ACGAN-based Data Augmentation Integrated with Long-term Scalogram for Acoustic Scene Classification

Hangting Chen, Zuozen Liu, Zongming Liu, and Pengyuan Zhang, Member, IEEE

Abstract—In acoustic scene classification (ASC), acoustic features play a crucial role in the extraction of scene information, which can be stored over different time scales. Moreover, the limited size of the dataset may lead to a biased model with a poor performance for records from unseen cities and confusing scene classes. In order to overcome this, we propose a long-term wavelet feature that requires a lower storage capacity and can be classified faster and more accurately compared with classic Mel filter bank coefficients (FBank). This feature can be extracted with predefined wavelet scales similar to the FBank. Furthermore, a novel data augmentation scheme based on generative adversarial neural networks with auxiliary classifiers (ACGANs) is adopted to improve the generalization of the ASC systems. The scheme, which contains ACGANs and a sample filter, extends the database iteratively by splitting the dataset, training the ACGANs and subsequently filtering samples. Experiments were conducted on datasets from the Detection and Classification of Acoustic Scenes and Events (DCASE) challenges. The results on the DCASE19 datasets demonstrate the improved performance of the proposed techniques compared with the classic FBank classifier. Moreover, the proposed fusion system achieved first place in the DCASE19 competition and surpassed the top accuracies on the DCASE17 dataset.

Index Terms—DCASE, acoustic scene classification, scalogram, generative adversarial network.

I. INTRODUCTION

E

vironmental sound carries a large amount of information related to the surroundings. Humans are able to understand the general context of the acoustic scene by listening to sounds and can automatically adapt the auditory systems. However, enabling devices to sense the environment is a challenging task of machine listening and computational auditory scene analysis [1]. Acoustic scene classification (ASC) is one of the key topics required to improve the understanding of sound. It aims to classify sound into one of several predefined classes, e.g., park, office, library [2] [3]. Obtaining key information of the environment has huge potential in audio-context-based searching [4] and the manufacturing of context-aware devices [5]. Detection and Classification of Acoustic Scenes and Events (DCASE) challenges are organized annually by the IEEE Audio and Acoustic Signal Processing (AASP) Technical Committee. Furthermore, the DCASE challenges have released some of the largest datasets in the field.

Real-world acoustic scene information is generally stored in background sounds, with variations in the time scale for cues. For example, the pitch and timbre are observed at a scale of milliseconds, while rhythm is of seconds [6]. Figure 1 plots the short-time Fourier transform (STFT) spectra of audios recorded at an airport and a metro. In particular, the sound from the airport contains short-term footsteps located sparsely in the frequency axis. The sound from the metro exhibits the rhythm of the carriage advancing, which covers a time scale of hundreds of milliseconds and fills all the high and a part of the low frequencies. However, most of the conventional features, such as the Mel-scale filter bank coefficients (FBank) [7], are short-term and designed for tasks such as speech recognition. More specifically, they describe signals with many fluctuations, suitable for short-term phoneme recognition [8] but inappropriate for depicting acoustic scenes. We assume that for ASC tasks, the scene cues are distributed along time at multiple scales, thus long-term features can act as a more accurate representation of background information. Long-term features still have the ability to describe short-term information by averaging over a long-term window, as long as the short-term cues are relatively distributed in a consistent manner within the spectra. Wavelet transform is able to model the signal at multiple resolutions [9]. In our previous work, we demonstrated that by adopting a long-term window for the wavelet filters, the convolutional wavelet transform was able to outperform the FBank features [10]. The wavelet filters were applied directly to the STFT spectra, achieving a more accurate prediction compared to short-term features [11]. In this paper, we uncover the design of the wavelet filters and compare different settings to give a detailed analysis under various classifiers.

Accurately defining an acoustic scene is a complicated task for real-life audios. Similar to the majority of the related research, the scene recognition task in the current paper is simplified as a classification problem with predefined classes [2]. The extracted acoustic features are modeled by statistical learning methods. Since the DCASE16 competition, a large amount of collected data has supported the training of neural networks. According to the DCASE16 competition report, deep learning methods, in particular convolutional neural...
Another concern is the generalization ability of models. Concretely, the classifier usually can be fit well on the training set but fails to classify testing records, particularly those from unseen cities \([13]\). The balanced training data still results in biased models, yielding ambiguous results on some confusing scenes. Therefore, data augmentation approaches are useful to extend datasets and to improve the performance of classifiers. Classic methods (e.g., speed and volume perturbation \([13]\), pitch shift \([14]\)) introduce diversity increase in training dataset but cannot generate samples with new content. Generative adversarial networks (GANs), which can generate samples via a nonlinear transform, can be adopted in order to overcome the limitations of classic methods. However, controlling the number of synthetic samples to ensure performance improvements can be difficult. In the current paper, a GAN-based data augmentation scheme is proposed to iteratively extend the dataset, ensuring the improvement of the final systems performance.

The contribution of this work is 4-fold. First, a wavelet scale is designed to extract long-term features, which outperforms the classic Mel filters. Second, a group of neural networks, including 1D and 2D convolutional architectures for scene classifiers and GANs, are described and compared under long-term and short-term features. In particular, the adversarial city training and the discrete cosine transform (DCT)-based temporal module are proposed to increase the diversity of the models. Third, the auxiliary classifier GANs (ACGANs) are employed to directly generate raw feature maps with input conditions, instead of generating bottleneck features with individual GANs as in \([15]\). Fourth, a novel data augmentation scheme is proposed to improve the performance by integrating ACGANs and neural network-based sample filters. This scheme demonstrates the potential of deep learning with the performance improvements achieved by utilizing iterative procedures of city-based dataset splitting, ACGAN training and sample filtering. Our system achieved the highest accuracies for DCASE19 Task 1A and delivered a state-of-the-art performance for DCASE17 Task 1.

The remainder of the paper is organized as follows. Section II reviews the previous related work. Section III introduces the proposed passband wavelet filters and the algorithm employed to extract features. Section IV details the design of the classifiers. Section V describes the ACGAN-based data augmentation scheme along with the detailed architectures. Section VI describes the dataset and the configuration of our experiments. Section VII presents and discusses the results. Finally, Section VIII concludes this work and outlines future research directions.

## II. RELATED WORK

The early acoustic scene classification \([16]\) \([17]\) \([18]\) and DCASE submission systems \([1]\) \([12]\) \([19]\) \([20]\) typically consist of 2 steps: feature extraction and classification. The first ASC system in record was built using the nearest neighbor classifier and a recurrent neural network (RNN) with FBank and other perceptual features \([16]\). In recent researches, a classic architecture is composed of FBank as the feature and a CNN as the back-end classifier \([21]\) \([22]\) \([23]\) \([24]\).

Several audio features have been exploited in ASC systems, including low level spectrum features (e.g., spectrograms \([25]\)), auditory filter banks (e.g., FBank \([22]\)), cepstral features (e.g., MFCC \([14]\) and others (e.g., linear prediction coefficients \([1]\) ). Several studies have proposed feature transform, such as non-negative matrix factorization \([26]\) and independent component analysis \([27]\), to derive new quantities. The constant-Q analysis is a spectral representation that fits the exponentially spaced filters \([28]\). Acoustic features based on constant-Q transform (CQT) are widely used in the field of music analysis \([29]\) \([30]\). The deep scattering spectrum proposed in \([5]\) involves a set of evenly and geometrically-spaced wavelet filters and scattering networks, which output a group of features with multi-frequency resolutions. The CQT-based features, explored in DCASE16 \([31]\) \([32]\) \([33]\) \([34]\) \([35]\), were generally extracted using a short-term window within a range of 10 to 50ms \([25]\) \([31]\) \([34]\) \([36]\) and serve as a complementary feature \([25]\) \([34]\).

Much early research adopted generative models as classifiers, such as mixture Gaussian models \([37]\), and hidden Markov models \([38]\). Constrained by limited dataset sizes, methods such as support vector machine (SVM) \([39]\) \([40]\) and non-negative matrix factorization \([26]\) were also employed. Some studies attempted to infer the acoustic scene from event cues \([41]\). Since the DCASE16 challenge, the majority of the successful classifiers in the competitions have adopted deep neural networks, including multi-layer perceptron (MLP) \([42]\), time-delay neural networks (TDNNs) \([43]\), CNNs \([14]\) \([23]\), RNNs \([44]\) and the hybrid models \([15]\). Some researchers built the networks involving the structures of ResNet \([45]\), self-attention \([46]\) and Inception \([47]\). CNNs were particularly popular in the competition, with all top 10 teams adopting CNNs as classifiers for the DCASE18 \([48]\) and DCASE19 competition \([49]\). CNNs have been demonstrated to outperform other methods in the ASC task \([50]\) \([10]\). This can be attributed to the little temporary dynamics but much local correlation existing in the acoustic scene spectra \([50]\). Two branches are employed for CNNs: 1D convolutional kernels along the time axis \([10]\) and 2D kernels along the time-scale axis \([21]\) \([23]\). However, limited studies have investigated the difference between these two branches.
Inspired by the game theory, GAN is a powerful generative model with the generator and discriminator competing with each other, first introduced by [51]. The generator learns from the training set and attempts to generate fake samples to fool the discriminator. Meanwhile, the discriminator attempts to classify whether the input is real or fake. Finally, a balance is reached when the fake samples from the generative model match the true distribution and the discriminator cannot distinguish between them. The GAN architecture has been developed to generate samples with labels, such as conditional GANs (CGANs) [52], ACGANs [53] and autoencoder based GANs [54]. Compared with the vanilla GAN, CGANs utilize an additional label input to generate samples from specific categories. In ACGANs, the discriminator provides the probability over the sources and the class labels. By optimizing both real/fake and label classification cost functions, the ACGANs can generate higher quality samples compared with the standard GAN [53]. GAN and its variants have archived impressive results over a wide range of applications, including image generation [55], style transfer [56] and image super-resolution [57]. Data augmentation using the GAN architecture has been demonstrated to improve classification performances [58] [59]. In ASC tasks, the classic data augmentation techniques, such as Mixup [35] and SpecAug [60], have been explored. Furthermore, standard GANs with SVM have suggested improved performance with the generated bottleneck features [15].

III. Passband Wavelet Filters

Different acoustic scenes exhibit background sounds with various time scales. A densely sampled raw signal contains considerable meaningless fluctuations. Thus a representation of the scene information without short-term fluctuations can benefit ASC tasks. In this section, we first introduce the scalogram extracted by a set of passband wavelet filters. Then we compare FBank with the scalogram in order to investigate their differences.

A. Filter definitions

A scalogram is extracted with a set of passband filters. We set $\lambda_0$ as the center frequency of the mother wavelet, $Q$ as the number of filters in each octave. A dilated wavelet, indexed by $k \in \mathbb{Z}$, can be determined by its center frequency $\lambda_k$ and bandwidth $\delta_k$,

$$\lambda_k = 2^{k/Q} \lambda_0,$$

$$\delta_k = \lambda_k/Q,$$  (2)

which follows the constant-Q property. In addition, its time interval can be described as,

$$T_k = \frac{1}{\delta_k}.$$  (3)

A Gaussian-like passband filter $\hat{\phi}_k(f)$ defined in the frequency domain can be expressed as follows,

$$\hat{\phi}_k(f) = \exp\left(-\frac{(\lambda_k - f)^2}{2(\delta_k)^2}\right).$$  (4)

where $f$ is the frequency. This filter can be transformed into the time domain using the following,

$$\phi_k(t) = \int_0^1 \hat{\phi}_k(f)e^{jft} df.$$  (5)

A specific set of wavelets are determined by the upper-bound $f_h$ and the maximum length of time window $T_{max}$, which usually corresponds to the double of the maximum time interval of the wavelets. In general, the center frequency of the mother wavelet is set as $\lambda_0 = f_h$ and a set of wavelets are defined by the index $k = 0, ..., -K + 1$, where $K$ is the total number of constant-Q-spaced wavelets. The maximum time interval is defined as,

$$T_K = \frac{Q}{2^{\frac{K}{Q}}f_h}.$$  (6)

By setting $T_K = T_{max}/2$, $K$ can be expressed as

$$K = 1 + Q\log_2\frac{T_{max} f_h}{2Q}.$$  (7)

In our practice, the direct use of constant-Q wavelets to extract the scalogram results in sub-optimal performances. The length of time window $T_{max}$ limits the maximum time interval, and also determines the lowest center frequency. To cover the low frequency parts, a set of evenly-spaced passband filters are defined on the frequency domain below $2^{(-K+1)/Q}f_h$, also denoted as the wavelet for simplicity here. Their center frequency intervals are determined by the last two constant-Q wavelets. With the lower bound frequency $f_i \approx 0$, the total number of low frequency wavelets is

$$P = \frac{1}{2^{1/Q} - 1}.$$  (8)

The center frequency and bandwidth of evenly-spaced passband filters, indexed by $k = -K, ..., -K + P + 1$, is

$$\lambda_k = \lambda_K - \frac{\lambda_K - f_i}{P}(K - k),$$  (9)

$$\delta_k = \lambda_K/Q.$$  (10)

In summary, given $f_h$, $f_i$ and $T_{max}$, a set of constant-Q filters are determined by Equation (7), (2), (1), (4), and a set of evenly-spaced passband filters are determined by Equation (8), (9), (10). In this paper, the feature extracted by this group of filters is denoted as the scalogram.

We reformulate the equations with index $j = 0, ..., J - 1$ ($J = K + P$) as follows,

$$\lambda_j = \begin{cases} f_i + \frac{2Q/T_{max} - f_i}{P}j, & j = 0, ..., P - 1 \\ \frac{2Q}{T_{max}}2^{j/Q}, & j = P, ..., J - 1 \end{cases},$$  (11)

$$\delta_j = \begin{cases} 2/T_{max}, & j = 0, ..., P - 1 \\ \lambda_j/Q, & j = P, ..., J - 1 \end{cases},$$  (12)

where the center frequency increases following index $j$.

The scalogram can be calculated by the convolution between the raw signal and the filters, which can be sped up using overlap-add method,

$$y(t, k) = \sum_{m} \phi_k(t - m)x(m)dm,$$  (13)
where \( y \) is the convolution result at every sample, which contains lots of fluctuations and consumes a substantial amount of computation and storage. Here the wavelet filters are directly applied to the STFT spectra following the FBank calculation. The scalogram is computed via the following steps,

1) Given frame length \( T_{\text{win}} \) and shift \( T_{\text{shift}} \), the signal is framed and the magnitude of the STFT spectrum is calculated.
2) Given \( f_h, f_l \) and \( T_{\text{max}} \), \( J \) and \( P \) are calculated. Following Equation (11), (12), (14), a group of wavelet filters are defined and their digit forms are calculated.
3) The filters are applied to the magnitude spectrum. The sum of the power of the magnitude for each filter is calculated.
4) The logarithm of the power coefficients is taken.

The setup calculates the convolution of the signal and the filter in a similar way to Equation (13), but the summation of energy downsamples the point-wise feature into a frame-wise one. The scalogram has a dimension of \( L \times N_{\text{wavelet}} \), where \( L \) is the number of frames and \( N_{\text{wavelet}} = J = K + P \) is the total number of filters.

B. Comparison between FBank and scalogram

The proposed wavelet scale is distinct to the Mel scale. The widely known Mel scale is expressed as

\[
m(f) = 781 \log_2(1 + f/700),
\]

where \( m \) denotes the Mel frequency. The maximum and minimal Mel frequencies are generally defined by \( m_l = m(f_l) \) and \( m_h = m(f_h) \). Here, the number of Mel filters is set to \( N_{\text{Mel}} \), thus the Mel frequency is evenly separated into \( N_{\text{Mel}} + 1 \) parts. We use \( m_i \) to represent the center frequencies of the \( i \)th filter, where \( i = 0, ..., N_{\text{Mel}} - 1 \). \( m_i \) is subsequently calculated as

\[
m_i = 700 \times 2^{m_i/781} - 700.
\]

The Mel scale is actually geometrically spaced with a negative bias.

Compared with Equation (11) and (12), the low frequency patterns of the scalogram and Mel scale are distinct. At the high frequencies, not considering the negative bias in Equation (15), the base center frequency is \( 700 \text{Hz} \) with \( 2^{m_i/781} \approx 0 \) and \( N_{\text{Mel}} \gg \log_2(1 + f_h/700) \), yet for the proposed wavelet filters, this value is calculated based on the maximum window length. We visualize these two scales in Figure 2 with \( f_h = 24kHz \), \( f_l = 0.5Hz \), \( T_{\text{max}} = 341ms \), \( Q = 35 \) and \( N_{\text{Mel}} = N_{\text{wavelet}} = 290 \). Under these settings, the wavelet filters focus more on the low frequency part and take larger steps at high frequencies.

IV. CONVOLUTIONAL CLASSIFIERS

In general, CNNs achieve a better performance compared to the other architectures described in Section II. As plotted in Figure 3, we introduce two different neural networks: a fully convolutional neural network (FCNN) and a deep convolutional neural network (DCNN) with 2D and 1D convolution kernels for short-term and long-term features, respectively.

A. FCNN

The FCNN classifier. The input feature map is of size frames \((L \times \text{channels}(c_s) \times \text{filters}(n)) \). The notation “5 \times 5 \text{Conv}(pad=2, stride=2)\times c_{s} - \text{BN}-\text{ReLU}” denotes a convolutional kernel with \( c_s \), output channels and a size of \( 5 \times 5 \), followed by batch normalization and ReLU activation.

| Layer | Settings |
|-------|----------|
| Input | Fbank \( L \times c_s \times n \) |
| Conv1 | 5 \times 5 \text{Conv}(pad=2, stride=2)\times 14c_s - \text{BN}-\text{ReLU} |
| Conv2 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 14c_s - \text{BN}-\text{ReLU} |
| Conv3 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 28c_s - \text{BN}-\text{ReLU} |
| Conv4 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 28c_s - \text{BN}-\text{ReLU} |
| Conv5 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 56c_s - \text{BN}-\text{ReLU} |
| Conv6 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 56c_s - \text{BN}-\text{ReLU} |
| Conv7 | 3 \times 3 \text{Conv}(pad=1, stride=1)\times 56c_s - \text{BN}-\text{ReLU} |
| Conv8 | 3 \times 3 \text{Conv}(pad=0, stride=1)\times 128c_s - \text{BN}-\text{ReLU} |
| Conv9 | 3 \times 3 \text{Conv}(pad=0, stride=1)\times 128c_s - \text{BN}-\text{ReLU} |
| Conv10 | 3 \times 3 \text{Conv}(pad=0, stride=1)\times 128c_s - \text{BN}-\text{ReLU} |
| Pooling1 | 1 \times 1 \text{Conv}(pad=0, stride=1)\times c_{s}^{\text{scene}} - \text{BN}-\text{ReLU} |
| Output | \( c_{s}^{\text{scene}} \)-way SoftMax |

The short-term features yield a strong local correlation along both the time and scale axis. To model this characteristic, a FCNN architecture similar to the VGG-style network [61] proposed in [43] is adopted here (Table I). The network consists of repeatedly stacked layers, each including a convolutional operation with small-sized 2D kernels, batch normalization and ReLU activation. The maximum pooling and dropout layers are interleaved between some convolutional layers. A \( 1 \times 1 \) kernel in the last convolutional layer reduces the channel to \( c_{s}^{\text{scene}} \), with the feature map averaged over each channel. The whole network is fully convolutional, which reduces the number of parameters and makes the learned transform both time and scale invariant. During the test, the scene is chosen corresponding to the maximum log probability.
B. DCNN

The long-term features exhibit a weak correlation along time. Thus, the DCNN serves as the back-end classifier (Table II), which employs small 1D convolutional kernels to learn high-level scale patterns. The network is first stacked by 4 repeated layers, each including a 1D convolutional operator along the scale axis, batch normalization, ReLU activation and average pooling. The channel number of convolutional kernels doubles after each layer. Note that before being fed into fully connected layers, the output of each convolutional layer, as well as the input feature, is concatenated into one super vector for each frame. The convolutional kernel generates high-level patterns to increase data discrimination. Both the input feature map and the output of the convolutional kernels benefit the system performance. During the test, the scene is chosen corresponding to the maximum sum of the frame-wise log probability.

Two additional techniques employed for DCNNs, adversarial city training and a temporal module based on the DCT, are introduced in order to improve the performance.

1) Adversarial city training: The scene classifier cannot generalize well on records from unseen cities. Thus, an adversary training branch, composed of a gradient reverse layer (GRL) [62] and a 2-layer fully-connected classifier, is employed following the “Concat” layer in Table II to perform adversarial city training. This branch classifies the audios into their recorded cities with cross entropy city loss \( L(\text{clf}_{\text{city}}(x_i), y_i^{\text{city}}) \) where \( \text{clf}_{\text{city}}(x_i), x_i, y_i^{\text{city}} \) represent the output of the city branch, the feature and the city label of sample \( i \), respectively. The overall loss \( L_{\text{adv}} \) can be written as

\[
L_{\text{adv}} = L(\text{clf}_{\text{scene}}(x_i), y_i^{\text{scene}}) - \gamma_{\text{adv}} L(\text{clf}_{\text{city}}(x_i), y_i^{\text{city}}),
\]

where \( \text{clf}_{\text{scene}}(x_i), y_i^{\text{scene}} \) represent the output of the main scene classifier and the scene label of sample \( i \), respectively. \( \gamma_{\text{adv}} \) balances the scene and the city loss. During training, the minimax game is conducted by the GRL, which introduces the negative part of \( L_{\text{adv}} \) by gradient reverse. In detail, the parameters of the main scene classifier including its shallow convolutional kernels are optimized in order to minimize \( L_{\text{adv}} \) (i.e., to minimize the scene loss \( L(\text{clf}_{\text{scene}}(x_i), y_i^{\text{scene}}) \)) and to maximize the city loss \( L(\text{clf}_{\text{city}}(x_i), y_i^{\text{city}}) \). The parameters of the adversary training branch are optimized to minimize \( L(\text{clf}_{\text{scene}}(x_i), y_i^{\text{scene}}) \) and \( L(\text{clf}_{\text{city}}(x_i), y_i^{\text{city}}) \). This minimax competition will first increase the discrimination of city branch and the city-invariance of the features generated by convolutional kernels from the scene classifier. Eventually it will occur to the point where the output of the convolutional kernels is similar for the same scene over different city domains.

2) DCT-based temporal module: The DCT-based temporal module utilizes information across time with learnable weight matrices (Figure 4). The input feature map is first split into non-overlapping chunks \( X \in \mathbb{R}^{T \times N} \) along with the second order statistical feature \( Y \in \mathbb{R}^{T \times N} \), where \( T \) is the frame number in a chunk and \( N \) is the feature dimension.
The chunks $X$ and $Y$ are transformed into the DCT domain and weighted by the learnable weights $W_X$ and $W_Y$. After the inverse DCT, the outputs $\tilde{X}$ and $\tilde{Y}$ are concatenated along the feature dimensions. They are subsequently linearly transformed into a feature with the same size of the original input.

![Feature Map]

Fig. 4. The DCT-based temporal module.

This module is inspired by noise reduction approaches in image processing. Here, the weights can be viewed as attention-like coefficients, which strengthen and weaken the target feature map bins. Moreover, the module can be inserted without changing the original network topology as its input and output share the same size.

V. DATA AUGMENTATION SCHEME

Although ASC systems are able to accurately classify most training samples, they may suffer from inferring some test records, particularly those lying on the margin of classification planes, collected in unseen cities and belonging to confusing scenes. To improve the generalization performance, we apply a GAN-based data augmentation scheme. Distinct from [13], which built individual GANs for each acoustic scene and generated bottleneck features, our proposed technique performs conditional acoustic feature synthesis with ACGANs to directly generate different spectrum samples from one unified model. In addition, a novel iterative scheme combining ACGANs and a deep-learning-based sample filter is deployed to yield stable performance improvements.

A. ACGAN

The GAN plays a minimax game where the discriminator is optimized to discriminate real/fake samples and the generator attempts to fool the discriminator. More specifically, for ACGAN with the condition, a generator learns to generate acoustic features with randomly sampled noise and a one-hot condition as input. The main branch of the discriminator is trained to discriminate whether the input sample is real or fake. On the other hand, an additional branch classifies the input into scenes. In formulation, the discriminator network $D$ with parameters $\theta_D$ predicts the posterior source probability of the real samples $s_i$ and the fake ones $\tilde{s}_i$, which are generated by the generator $G$ with parameter $\theta_G$, 

$$P(\text{Real}|x_i; \theta_D) = D_{source}(x_i),$$  \hspace{1cm} (18)

where $x_i$ is $s_i$ or $\tilde{s}_i$. The $G/D$ is optimized to maximize/minimize the following real/fake loss,

$$L_{source} = -\sum_i \{\log P(\text{Real}|s_i; \theta_D) + \log P(\text{Fake}|\tilde{s}_i; \theta_D)\},$$

$$= -\sum_i \{\log D_{source}(s_i) + \log [1 - D_{source}(G(z, y_i))]\},$$  \hspace{1cm} (19)

where $z$ and $y_i$ are the noise and the scene condition, respectively.

To make the generated samples not only look real but belong to the target scene, we perform an additional branch located at $D$ for the scene classification. The auxiliary classification loss is formulated as

$$L_{scene} = -\sum_i \sum_{a \in A} \mathbb{I}_{a=y_i} \{\log (D_{scene}(x_i)) + \log (D_{scene}(G(z, y_i)))\},$$  \hspace{1cm} (20)

where $A$ is a collection of scene classes, $\mathbb{I}$ is the indicator function, $y_i$ is the scene label of $x_i$. $G$ and $D$ are separately trained to optimize the multi-task loss as follows,

$$\hat{\theta}_D = \arg \min_{\theta_D} L_{source} + \gamma_{aux} L_{scene},$$

$$\hat{\theta}_G = \arg \min_{\theta_G} -L_{source} + \gamma_{aux} L_{scene},$$  \hspace{1cm} (21)

where $\gamma_{aux}$ controls the ratio between the source and scene classification loss.

B. Augmentation scheme

The generated fake samples are included into the dataset with verification processes to ensure performance improvements. An iterative procedure is conducted to test whether the generated samples are indeed useful for the performance. As shown in Figure 5 at the $k$th iteration, the whole training database is first split into training and testing subset. A scene classifier $clf^A$ is trained using the training subset with early-stopping controlled by the testing subset. Meanwhile, an ACGAN is trained on the training subset. To guarantee that the generated samples benefit both the current and final classifiers, a simple sample filter is deployed by piping the fake samples through classifier $clf^A$, and keeping only those lying on the classification margin. A new scene classifier $clf^B$ is further trained on the extended dataset, which contains current fake samples filtered by the sample filter and the original subset. The $clf^A$ and $clf^B$'s performance test is conducted on the testing subset. If $clf^B$ achieves a higher classification accuracy, these samples are accepted and merged into the whole database, otherwise they are abandoned. A final classifier is built on the whole database, including both real and accumulated fake samples. We address the dataset splitting and the sample filter as follows.

We apply 3 methods to split the dataset in the experiments. The simplest one is the fixed splitting, which splits the whole database using a fixed random seed and exchanges the training and testing sets during the iterations. The second method employs different random seeds to split those datasets at each
testing subset

N different generator architectures. To sample a scene, we can generate frame-wise and segment-wise samples with information.

iteration. The third randomly splits samples according to city information.

Depending on the characteristics of the input features, we can generate frame-wise and segment-wise samples with different generator architectures. To sample a scene, $N_{\text{sample}}$ and $C_{\text{sample}}$ are set as the maximum number of samples and sample time, respectively. More specifically, the sampling procedure stops if either of these two conditions is met. A sample filter determines whether the sample lies on the classification borderline. For the $C$-class classification, the borderline is set as the output probability equal to $1/C$ with margin $m_p$. The algorithm used to filter frame-wisely generated samples is described in Algorithm 1. The segment-wisely generated samples are processed similarly, except that the operation is performed on the segment level.

C. Network architecture

Two sets of ACGAN architectures are designed to generate segment-wise and frame-wise samples, corresponding to short-term and long-term features respectively (Table III and IV). The discriminators are simplified versions of the aforementioned DCNNs and FCNNs. The generators are mirror images of the discriminators. To generate long-term features, $1D$ convolutional kernels are applied throughout, which means that the temporal dynamics is not considered. To avoid the training collapse of generators, the leaky ReLU replaces the original sparse ReLU [63]. Meanwhile, the number of convolution layers is reduced to half. The randomness of the generator is from the noise multiplying the embeddings indicated by the one-hot scene condition.

VI. EXPERIMENTAL CONFIGURATION

A. Dataset

The experiments were conducted on the DCASE19 Task 1A dataset, which includes both the development and evaluation set. As the DCASE committee does not intend to release the ground truth of the evaluation set, we directly evaluated the systems on the officially provided fold-1 setup. In terms of the published competition results, the accuracies of the fold-1 evaluation procedure are sufficient for comparisons. Approximately 13% of the fold-1 training set was randomly selected and reserved as the validation set for hyperparameter fine-tuning and early stopping. The systems were evaluated using the fold-1 evaluation set. Key experiments were also performed on the DCASE17 Task 1 dataset for fair comparisons with the SVM-GAN-based data augmentation framework [15]. Overall accuracy is used as the evaluation metric, calculated as the percentage of correctly classified segments out of the total number. This measure is slightly different from the average of the class-wise accuracy when the number of samples is imbalanced among different classes. Experimental results demonstrate that the overall accuracy is slightly lower (by approx. 0.10%) than the average class-wise accuracy on the DCASE19 fold-1 dataset.

Algorithm 1 The sample filter for frame-wisely generated features

clf$^A$: Scene classifier trained on the training subset 
{$G$}: A set of generators from different training epochs
$N_{\text{sample}}$: maximum number of samples for each scene
$T_{\text{sample}}$: maximum number of sample times
$C$: The number of scene classes

The feature is of size $L \times c_s \times n$

1: Maximum number of frames sampled from each $G$

$N_{\text{frame}} = L \times N_{\text{sample}}/|\{G\}|

2: An empty candidate database $\Lambda$

3: for each scene label $z$, each $G$ in $\{G\}$ do

4: Current state $n_{\text{sample}} := 0$, $t_{\text{sample}} := 0$

5: while $t_{\text{sample}} < T_{\text{sample}}$ and $n_{\text{frame}} < N_{\text{frame}}$ do

6: Generate $N_{\text{frame}}$ frames from $G$, each frame $f \in \mathbb{R}^{c_s \times n}$

7: for each $f$ in $N_{\text{frame}}$ frames do

8: if $1/C - m_p < \text{clf}^A(f)[z] < 1/C + m_p$ then

9: $\Lambda := \Lambda \cup f$

10: $n_{\text{frame}} := n_{\text{frame}} + 1$

11: end if

12: end for

13: $t_{\text{sample}} := t_{\text{sample}} + 1$

14: end while

15: end for

16: The number of generated samples $N_{\text{gen}} = \lceil \frac{|\Lambda|}{L} \rceil$

17: Reshape $\Lambda$ into a size of $N_{\text{gen}} \times L \times c_s \times n$

18: return $\Lambda$

fig5.png: The scheme of data augmentation to iteratively extend the database.
The DCASE19 audios were recorded with microphones that mimic headphone recordings, with recordings sounding similar to those listened to in ears [20] [22]. The dataset contains 10 acoustic scene classes. For the fold-1 setup, 5 European cities are included in both the training and evaluation sets. In addition, the evaluation set contains a city that is unseen for the trained systems. A detailed description of the datasets is reported in Table V. The official baseline achieved an accuracy of 62.5% on the fold-1 evaluation set [20].

The DCASE17 dataset contains 15 scene classes. The official baseline achieved 61.0% on the evaluation set [3].

### B. Features

Table VI reports the detailed settings of the scalogram and FBank. The features were extracted on the STFT spectra with predefined filters. Global normalization was performed according to the mean and standard derivation calculated on the training set. The features were extracted separately from the dual channels and concatenated along the channel axis to form a 3D feature map. The left-right feature was named after being from the left and right audio channels. Furthermore, the ave-diff feature was extracted from the mean and differential signals.

### Table III
The segment-wise ACGAN with segment-wise input and output.

| Layer       | Settings                                                                 |
|-------------|--------------------------------------------------------------------------|
| Input       | One-hot scene Label $c_{t_{wise}}^m$ & Noise $h_{end}\_m$              |
| Embedding   | $\text{Linear}(h\text{-units})\text{-BN-LeakyReLU}$                     |
| FC1         | $\text{Linear}(c_{t_{wise}}\_\text{units})\text{-BN-LeakyReLU}$         |
| Reshape     | Reshape to bottleneck feature $32c_{G}\times L_{bs} \times n_{bs}$      |
| DeConv1     | $5 \times 5$ $\text{DeConv}(pad=2, stride=4)\times 8c_{G}\text{-BN-LeakyReLU}$ |
| DeConv2     | $5 \times 5$ $\text{DeConv}(pad=2, stride=4)\times c_{G}\text{-BN-LeakyReLU}$ |
| FC3         | $\text{Linear}(L \text{ units along time axis})$                       |
| FC4         | $\text{Linear}(n \text{ units})$                                       |
| Output      | Permute the dimensions to size $L \times c_{G} \times n$                |
| Input       | $\text{Feature}\ L \times c_{G} \times n$                              |
| Conv1       | $5 \times 5$ $\text{Conv}(pad=2, stride=1)\times 8c_{G}\text{-BN-LeakyReLU}$ |
|             | $\text{Pooling}(stride=4)\text{-Dropout2D}$                            |
| Conv2       | $5 \times 5$ $\text{Conv}(pad=2, stride=1)\times 32c_{G}\text{-BN-LeakyReLU}$ |
|             | $\text{Pooling}(stride=4)\text{-Dropout2D}$                            |
| Reshape     | Reshape each segment into a vector                                       |
| FC1         | $\text{Linear}(h\text{-units})\text{-BN-LeakyReLU}$                     |
| Output      | $\text{Linear}(c_{t_{wise}}\text{-units})\text{-SoftMax} \& \text{Linear}(1\text{-unit})\text{-Sigmoid}$ |

### Table V
Informations of the fold-1 setup on the DCASE19 Task 1A dataset

| Attribute           | Development set | Evaluation set |
|---------------------|-----------------|----------------|
| Total number        | 9185            | 4185           |
| Total runtime(h)    | 25.5            | 11.6           |
| WAV format          | Dual channel, 48kHz sample rate | 24-bit resolution, 10-second duration |
| Acoustic scenes     | Airport, Indoor shopping mall, Metro station, Pedestrian street, Public square, Street with medium level of traffic, Traveling by a tram, Traveling by a bus, Traveling by an underground metro, Urban park |

For scalogram, given $f_h$, $f_l$ and $T_{max}$, $K$ the number of constant-Q-spaced wavelets was set to 241 (Equation (7)). The number of evenly-space passband filter covering the low frequency part was 49 (Equation (8)).

We followed the procedure in [64] to define the Mel filters. The vanilla FBank of a 10-second stereo audio had dimensions of $500 \times 2 \times 128$. The delta and delta-delta coefficients were calculated and stacked along the channel axis to form a feature map of size $500 \times 6 \times 128$.

The short-term FBank exhibited a higher time resolution compared to the long-term scalogram. In order to illustrate the long-term and short-term properties, the cosine similarity was averaged every 2 frames with randomly chosen audio samples. The dimensions of the features were unified with principle component analysis. The similarity values for FBank and the scalogram were 0.80 and 0.54, respectively. This indicates that the FBank extracted by a 40ms window exhibits a greater temporal correlation compared to the scalogram.

### C. Networks

The detailed hyperparameters for the networks in Table I are listed in Table VII. We designed the FCNNs and the segment-wise ACGANs for FBank, the DCNNs and framework ACGANs for the scalogram.

The networks were trained and evaluated using PyTorch toolkit [65]. Adam was used to update the parameters [66]. The batch normalization was performed in 1D and 2D for each channel in DCNNs and FCNNs. A slow version of early stopping was employed for all models trained on the whole training database. More specifically, the learning rate was reduced by 50% following 5 continuous epochs without a reduction of validation loss. The training would be terminated if the validation loss failed to decrease over 15 continuous epochs. The models trained on the training subset during data augmentation, containing $el_{fA}$ and $el_{fB}$, were optimized via a fast early-stopping. The learning rate was reduced to 50% following 3 continuous epochs, the training was terminated after 6 continuous epochs based on the testing subset.
was carried out according to Figure 5, with maximum generated fake samples. The GAN-based data augmentation included real samples extracted from raw signals as well as the feature database used for the training of the final classifiers.

### D. Data augmentation

The data augmentation configuration is reported in Table [VI]. The feature database used for the training of the final classifiers included real samples extracted from raw signals as well as generated fake samples. The GAN-based data augmentation was carried out according to Figure [5] with maximum 10 iterations. The generation and filtering of samples was performed via Algorithm [I]. If no sample successfully passed the sample filter, the current iteration was rejected. If scene classifier $clf^B$ performed worse than $clf^A$, the iteration was also marked as a rejection, otherwise it was accepted. The iterative data augmentation scheme would be terminated if rejected over 3 continuous iterations.

### E. System fusion

Due to the limited dataset size and training stability, the performance of each system was represented as the average voting of the system trained by 3 random seeds. This was able to partially remove randomness and allowed for fair comparisons.

In the experiments, our base classifiers were able to achieve high accuracies. Complex fusion methods, such as the random forest and SVMs, resulted in severe overfitting. Average voting served as a simple yet flexible approach by averaging the segment-wise log-probability of all systems. For DCNNs, the segment-wise log-probability is the mean of the frame-wise ones. Note that for our DCASE19 challenge submission, weighted voting was adopted, which directly used weights tuned on the fold-1 evaluation set. However, this resulted in minimal performance improvements and was thus not considered here.

### VII. RESULTS AND DISCUSSION

In order to evaluate the effectiveness of the FCNN and DCNN architectures, the short-term FBank and long-term scalogram features were tested separately on both neural networks. As demonstrated in the first 4 rows of Table [VII], a poorer performance was observed for FBank-DCNN and scalogram-FCNN compared with FBank-FCNN and scalogram-DCNN. This can be attributed to the convolution methods used for each architecture. FCNNs adopt 2D convolutional kernels that are applied to features with a strong time-scale correlation, while DCNNs adopt 1D convolutional kernels that are applied to features that only exhibit a scale correlation. Moreover, the ave-diff features exhibited higher accuracies compared to the left-right features. This indicates that the averaged and differential sound contains a larger amount of straight-forward scene information. For stereo audios, the sound of a passing car, for example, contains clear spatial information and the differential signal can express this more directly.

Besides the long-term scalogram, 3 additional long-term features were extracted to yield the efficiency of the wavelet scales formulated in Section [III]. A long-term FBank feature was extracted via the same window settings and the same number of filters as that of the scalogram, resulting in the same time-scale resolution. Furthermore, an additional extraction method employed triangle filters on the scale of Equation (11). The start and end frequencies of each triangle filter were set to $f_0$ and $f_h$, respectively. The third long-term scalogram was extracted via the same window settings and the same number of filters as that of the scalogram, resulting in the same time-scale resolution.

Due to the limited dataset size and training stability, the performance of each system was represented as the average voting of the system trained by 3 random seeds. This was able to partially remove randomness and allowed for fair comparisons.
of the long-term FBank did not reach that of the long-term scalogram and short-term FBank. However, with the triangle filters defined on the wavelet scale, the DCNN classifier attained similar accuracies. This indicates that the wavelet scale describes the scene information more accurately than the classic Mel scale on long-term windows, while the filter shape plays a less crucial role. In addition, after the removal of the low-frequency part, the performance degradation highlights the importance of low-frequency filters, which covered 0 to 205 Hz in our experiments.

In summary, the long-term ave-diff scalogram integrated with the DCNN was able to achieve the optimal performance among the different combinations of FBank, scalogram and FCNN, DCNN, exceeding the short-term left-right FBank-FCNN performance by approximately 6%. In order to demonstrate the differences among the short-term FBank, the long-term FBank and the long-term scalogram, we take the audio spectrum depicted in Figure 6 as an example. The scalogram has a lower time resolution but a higher scale resolution. Furthermore, the wavelet scale places more attention on lower frequencies, which may lead to the discovery of additional acoustic cues.

In practical terms, compared with FBank-FCNN, the scalogram-DCNN is the preferred choice for 3 reasons. First, the scalogram occupies less storage space, with dimensions of 58 x 290, approximately 1/4 times of 500 x 128 FBank for a 10-second mono audio. Second, the DCNN architecture with 1D convolution only depends on the current frame, leading to a faster computational speed under similar numbers of network parameters compared with FCNN. Third, for real-life applications, the frame-wise predictions of acoustic scenes are more flexible than the segment-wise ones.

The ave-diff-scalogram-DCNN classifier was chosen to evaluate the data augmentation scheme. We first conducted experiments on the classic data augmentation techniques (row 1-5 in Table VII). The speed and volume perturbation generated duplicated samples with different speed and volume factors. It improved the accuracy on the seen cities by training on more source data but failed on unseen cities. The Mixup and SpecAug interpolated and masked input features to improve the system’s generalization on unseen cities, but the accuracies decreased on seen cities. We tried to combine speed and volume perturbation with SpecAug, but no further improvement occurred.

At each iteration of the proposed GAN-based data augmentation, dataset split, the ACGAN and the neural network-based sample filter play crucial roles in generating fake feature samples. An ablation study of these modules was performed, with results reported in Table VIII. With the absence of the sample filter, the fake features generated by CGAN were not able to improve the performance of the final classifier, despite the accuracy improvements in the testing subset. By replacing CGANs with ACGANs and adding the sample filter, the final classifier was able to perform better. The ACGAN-based data augmentation with the sample filter exhibited the highest accuracies both on the records from seen and unseen cities. In particular, a relative error rate reduction of 6.7% compared with the original ave-diff-scalogram-DCNN system was observed. The evolution of the generated fake scalograms is depicted in Figure 7. The quality of the generated data increased following the iterations, yet it was difficult to de-

![Fig. 6. The FBank and scalogram extracted from the differential channel of a sample recorded in an airport. The orange box demonstrates that the feature extracted by the wavelet scale shows more high-energy details on the low frequency part.](image)

![Fig. 7. The evolution of the generated differential-channel scalograms. The frame-wisely generated samples do not exhibit temporal correlation, but this shortcoming does not affect the DCNN classifier.](image)
term whether it benefited the final classifier, thus requiring the application of the sample filter.

Three methods were employed to split the datasets during ACGAN training, as listed in the last 3 rows in Table VIII. Results demonstrate that the splitting method was crucial to final performance. Although the fixed and random split methods improved the accuracies for the unseen city records, they failed to classify audios from the seen cities. The reason might be that the city-based dataset split optimized the classifiers more robust on the seen city while the data augmentation improves the performance for the unseen cities records. In summary, 3 most important modules of data augmentation are addressed, which are the city-based dataset split, the ACGAN architectures and the sample filter.

The adversary training branch was added to the $cl_{fA}, cl_{fB}$ and final classifier. As shown in Table IX, its performance was inferior to the original classifier (row 1 and 4). Improvements were observed for the extended dataset (row $2−3$ and $5−6$). The results can be attributed to the potential of the adversary training to improve the system performance, yet it should be adopted under data augmentation.

Moreover, the DCT-based temporal module was tested on the final classifiers (Table IX). The temporal module was observed to dramatically decrease the loss on the training and validation, however overfitting might be introduced. Despite inconsistencies in improvements, this approach was adopted for the diversity of the fusion systems.

The highest accuracy without the system fusion was 84.23% on the fold-1 evaluation set with the frame-wise ACGAN-based data augmentation scheme, the ave-diff scalogram as the input feature, the DCNNs as the classifiers, combined with the adversarial city training and the DCT-based temporal module.

The data augmentation scheme was also tested under the short-term FBank scheme. The generator and discriminator for FBank adopted the architectures described in Tables III. The majority of the DCASE19 settings remained unchanged, with the exception of the slight modification of the DCNNs and FCNNs for this bit small dataset. The number of hidden units $h$ for DCNN was 512 (Table VI), while Conv7 and Conv8 (Table I) were removed. As no city information was revealed, the dataset was split according to the recording index and adversarial training was not performed. The detailed results are reported in Table X. The scalogram-DCNN architecture combined with the proposed data augmentation scheme achieved an accuracy of 82.72%. After fusion, the accuracy increased to 84.57%, exceeding the top systems in the competition by 1.3%.

The confusion matrix of the DCASE19 fusion system is presented in Figure 8. Some scenes that are confused by classifiers are also difficult for humans to process, such as public squares and street pedestrians, or airports and shopping malls.
| Feature type       | Channel   | Classifier | DCASE19 Acc. (%) | DCASE17 Acc. (%) |
|-------------------|-----------|------------|------------------|------------------|
| Long-term scalogram | ave-diff  | DCNN       | 84.06            | 82.72            |
| Long-term scalogram | ave-diff  | DCNN + DCT | 83.58            | 81.73            |
| Long-term scalogram | ave-diff  | DCNN + adv | 84.16            | -                |
| Long-term scalogram | ave-diff  | DCNN + adv + DCT | 84.23 | -    |
| Short-term FBank   | left-right| FCNN       | 80.10            | 71.11            |
| Short-term FBank   | ave-diff  | FCNN       | 77.56            | 72.53            |

Average voting: 85.16, 84.57

Fig. 8. The confusion matrix of the fusion system for DCASE19 Task 1A.

VIII. CONCLUSION

We propose a novel ACGAN-based data augmentation scheme for acoustic scene classification. Integrated with the long-term scalogram extracted by the designed wavelet scale, this framework can achieve high accuracies on the testing samples, even for those recorded in unseen cities. However, the filters are still hand-tailored, with more advanced scales existing for ASC tasks. In the future, we will focus on automatically training filters without the occurrence of overfitting.

REFERENCES

[1] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, “Acoustic scene classification: Classifying environments from the sounds they produce,” IEEE Signal Processing Magazine, vol. 32, no. 3, pp. 16–34.

[2] A. Mesaros, T. Heittola, and T. Virtanen, “Tut database for acoustic scene classification and sound event detection,” in Signal Processing Conference, 2016.

[3] A. Mesaros, T. Heittola, A. Dinent, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, “DCASE2017 challenge setup: Tasks, datasets and baseline system,” in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017), November 2017, pp. 85–92.

[4] M. Bugallo, J. Portelo, I. Trancoso, T. Pellegrini, and A. Abad, “Detecting audio events for semantic video search,” in Tenth Annual Conference of the International Speech Communication Association, 2009.

[5] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi,”Audio-based context recognition,” IEEE Transactions on Audio, Speech, and Language Processing, vol. 14, no. 1, pp. 321–329, 2005.

[6] J. Andén and S. Mallat, “Deep scattering spectrum,” CoRR, vol.abs/1304.6763, 2013. [Online]. Available: http://arxiv.org/abs/1304.6763

[7] S. S. Stevens, J. Volkman, and E. B. Newman, “A scale for the measurement of the psychological magnitude pitch,” J. acoust. soc. am., vol. 8, no. 3, pp. 185–190, 1937.

[8] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” IEEE transactions on acoustics, speech, and signal processing, vol. 28, no. 4, pp. 357–366, 1980.

[9] S. Mallat, A wavelet tour of signal processing. Elsevier, 1999.

[10] H. Chen, P. Zhang, H. Bai, Q. Yuan, X. Bao, and Y. Yan, “Deep convolutional neural network with scalogram for audio scene modeling.” in Interspeech, 2018, pp. 3304–3308.

[11] H. Chen, P. Zhang, and Y. Yan, “An audio scene classification framework with embedded filters and a dct-based temporal module,” in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 835–839.

[12] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, “Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 26, no. 2, pp. 379–393, 2017.

[13] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, “Audio augmentation for speech recognition,” in Sixteenth Annual Conference of the International Speech Communication Association, 2015.

[14] M. Dorfer, B. Lehner, H. Eghbal-zadeh, H. Christop, P. Fabian, and W. Gerhard, “Acoustic scene classification with fully convolutional neural networks and I-vectors,” DCASE2018 Challenge, Tech. Rep., September 2018.

[15] S. Mun, S. Park, D. Han, and H. Ko, “Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane,” DCASE2017 Challenge, Tech. Rep., September 2017.

[16] N. Sawhney and P. Maes, “Situational awareness from environmental sounds,” Project Rep. for Pattie Maes Project Rep. for Pattie Maes, pp. 1–7, 1997.

[17] B. Clarkson, N. Sawhney, and A. Pentland, “Auditory context awareness via wearable computing,” Energy, vol. 400, no. 600, p. 20, 1998.

[18] A. Eronen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, “Audio-based context awareness-acoustic modeling and perceptual evaluation,” in 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03)., vol. 5. IEEE, 2003, pp. V–529.

[19] M. D. Plumbley, C. Kroos, J. P. Bello, G. Richard, D. P. Ellis, and A. Mesaros, Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018), Tampere University of Technology. Laboratory of Signal Processing, 2018.

[20] A. Mesaros, T. Heittola, and T. Virtanen, “Acoustic scene classification in dcase 2019 challenge: Closed and open set classification and data mismatch setups,” in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), New York University, NY, USA, October 2019, pp. 164–168.

[21] Y. Han and J. Park, “Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification,” DCASE2017 Challenge, Tech. Rep., September 2017.

[22] A. Mesaros, T. Heittola, and T. Virtanen, “A multi-device dataset for urban acoustic scene classification,” in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2018), November 2018, pp. 9–13.

[23] Y. Sakashita and M. Aono, “Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions,” DCASE2018 Challenge, Tech. Rep., September 2018.

[24] H. Chen, Z. Liu, Z. Liu, P. Zhang, and Y. Yan, “Integrating the data augmentation scheme with various classifiers for acoustic scene modeling,” DCASE2019 Challenge, Tech. Rep., June 2019.

[25] Z. Weiping, Y. Jiantao, X. Xiaotao, L. Xiangtao, and P. Shaohu, “Acoustic scene classification using deep convolutional neural network
and multiple spectrograms fusion,” DCASE2017 Challenge, Tech. Rep., September 2017.

[26] V. Bisot, R. Serizel, S. Essid, and G. Richard, “Supervised nonnegative matrix factorization for acoustic scene classification,” DCASE2016 Challenge, Tech. Rep., September 2016.

[27] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, “CP-JKU submissions for DCASE2016: a hybrid approach using binaural i-vectors and deep convolutional neural networks,” DCASE2016 Challenge, Tech. Rep., September 2016.

[28] J. C. Brown, “Calculation of a constant q spectral transform,” The Journal of the Acoustical Society of America, vol. 89, no. 1, pp. 425–434, 1991.

[29] F. Argenti, P. Nesi, and G. Puntaloro, “Automatic transcription of polyphonic music based on the constant-q bi-pitch detection analysis,” Transactions on Audio, Speech, and Language Processing, vol. 19, no. 6, pp. 1610–1630, 2010.

[30] D. Giannoulis, D. Barchiesi, A. Klapuri, and D. M. Plumbley, “On the disjointness of sources in music using different time-frequency representations,” in 2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE, 2011, pp. 261–264.

[31] T. Lidy and A. Schindler, “CQT-based convolutional neural networks for audio scene classification and domestic audio tagging,” DCASE2016 Challenge, Tech. Rep., September 2016.

[32] Q. Kun, R. Zhao, P. Vedhas, Y. Zijiang, Z. Zixing, and S. Bijn, “Wavelet features for the classification of acoustic scenes,” DCASE2017 Challenge, Tech. Rep., September 2017.

[33] S. Amiriparian, N. Cummins, M. Freitag, K. Qian, Z. Ren, V. Pandit, and B. Schuller, “The combined augsburg/passau/tum/icl system for dcase 2017,” DCASE2017 Challenge, Tech. Rep, Tech. Rep., 2017.

[34] H. Zeinali, L. Burget, and H. Cernocky, “Convolutional neural networks and x-vector embedding for dcase2018 acoustic scene classification challenge,” DCASE2018 Challenge, Tech. Rep., September 2018.

[35] Z. Huang and D. Jiang, “Acoustic scene classification based on deep convolutional neural network with spatial-temporal attention pooling,” DCASE2019 Challenge, Tech. Rep., June 2019.

[36] V. Bisot, R. Serizel, S. Essid, and G. Richard, “Nonnegative feature learning methods for acoustic scene classification,” DCASE2017 Challenge, Tech. Rep., September 2017.

[37] S. Chu, S. Narayanan, C.-C. J. Kuo, and M. J. Matarić, “Where am i? scene recognition for mobile robots using audio features,” in 2006 IEEE International conference on multimedia and expo. IEEE, 2006, pp. 885–888.

[38] L. Ma, B. Milner, and S. Dan, “Acoustic environment classification,” Acm Transactions on Speech & Language Processing, vol. 3, no. 2, pp. 1–22, 2006.

[39] B. Eltzalde, A. Kumar, A. Shah, R. Badlani, E. Vincent, B. Raj, and I. Lane, “Experiments on the DCASE challenge 2016: Acoustic scene classification and sound event detection in real life recording,” DCASE2016 Challenge, Tech. Rep., September 2016.

[40] A. Rakotomamonjy and G. Giasso, “Histogram of gradients of time–frequency representations for audio scene classification,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 23, no. 1, pp. 142–153, 2014.

[41] J. Ye, T. Kobayashi, M. Murakawa, and T. Higuchi, “Acoustic scene classification based on sound textures and events,” in Proceedings of the 23rd ACM international conference on Multimedia. ACM, 2015, pp. 1291–1294.

[42] S. Mun, S. Park, Y. Lee, and H. Ko, “Deep neural network bottleneck feature for acoustic scene classification,” DCASE2016 Challenge, Tech. Rep., September 2016.

[43] N. Moritz, J. Schader, S. Goetzke, J. Anemiller, and B. Kollmeier, “Acoustic scene classification using time-delay neural networks and amplitude modulation filter bank features,” DCASE2016 Challenge, Tech. Rep., September 2016.

[44] I. Kukanov, V. Hautamaki, and K. A. Lee, “Recurrent neural network and maximal figure of merit for acoustic event detection,” DCASE2017 Challenge, Tech. Rep., September 2017.

[45] L. Mingjie and L. Xingxiang, “The system for acoustic scene classification using resnet,” DCASE2019 Challenge, Tech. Rep., June 2019.

[46] H. Zhu, C. Ren, J. Wang, S. Li, L. Wang, and L. Yang, “DCASE 2019 challenge task1 technical report,” DCASE2019 Challenge, Tech. Rep., June 2019.

[47] W. Wang and M. Liu, “The SEIE-SUT systems for acoustic scene classification using CNN ensemble,” DCASE2019 Challenge, Tech. Rep., June 2019.

[48] http://dcase.community/challenge2018/task-acoustic-scene-classification-results-a

[49] http://dcase.community/challenge2019/task-acoustic-scene-classification-results-a

[50] J. Li, W. Dai, F. Metze, S. Qu, and S. Das, “A comparison of deep learning methods for environmental sound,” 2017.

[51] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672–2680.

[52] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” arXiv preprint arXiv:1411.1784, 2014.

[53] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 2642–2651.

[54] A. Makhzani, J. Shlens, N. Jaitly, and I. J. Goodfellow, “Adversarial autoencoders,” CoRR, vol. abs/1511.05644, 2015. [Online]. Available: http://arxiv.org/abs/1511.05644

[55] E. L. Denton, S. Chintala, R. Fergus et al., “Deep generative image models using a laplacian pyramid of adversarial networks,” in Advances in neural information processing systems, 2015, pp. 1486–1494.

[56] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[57] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang et al., “Photo-realistic single image super-resolution using a generative adversarial network,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4681–4690.

[58] A. Antoniou, A. Starkey, and H. Edwards, “Data augmentation generative adversarial networks,” arXiv preprint arXiv:1711.04340, 2017.

[59] X. Xia, R. Togneri, F. Sobel, and D. Huang, “Auxiliary classifier generative adversarial network with soft labels in imbalanced acoustic event detection,” IEEE Transactions on Multimedia, vol. 21, no. 6, pp. 1359–1371, 2018.

[60] X. Zheng and J. Yan, “Acoustic scene classification combining log-mel CNN model and end-to-end model,” DCASE2019 Challenge, Tech. Rep., June 2019.

[61] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” international conference on learning representations, 2015.

[62] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” Journal of Machine Learning Research, vol. 17, no. 1, pp. 2096–2030, 2015.

[63] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” Computer Science, 2015.

[64] D. U. Young, G. Evermann, M. Gales, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, “The bk book version 3.4,” Cambridge University Engineering Department, 2006.

[65] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.

[66] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” Computer Science, 2014.

[67] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” CoRR, vol. abs/1710.09412, 2017. [Online]. Available: http://arxiv.org/abs/1710.09412

[68] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, “SqueezeNet: A simple data augmentation method for automatic speech recognition,” Interspeech 2019, Sep 2019. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680