBYOL.

### B. WaveBYOL AUGMENTATION LAYER

The augmentation layer of the WaveBYOL consists of three steps, as shown in Figure 2. In the first step, an audio segment of 1.28 seconds is extracted from a raw waveform at an arbitrary location. A single audio segment length of 1.28 seconds is the same as the audio segment length used by wav2vec [22] and applies only to the pretext task. For each downstream task, the average audio segment length of that dataset is applied.

In the second step, time dropout, reverberation, pitch shift, audio clipping, speed change, and additive noise are applied to an audio segment in any number and order to generate an augmented audio segment. Time dropout is applied to prevent the encoder from overfitting the dataset by removing random time periods between 0 and 0.5 s from the raw waveform. Additive noise is a method of adding noise to the background of the original sound source and is used to enable the encoder to separate the background and foreground. We use the music, speech, and noise corpus (MUSAN) dataset [23] and randomly select a signal-to-noise ratio between 5.0 and 20.0 dB. Reverberation is a method of adding reverberation that is generated by reflections in a specific space to audio, and it is used to enable the encoder to find real sound in response to reverberation. We use random values between 50.0 and 100.0 $m^3$ for the size of the space. Pitch shifting is a technique in which the original pitch of a sound is raised or lowered. The applied change in the pitch is an integer sampled uniformly between $-300$ and $+300$, measured by 1/100 of a tone. The speed change coefficient is randomly selected from $\{0.95, 0.93, 0.9, 0.85, 0.83, 0.83, 0.8, 0.75, 0.6, 0.5\}$ speeds. That is, if 0.75 is selected, the speed becomes 3/4 of the original speed. The audio clipping we applied is distortion of the waveform to cut 0-100% based on the maximum amplitude of the audio segment. These six augmentation techniques help to generate *various views* of the audio segment. Finally, in the audio normalization step, the augmented audio segment is L2-normalized.

### C. WaveBYOL ENCODING LAYER

The encoding layer of the WaveBYOL consists of multiple steps, as shown in Figure 3. The feature extractor, the first sublayer of the WaveBYOL encoding layer, extracts features from augmented audio segments that have been passed through the augmentation layer, replacing the typical handcrafted methods. The existing handcrafted methods focus on feature extraction processes that are optimized for specific tasks, but the proposed model allows the encoder to directly extract general-purpose audio features from raw waveforms. The learned general-purpose audio representations can be optimized for various tasks during fine-tuning.

The feature extractor consists of $S$ stacks with $B$ blocks in each (in Figure 3, $B = 5$). Each block contains 1D convolution, 1D batch normalization, and ReLU activation functions, as shown in Figure 3. It focuses on analyzing the input components of a specific frequency range using a 1D convolution layer with a kernel size of $k_\ell$ from the input of each block $\ell$. Larger $k_\ell$ values tend to cut more high-frequency components from the input of the block, forcing the block to focus on analyzing low-frequency content. Blocks with larger kernel sizes $k_\ell$ help to focus on learning low-frequency features. The feature extractor is implemented based on the convolutional network of wav2vec [22].

The segmentation and reassembly (SAR) layer divides the output of each stack into three segments of equal length, takes one segment from each stack, and reassembles it into a structure with three channels. The segments used for reassembly do not overlap each other on the time axis. Since the number of stacks is one (i.e., $S = 1$) in the current setup, we take all three segments from the stack to create a feature with a three-channel structure. Then, the augmented representations are L2-normalized.

In the computer vision domain, various methods, such as context prediction [24], rotation [25], jigsaw puzzle [26], colorization [27], and inpainting [28], are used so that the encoder can learn general-purpose representations. In the audio field, there is also a study in which a jigsaw puzzle is applied [29]. For example, in [14] intermediate representations are divided into $n \times n$ patches and sequentially used as inputs to the encoder. Inspired by these methods, in this layer, the semantic region of the audio feature is transformed so that the encoder can learn the general-purpose audio representation.

Finally, in the feature encoder, the model is designed to learn representations from the two-dimensional features. The feature encoding module consists of repeated two-layer 2D convolution, 2D batch normalization, and ReLU activation functions. Afterward, adaptive max pooling and adaptive average pooling are taken to pass through the projection

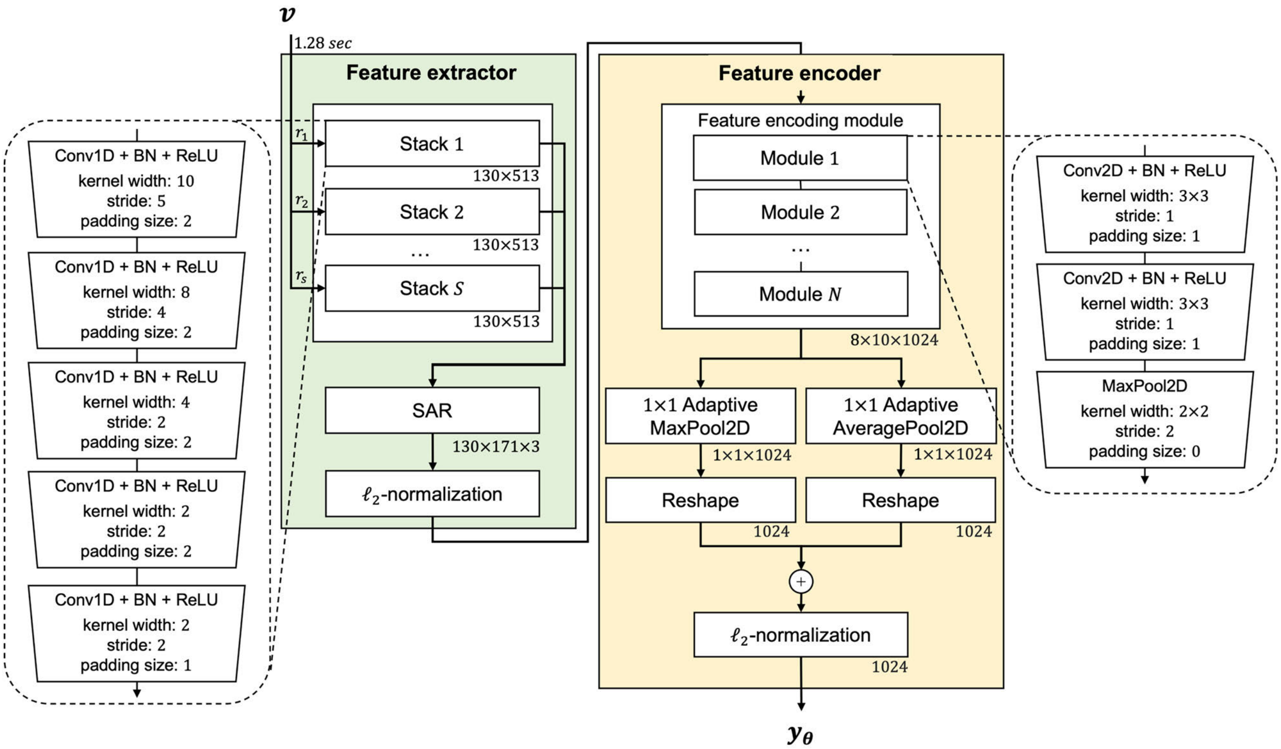

**FIGURE 3.** Encoding Layer architecture of WaveBYOL. The feature extractor on the left and the feature encoder on the right. In the feature extractor, $r_l$ is the audio sampling rate.

layer, and then elementwise adding and L2-normalization are performed to generate an embedding vector.

## III. MODEL TRAINING AND PERFORMANCE EVALUATION

We evaluate the performance of our self-supervised representation learned on FSD50K [30] under two settings: frozen-model evaluation and fine-tuning. In frozen-model evaluation, a linear classifier with a multilayer perceptron (MLP) layer is trained to classify a new dataset based on top of the frozen pretrained network, and in fine-tuning, we allow all weights to vary during training. In the frozen-model evaluation experiment, WaveBYOL is compared with COLA [12], DeLoRes [15], BYOL-A [17], [18], and ATST [19], and in the fine-tuning experiment, it is compared with COLA, DeLoRes, SSAST [14], and ATST.

### A. DATASET

Our study for unsupervised pretraining, which trains the encoder network $f_\theta(\cdot)$ without labels, is done by using the FSD50K [30] dataset. FSD50K is an open dataset containing over 51,000 audio clips, corresponding to a total of 108.3 hours of manually labeled audio by using 200 classes drawn from the AudioSet [31] ontology. AudioSet was released in 2017 to address the shortage of large-scale sound event datasets. It consists of 5,731 hours of data and is being used in various fields. However, AudioSet is not an open dataset because it consists of audio tracks taken from YouTube videos. Additionally, the video may disappear at the request of a YouTuber, making it difficult to use as a benchmark dataset. Currently, our model is trained using the FSD50K dataset.

We assess the performance of the representation from WaveBYOL after self-supervised pretraining on the training set of the FSD50K dataset. We evaluate it on other tasks, including UrbanSound8K (US8K) [32] and ESC-50 [33] for sound classification, VoxCeleb1 [34] for speaker identification, VoxForge [35] for language identification, SpeechCommandV2 (SPCV2) [36] for keyword recognition, the Ryerson audio-visual database of emotional speech and song (RAVDESS) [37] for emotion recognition, and NSynth [38] for musical instrument identification. For the US8K dataset [32], a predefined 10 folds without shuffling of the data is used, and a 10-fold cross-validation is performed, as indicated in the instructions. The VoxCeleb1 [34] dataset contains both development and test sets. We split the development set 4:1 to use for training and validation. For the rest of the dataset, we put 56% of the data in the training set, 19% in the validation set, and 25% in the test set. The sampling rate of all data used for training and testing is set to 16,000 Hz.

### B. MODEL SETUP

For the implementation of the proposed WaveBYOL model, we used the Torchaudio library of the PyTorch framework. Utilizing the WavAugment library [39], we implemented six raw waveform augmentation techniques of the augmentation layer. The parameters of the feature extractor in the encoding layer are set as follows. The number of feature extraction blocks is 5, the kernel widths of the convolution layer are (10, 8, 4, 2, 2), the strides are (5, 4, 2, 2, 2), the zero-padding sizes are (2, 2, 2, 2, 1), and the output consists of 513 channels. This structure is identical to that of the wav2vec [22] encoder, except that the kernel sizes of the 4th and 5th convolution layers are 2, which is smaller than the 4 in wav2vec. The next step is to reshape the audio feature to have a 3-channel shape. Starting with the first segment, the remaining segments are stacked down in order, and normalized features are obtained through L2-normalization. The normalized segments are fed as inputs to the feature encoder as depicted in Figure 3. There are 4 feature encoding modules in this step. The kernel widths of the 2D convolution of each feature encoding module are (3, 3), the strides are (1, 1), and the zero-padding sizes are (1, 1). The channel sizes of the first feature encoding module are (64, 128), the channel sizes of the second module are (256, 512), the channel sizes of the third module are (512, 512), and the channel sizes of the last module are (1024, 1024). For 2D max pooling, the filter width is set to 2, and the stride is set to 2 without zero-padding and dilation. Then, 2D adaptive max pooling and 2D adaptive average pooling are applied followed by a reshape block to produce a 1 × 1 output with 1024 channels. Each output is elementwise added to create embeddings.

The projection and prediction layers have MLP structures, and the structures of the two layers are identical to each other. Each MLP consists of a linear layer with an output size of 4096 followed by batch normalization, ReLU, and a final linear layer with an output dimension of 4096. The decay factor $\alpha$ of BYOL is raised to a value close to 1 by using cosine annealing according to iteration but is fixed to 0.99 in the proposed WaveBYOL model. AdamP [40] is used as an optimizer for training WaveBYOL. AdamP can suppress excessive weight norm growth; it removes the gradient component parallel to the direction of the weight generated by momentum through projection. The learning rate used for training is 0.0001, the batch size is 64, the epoch is 200, and the weight decay is set to $1.5 \times 10^{-6}$.

We manually tune the hyperparameters for the WaveBYOL framework. The hyperparameters used in WaveBYOL are summarized in Table 1. We use Docker on Ubuntu 18.04 LTS. One Tesla V100 GPU is used for training WaveBYOL. Our implementation and pretrained models are given on GitHub [41].

### C. DOWNSTREAM SETUP

Both frozen-model evaluation and fine-tuning are performed by adding one MLP layer to the output of the pretrained encoder. The structure of the MLP layer is the same as that of the projection layer, and the output dimension of the last linear layer is the number of classes in the given dataset. The frozen-model evaluation freezes the encoder weights so that they are not updated, and only the MLP layer is optimized for the dataset. In this case, the learning rate is set to 0.0008, and weight decay value for regularization is set to $1.5 \times 10^{-6}$. Fine-tuning enables backpropagation for both the encoder and the MLP layers. In this case, the learning rate is set to 0.00001, and the weight decay is $1.5 \times 10^{-6}$. In both evaluations, the model is trained for up to 100 epochs, and the training process stops early if no loss decrease is detected over 10 epochs. The tests are performed using the model trained up to the point in time when an early stop is detected. The input audio length of each task is set as the average length of the dataset for each task.

**TABLE 1. Considered hyperparameters.**

| Layer | Hyperparameters | Considered values |
|---|---|---|
| Common (pre-trained model) | Decay rate ($\alpha$) | 0.99 |
| | Initial learning rate | 0.0001 |
| | Batch size | {64} |
| | Activation function | ReLU |
| | Number of training epochs | 200 |
| | Weight decay | $1.5 \times 10^{-6}$ |
| | MLP output size | 4096 |
| Augmentation layer | Time dropout ($\tau$) | $\tau \in [0, 0.5]$ s |
| | SNR for additive noise ($S/N$) | $S/N \in [5.0, 20.0]$ dB |
| | Room size for reverberation ($s$) | $s \in [50, 100]$ $m^3$ |
| | Pitch shifting measured by 1/100 of a tone ($p$) | $p \in [-300, 300]$ |
| | Speech changes coefficient ($v$) | $v \in \{0.95, 0.93, 0.9, 0.85, 0.83, 0.83, 0.8, 0.75, 0.6, 0.5\}$ |
| | Audio clipping ratio ($r$) | $r \in [0, 100]\%$ |
| Encoding layer | Data sampling rates ($r_l$) | 16000 Hz |
| | Number of stacks ($S$) | 1 |
| | Number of feature extraction blocks ($B$) | 5 |
| | Conv1D kernel sizes | {10, 8, 4, 2, 2} |
| | Conv1D strides | {5, 4, 2, 2, 2} |
| | Conv1D zero-paddings sizes | {2, 2, 2, 2, 1} |
| | Number of feature encoding modules ($N$) | 4 |
| | Conv2D kernel widths | {3} |
| | Conv2D strides | {1} |
| | Conv2D depths | {64, 128, 256, 512, 1024} |
| | 2D pooling kernel size | 2 |
| | 2D pooling stride | 2 |

### D. RESULTS

Table 2 shows the results of comparing the existing method through a frozen-model evaluation with the proposed WaveBYOL. The dataset for the pretrained models of COLA, DeLoRes, ATST, and BYOL-A is AudioSet [31], and the dataset for pretrained WaveBYOL model is FSD50K [30]. AudioSet is a dataset that is 41 times larger in terms of number of audio clips and 53 times larger in terms of total duration than FSD50K. The input format of the COLA, DeLoRes, and BYOL-A models is a log-mel spectrogram, the input format of the ATST model is a mel spectrogram, and the input format of the WaveBYOL model is a raw waveform.

**TABLE 2.** Comparing the accuracies (%) of the proposed WaveBYOL with those of the existing models under the frozen-model evaluation setting.

| Model | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|
| | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| COLA [12] | - | 29.9 | 62.4 | - | - | 71.3 | 63.4 |
| BYOL-A [17] | 79.7 | 57.6 | 93.1 | **83.2** | - | 93.3 | 73.1 |
| BYOL-A* [18] | 79.1 | 40.1 | 92.2 | - | - | 90.2 | 74.1 |
| DeLoRes [15] | - | 31.2 | 80.0 | - | - | 76.5 | 66.3 |
| ATST [19] | **84.1** | **72.0** | **95.1** | - | - | - | **75.6** |
| WaveBYOL** | 54.7 | 56.4 | 87.0 | 80.4 | **73.1** | **93.7** | 68.1 |

* BYOL-A [17] is trained with AudioSet [31]. BYOL-A* [18] is trained with FSD50K [30].
** The frozen-model evaluation of WaveBYOL is not a standard linear evaluation [4] but rather utilizes a two-layered MLP.

**TABLE 3.** Comparing the accuracies (%) of the proposed WaveBYOL with those of the existing models under the fine-tuning setting.

| Model | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|
| | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| COLA [12] | - | 37.7 | 95.5 | - | - | 82.9 | 73.0 |
| SSAST [14] | - | 66.6 | **98.2** | - | - | - | **-** |
| DeLoRes [15] | - | 60.3 | 95.9 | - | - | 95.6 | **78.6** |
| ATST [19] | - | **94.3** | 98.0 | - | - | - | - |
| WaveBYOL | **62.9** | 92.9 | 95.9 | **83.4** | **83.5** | **99.2** | 71.5 |

COLA is trained using contrastive learning, BYOL-A is trained using BYOL, and DeLoRes is trained using Barlow Twins. For the results of the existing model, the results published in the relevant papers are referred to. As a result of the experiment, WaveBYOL shows the best performance on the VoxForge dataset but the lowest performance on the US8K dataset. For the rest of the dataset, it shows moderate performance compared to that of the recent state-of-the-art models. WaveBYOL has confirmed that it can directly extract features and can learn useful representations from raw waveforms, even though it is trained with a smaller dataset.

The VoxCeleb1 [34] dataset contains 153,514 utterances for 1,251 celebrities extracted from videos uploaded to YouTube with an average duration of 8.2 s. Because the number of classes to classify is quite large, the 56.4% accuracy from the WaveBYOL model shown in the speaker identification problem is a competitive performance compared to that of the other models. On the SPCV2 [36] dataset, which is a keyword recognition dataset, the accuracy of our WaveBYOL is slightly lower than that of the existing models. WaveBYOL uses an augmented raw waveform segment with a duration of 1.28 seconds, whereas the average audio segment length of SPCV2 is 1 second. As this is shorter than the audio segment length used for WaveBYOL training, it seems that there is a limitation with regard to learning representations. The NSynth [38] dataset in the musical instrument identification area has a large dataset imbalance, so it seems that learning about the characteristics of each class is insufficient. On the NSynth dataset, the amount of data in each class differs by up to 5 times or more. The performance achieved on the RAVDESS dataset is not shown in Table 2 because it is not tested with the comparative models.

Table 3 shows the results of comparing the accuracy of the existing models and WaveBYOL when using fine-tuning. In this experiment, all the existing models use AudioSet as the training dataset for the pretext task, and features are extracted from intermediate representations such as mel spectrograms. The input format of the COLA, DeLoRes, and WaveBYOL models is the same as that of the previous experiment, and the input format of the SSAST model is a log-mel spectrogram. The results of the existing models are derived from their original published papers. As shown in Table 3, the proposed WaveBYOL model shows accuracies that are comparable to the state-of-the-art results achieved in the VoxCeleb1 [32], SPCV2 [34] and VoxForge [33] downstream tasks. In particular, WaveBYOL achieves a great performance improvement on the VoxForge dataset for used for language identification. WaveBYOL achieves a certain level of accuracy without using intermediate representations such as mel spectrograms. It can be seen that the model itself learns meaningful general audio representations from raw waveforms. Compared to the existing models, WaveBYOL can extract features and learn representations directly from raw waveforms, so all weights are optimized from the feature extraction step to the feature encoding step during fine-tuning to fit the downstream task.

In the frozen-model evaluation, only the MLP layer is trained with the weights frozen, so the number of weights that the model can fit is very small. On the other hand, fine-tuning shows relatively high accuracy because the model fine-tunes

**TABLE 4.** Performance evaluation of the proposed WaveBYOL model under the frozen-model evaluation setting.

| Metrics | | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| Macro average | Precision | 0.523 | 0.564 | 0.865 | 0.825 | 0.728 | 0.926 | 0.646 |
| | Recall | 0.501 | 0.511 | 0.860 | 0.804 | 0.753 | 0.921 | 0.633 |
| | F1-score | 0.511 | 0.506 | 0.861 | 0.803 | 0.734 | 0.923 | 0.629 |
| Weighted average | Precision | 0.554 | 0.564 | 0.873 | 0.825 | 0.730 | 0.943 | 0.705 |
| | Recall | 0.516 | 0.564 | 0.870 | 0.804 | 0.731 | 0.943 | 0.681 |
| | F1-score | 0.534 | 0.564 | 0.870 | 0.803 | 0.724 | 0.943 | 0.684 |

**TABLE 5.** Performance evaluation of the proposed WaveBYOL model under the fine-tuning setting.

| Metrics | | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| Macro average | Precision | 0.633 | 0.921 | 0.956 | 0.845 | 0.852 | 0.990 | 0.664 |
| | Recall | 0.602 | 0.912 | 0.954 | 0.834 | 0.836 | 0.990 | 0.658 |
| | F1-score | 0.617 | 0.910 | 0.954 | 0.834 | 0.842 | 0.990 | 0.652 |
| Weighted average | Precision | 0.642 | 0.936 | 0.959 | 0.845 | 0.841 | 0.992 | 0.736 |
| | Recall | 0.610 | 0.929 | 0.959 | 0.834 | 0.835 | 0.992 | 0.715 |
| | F1-score | 0.625 | 0.928 | 0.959 | 0.834 | 0.836 | 0.992 | 0.714 |

the pretrained model and trains the MLP layer. Thus, the model can learn representations that are more suitable for downstream tasks.

Tables 4 and 5 are the evaluation results produced by WaveBYOL with respect to the precision, recall, and F1-score metrics for the frozen-model evaluation and the fine-tuning of downstream tasks. Since the weighted-average F1-score is a more useful performance evaluation metric for an imbalanced dataset, we observe both the macro-average and weighted-average performance metrics. The weighted-average weights each class value with its proportion in the dataset. As a result of the experiment, the difference between precision and recall is very small and predicts uniformly without bias in all downstream tasks. Additionally, since the difference between accuracy and F1-score is small, the accuracy values of Tables 2 and 3 can be trusted. In Table 4, since the difference between precision and recall is less than 0.053, the proposed model accurately predicts across all classes even in an imbalanced dataset. In particular, it shows very high recall and precision values in language identification.

Table 5 shows the results of fine-tuning, which yields higher performance than the frozen-model evaluation. In addition, the difference between precision and recall is 0.032 or less, making very stable inferences in all classes. In particular, the prediction performance is excellent and stable on the VoxForge dataset, a language identification dataset, and the SPCV2 dataset, a keyword recognition dataset.

## IV. ABLATION STUDY

We believe that the advantages of the WaveBYOL architecture are rooted in its end-to-end feature extraction nature without using handcrafted intermediate representations. In this experiment, six augmentation techniques applied to the augmentation layer are evaluated to determine their contribution to WaveBYOL model training. In addition, we check how much the normalization applied to the augmentation layer and encoding layer affect model training.

Table 6 shows the results of removing the data augmentation techniques one by one after setting the frozen-model evaluation with a pretrained model trained up to 100 epochs. All parameters and the environment of the pretrained model are the same as those in Table 1 except for the number of training epochs. As shown in Table 6, among the six augmentation techniques, the factors that have the greatest influence on the training of the WaveBYOL model are the order of pitch shift, time dropout, reverberation, speed change, additive noise, and audio clipping. The pretrained model that removes the pitch shift and applied only the remaining 5 augmentation techniques shows the lowest performance in most downstream tasks. When the time dropout function is removed, a relatively large performance degradation occurs in the frozen-model evaluation. It can be observed that the six audio augmentation techniques applied to this model create various augmented *views* that affect WaveBYOL's ability to learn audio representations.

Table 7 shows the results of the frozen-model evaluation performed by generating a pretrained model after removing the L2-normalization contained in the augmentation layer and encoding layer of the proposed model. The L2-normalization removed from the encoding layer is located before passing through the feature encoder.

**TABLE 6.** Ablation study results. Comparison of the effects of different component in the augmentation layer in terms of accuracy (%).

| Model | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|
| | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| Baseline (WaveBYOL) | **54.7** | **56.4** | **87.0** | **80.4** | **73.1** | **93.7** | **68.1** |
| w/o pitch shift | 47.8 | 28.6 | 80.5 | 67.2 | 57.9 | 88.7 | 61.9 |
| w/o time dropout | 50.4 | 33.0 | 82.3 | 68.0 | 58.9 | 89.1 | 62.4 |
| w/o reverberation | 51.0 | 37.0 | 82.5 | 72.8 | 66.7 | 90.9 | 62.1 |
| w/o speech change | 43.5 | 35.5 | 84.4 | 69.6 | 62.4 | 89.9 | 61.1 |
| w/o additive noise | 52.1 | 41.4 | 84.7 | 75.0 | 66.8 | 91.8 | 64.3 |
| w/o clipping audio | 50.3 | 43.9 | 85.9 | 73.8 | 67.5 | 91.3 | 68.1 |

**TABLE 7.** Ablation study results. Evaluation of the contributions of the augmentation layer (AL) and encoding layer (EL) with L2 normalization in terms of accuracy (%).

| Model | Datasets | | | | | | |
|---|---|---|---|---|---|---|---|
| | US8K [32] | VoxCeleb1 [34] | SPCV2 [36] | ESC-50 [33] | RAVDESS [37] | VoxForge [35] | NSynth [38] |
| Baseline (WaveBYOL) | **54.7** | **56.4** | **87.0** | **80.4** | **73.1** | **93.7** | **68.1** |
| w/o AL L2-normalization | 50.6 | 41.8 | 82.3 | 70.8 | 56.1 | 91.1 | 62.8 |
| w/o EL L2-normalization | 42.0 | 44.1 | 81.9 | 47.0 | 46.8 | 83.2 | 58.9 |

In the pretrained model, the architecture and parameters of the model are set the same as those in the previous experiment except for L2-normalization. In this ablation study, we can determine the contribution of the L2-normalization to representation learning in the pretext task. As shown in Table 7, even if only one L2-normalization applied to the model is removed, a decrease in performance occurs. In particular, when the L2-normalization process of the encoding layer is removed, the accuracy drops significantly for most downstream tasks. The application of L2-normalization to the encoding layer plays a role in preventing collapsed representations so that a general-purpose representation can be continuously learned in the WaveBYOL model. Since applying L2-normalization to the augmentation layer also normalizes the augmented raw waveform to create views, we believe that it helps the model learn general-purpose audio representations. Through two ablation studies, the contributions of the augmentation layer and encoding layer proposed in this paper are observed.

## V. CONCLUSION

In this paper, we proposed the WaveBYOL model, which can learn general-purpose audio representations directly from raw waveforms based on the BYOL approach. The augmentation layer in the WaveBYOL model is designed to create various views from the time domain of the audio waveform; the encoding layer is designed to learn representations by extracting features from the views, which are augmented audio segments. We assess the representations learned by WaveBYOL by conducting experiments involving five audio applications with seven audio downstream tasks under both frozen-model evaluation and fine-tuning settings. For a performance evaluation, we compared WaveBYOL with state-of-the-art models. In most downstream tasks, WaveBYOL showed competitive performance compared to that of the recently developed state-of-the-art models such as COLA, BYOL-A, SSAST, and DeLoRes. In particular, the proposed model achieved high performance improvements in speaker and language identification.

Two follow-up studies are currently in progress. First, we are conducting experiments that utilize the large-scale AudioSet [31] for pretraining. Second, we are redesigning the feature encoder structure so that each stack can focus on learning different audio frequency components by applying different sampling rates and convolution kernel sizes for each stack.

## ACKNOWLEDGMENT

The experiment in this article was conducted with GPU resources provided by the National IT Industry Promotion Agency (NIPA). The authors would like to thank the anonymous reviewers for their quality reviews.

## REFERENCES

[1] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2005, pp. 539–546, doi: 10.1109/CVPR.2005.202.

[2] X. Chen and K. He, "Exploring simple Siamese representation learning," 2020, *arXiv:2011.10566*.

[3] F. Wang and H. Liu, "Understanding the behaviour of contrastive loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2495–2504, doi: 10.1109/CVPR46437.2021.00252.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hintonm, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn. (PMLR)*, 2020, pp. 1597–1607, doi: doi.org/10.48550/arXiv.2002.05709.

[5] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 22243–22255.

[6] Y. You, I. Gitman, and B. Ginsburg, "Large batch training of convolutional networks," 2017, *arXiv:1708.03888*.

[7] Z. Wu, Y. Xiong, S. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," 2018, *arXiv:1805.01978*.

[8] 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. (CVPR)*, Jun. 2020, pp. 9726–9735, doi: 10.1109/CVPR42600.2020.00975.

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

[10] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. 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. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 1–14.

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

[12] 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, doi: 10.1109/ICASSP39728.2021.9413528.

[13] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio spectrogram transformer," 2021, *arXiv:2104.01778*.

[14] Y. Gong, C. Lai, Y. Chung, and J. Glass, "SSAST: Self-supervised audio spectrogram transformer," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 1–11, doi: 10.1609/aaai.v36i10.21315.

[15] S. Ghosh, A. Seth, a. Deepak Mittal, M. Singh, and S. Umesh, "DeLoRes: Decorrelating latent spaces for low-resource audio representation learning," 2022, *arXiv:2203.13628*.

[16] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," in *Proc. Int. Conf. Mach. Learn. (PMLR)*, Sep. 2022, pp. 12310–12320.

[17] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "BYOL for audio: Exploring pre-trained general-purpose audio representations," 2022, *arXiv:2204.07402*.

[18] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "BYOL for audio: Self-supervised learning for general-purpose audio representation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–8, doi: 10.1109/IJCNN52387.2021.9534474.

[19] X. Li and X. Li, "ATST: Audio representation learning with teacher-student transformer," 2022, *arXiv:2204.12076*.

[20] G. Sharma, K. Umapathy, and S. Krishnan, "Trends in audio signal feature extraction methods," *Appl. Acoust.*, vol. 158, Jan. 2020, Art. no. 107020.

[21] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, Sep. 2019, pp. 2613–2617.

[22] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "Wav2vec: Unsupervised pre-training for speech recognition," in *Proc. Interspeech*, 2019, pp. 3465–3469. [Online]. Available: https://arxiv.org/abs/1904.05862

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

[24] 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, doi: 10.1109/ICCV.2015.167.

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

[26] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 69–84.

[27] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 649–666.

[28] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2536–2544.

[29] A. N. Carr, Q. Berthet, M. Blondel, O. Teboul, and N. Zeghidour, "Self-supervised learning of audio representations from permutations with differentiable ranking," *IEEE Signal Process. Lett.*, vol. 28, pp. 708–712, 2021, doi: 10.1109/LSP.2021.3067635.

[30] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An open dataset of human-labeled sound events," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 30, pp. 829–852, 2022, doi: 10.1109/TASLP.2021.3133208.

[31] 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, doi: 10.1109/ICASSP.2017.7952261.

[32] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. 22nd ACM Int. Conf. Multimedia*, Nov. 2014, pp. 1041–1044.

[33] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proc. 23rd ACM Int. Conf. Multimedia*, Oct. 2015, pp. 1015–1018.

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

[35] K. MacLean. (2018). *Voxforge*. [Online]. Available: http://www.voxforge.org/home

[36] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018, *arXiv:1804.03209*.

[37] S. R. Livingstone and F. A. Russo, "The ryerson audio-visual database of emotional speech and song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in north American English," *PLoS ONE*, vol. 13, no. 5, May 2018, Art. no. e0196391.

[38] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1068–1077.

[39] E. Kharitonov, M. Riviere, G. Synnaeve, L. Wolf, P.-E. Mazare, 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, doi: 10.1109/SLT48900.2021.9383605.

[40] B. Heo, S. Chun, S. Joon Oh, D. Han, S. Yun, G. Kim, Y. Uh, and J.-W. Ha, "AdamP: Slowing down the slowdown for momentum optimizers on scale-invariant weights," 2020, *arXiv:2006.08217*.

[41] *WaveBYOL Implementation and Pretrained Models*. Accessed: 2022. [Online]. Available: https://github.com/waverDeep/WaveBYOL

**SUNGHYUN KIM** received the A.S. degree in computer and mobile convergence engineering from the Gyeonggi University of Science and Technology, Gyeonggi-do, South Korea, in 2020, the B.S. degree in computer engineering from The Korean Academic Credit Bank System, Seoul, South Korea, in 2020, and the M.S. degree in robotics from Kwangwoon University, Seoul, in 2022. Since 2022, he has been working with NEOWIZ, Gyeonggi-do. His research interests include communication networks, software development, positioning technology, machine learning, self-supervised learning, and speech recognition and synthesis.

**YONG-HOON CHOI** (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronic engineering from Yonsei University, Seoul, South Korea, in 1995, 1997, and 2001, respectively. From 2001 to 2002, he was a Research Associate at the Institute for Systems Research (ISR), University of Maryland, College Park, MD, USA. From 2002 to 2005, he was a Chief Research Engineer at LG Electronics. Since 2005, he has been a Professor with the College of Electronics and Information Engineering, Kwangwoon University, Seoul. His research interests include communication networks, machine learning, and speech/sound recognition and synthesis. He received the 28th Choon-Gang Award from the Choon-Gang Memorial Association, in 2013. He has served in numerous international conferences as the Organizing Committee Chair, such as IEEE WF-IoT, ICOIN, ICTC, and ICUFN, and the Technical Program Chair for international conferences, including ICOIN 2015, ICOIN 2016, ICOIN 2017, ICOIN 2021, and ICOIN 2022.