applied sciences

MDPI

Article

An Ensemble of Convolutional Neural Networks for Audio Classification

Loris Nanni [1],*, Gianluca Maguolo [1], Sheryl Brahnam [2] and Michelangelo Paci [3]

[1] Department of Information Engineering, University of Padua, 35122 Padua, Italy; gianluca.maguolo@dei.unipd.it

[2] Department of Information Technology and Cybersecurity, Missouri State University, Springfield, MO 65804, USA; sbrahnam@missouristate.edu

[3] BioMediTech, Faculty of Medicine and Health Technology, Tampere University, Arvo Ylpön katu 34, FI-33520 Tampere, Finland; michelangelo.paci@tuni.fi

* Correspondence: loris.nanni@unipd.it; Tel.: +39-373-85-35-801

Abstract: Research in sound classification and recognition is rapidly advancing in the field of pattern recognition. One important area in this field is environmental sound recognition, whether it concerns the identification of endangered species in different habitats or the type of interfering noise in urban environments. Since environmental audio datasets are often limited in size, a robust model able to perform well across different datasets is of strong research interest. In this paper, ensembles of classifiers are combined that exploit six data augmentation techniques and four signal representations for retraining five pre-trained convolutional neural networks (CNNs); these ensembles are tested on three freely available environmental audio benchmark datasets: (i) bird calls, (ii) cat sounds, and (iii) the Environmental Sound Classification (ESC-50) database for identifying sources of noise in environments. To the best of our knowledge, this is the most extensive study investigating ensembles of CNNs for audio classification. The best-performing ensembles are compared and shown to either outperform or perform comparatively to the best methods reported in the literature on these datasets, including on the challenging ESC-50 dataset. We obtained a 97% accuracy on the bird dataset, 90.51% on the cat dataset, and 88.65% on ESC-50 using different approaches. In addition, the same ensemble model trained on the three datasets managed to reach the same results on the bird and cat datasets while losing only 0.1% on ESC-50. Thus, we have managed to create an off-the-shelf ensemble that can be trained on different datasets and reach performances competitive with the state of the art.

Keywords: audio classification; data augmentation; ensemble of classifiers; pattern recognition

check for updates

Citation: Nanni, L.; Maguolo, G.; Brahnam, S.; Paci, M. An Ensemble of Convolutional Neural Networks for Audio Classification. *Appl. Sci.* **2021**, *11*, 5796. https://doi.org/10.3390/app11135796

Academic Editor: Sławomir K. Zieliński

Received: 24 May 2021
Accepted: 16 June 2021
Published: 22 June 2021

1. Introduction

Sound classification and recognition have long been included in the field of pattern recognition. Some of the more popular application domains include speech recognition [1], music classification [2], biometric identification [3], and environmental sound recognition [4], the topic of interest in this work. Following the three classical pattern recognition steps of (i) preprocessing, (ii) feature/descriptor extraction, and (iii) classification, most early work in sound classification began by extracting features from audio recordings such as the Statistical Spectrum Descriptor or Rhythm Histogram [5]. Once it was recognized, however, that visual representations of audio, such as spectrograms [6] and Mel-frequency Cepstral Coefficients spectrograms (Mel) [7], contain valuable information, powerful texture extraction techniques like local binary patterns (LBP) [8] and its many variants [9] began to be explored for audio classification [2,10]. In [2], for example, ensembles of classifiers designed to fuse sets of the most robust texture descriptors with acoustic features extracted from the audio traces on multiple datasets were exhaustively investigated; this study showed that the accuracy of systems based solely on acoustic or visual features could be enhanced by combining texture features.

Recently, deep learners have proven even more robust in image recognition and classification than have texture analysis techniques. In audio biodiversity assessment, for example, a task that intends to monitor animal species at risk, convolution neural networks (CNNs) have greatly enhanced the performance of animal [11] and bird identification [12–14]. Deep learners have also been adapted to identify the sounds of marine animals [11] and fish [15]. In both these works, the authors combined CNN with visual features; the fusion of CNNs with traditional techniques was shown to outperform both the stand-alone conventional and single deep learning approaches.

Another environmental audio recognition problem that is growing in relevancy has to do with identifying sources of noise in environments. This audio classification problem is of particular concern for cell phone developers since noise interferes with conversation. Consequently, datasets of extraneous sounds have been released to develop systems for handling different kinds of noise. The Environmental Sound Classification (ESC-50) dataset, for instance, contains 2000 labeled samples divided into fifty classes of environmental sounds that range from dogs barking to the sound of sea waves and chainsaws. In [16], a deep CNN achieved results superior to human classification on this dataset. Other work of interest in this area includes [17–21].

For all its power, deep learning also has significant drawbacks when it comes to environmental sound classification. For one, deep learning approaches require massive training data [22]. For audio classification, this means collecting large numbers of labeled audio signals and transforming them into visual representations, a task that is prohibitively expensive and labor-intensive. Fortunately, there are methods for increasing the number of images in small datasets. One such method is to apply data augmentation techniques. Audio signals can be augmented in both the time and frequency domains, and these augmentation techniques can be directly applied either on the raw signals themselves or on the images obtained after they have been converted into spectrograms. In [23], for example, several augmentation techniques were applied to the training set in the BirdCLEF 2018 dataset. The augmentation pipeline involved taking the original bird audio signals, chunking them, and then augmenting them in the time domain (e.g., by adding background/atmospheric noise) and in the frequency domain (e.g., by applying pitch shifts and frequency stretches). This augmentation process not only enlarged the dataset but also produced nearly a 10% improvement in identification performance. Similarly, some standard audio augmentation techniques such a time and pitch shifts for bird audio classification were applied in [24]. Samples were also generated in [24] by summing separate samples belonging to the same class. This summing technique was used for domestic sound classification in [25,26]. In [27], new data was generated by computing the weighted sum of two samples belonging to different classes and teaching the network to predict the weights of the sum. Audio signal augmentation on a domestic cat sound dataset was produced in [28] by random time stretching, pitch shifting, compressing the dynamic range, and inserting noise. Data augmentation techniques that are standard in speech recognition have also proven beneficial for animal sound identification, as in [29,30].

The goal of this work is to perform an extensive study of CNN classification using different architectures combined with an investigation of multiple sets of different data augmentation approaches and methods for representing audio signals as images. Building such ensembles is motivated by two observations: (1) it is well known that ensembles of neural networks generally perform better than stand-alone models due to the instability of the training process [31], and (2) it has been shown in other classification tasks that an ensemble of multiple networks trained with different augmentation protocols performs much better than do stand-alone networks [32]. The classification scores of the CNNs are combined by sum rule and compared and tested across three different audio classification datasets: domestic cat sound classification [28,33], bird call classification [34], and environmental noise classification [4].

The main contribution of this study is the exhaustive tests performed on ensembles made by fusing (see Figure 1) five CNNs trained with four audio representations of raw sound signals combined with six different data augmentation methods (totaling thirty-five subtypes). The performances of the best performing ensembles are compared across the three datasets and with the state of the art. In the fair comparisons, our method outperforms all others. Another contribution of this work is the free availability of the MATLAB source code used in this study, available at https://github.com/LorisNanni.

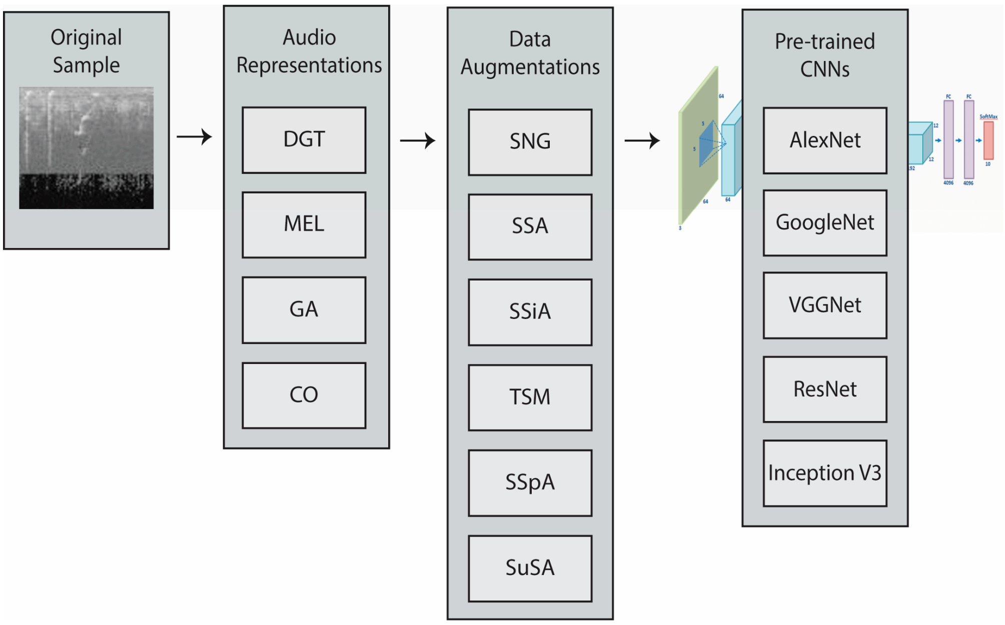

Figure 1. Proposed system. Many ensembles are generated by fine-tuning five pre-trained convolution neural networks (CNNs) separately on four possible audio representations combined with six main data augmentation methods. The best combinations are experimentally derived. The arrows indicate one possible combination of methods for building ensembles.

2. Audio Image Representations

Since the input to a CNN is in the form of a matrix, the following four methods were used to map the audio signals into spectrograms:

1. The Discrete Gabor Transform (DGT): this is a short-time Fourier transform (STFT) with a Gaussian kernel as the window function. The continuous version of DGT can be defined as the convolution between the product of the signal with a complex exponential and a Gaussian, as:

$$G(\tau,\omega) = \frac{1}{\sigma^2}\int_{-\infty}^{+\infty} x(t)e^{i\omega t}e^{-\pi\sigma^2(t-\tau)^2}\,dt, \tag{1}$$

   where s(t) is the signal, ω is a frequency, and i is the imaginary unit. The width of the Gaussian window is defined by σ^2. The discrete version of DGT applies the discrete convolution rather than the continuous convolution. The output $G(\tau,\ \omega)$ is a matrix, where the columns represent the frequencies of the signal at a fixed time. The DGT implementation used in this study (see, [35]) is available at http://ltfat.github.io/doc/gabor/sgram.html (accessed on 6 January 2021).

2. Mel spectrograms (MEL) [36]: these spectrograms are computed by extracting the coefficients relative to the compositional frequencies with STFT. Extraction is accomplished by passing each frame of the frequency-domain representation through a Mel filter bank (the idea is to mimic the non-linear human ear perception of sound, which discriminates lower frequencies better than higher frequencies). Conversion between Hertz (f) and Mel (m) is defined as:

$$m = 2595\log_{10}(1+700f). \tag{2}$$

The filters in the filter bank are all triangular, which means that each has a response of 1 at the center frequency, which decreases linearly towards 0 until it reaches the center frequencies of the two adjacent filters, where the response is 0.

3. Gammatone (GA) band-pass filters: this is a bank of GA filters whose bandwidth increases with the increasing central frequency. The functional form of Gammatone is inspired by the response of the cochlea membrane in the inner ear of the human auditory system [37]. The impulse response of a Gammatone filter is the product of a statistical distribution (Gamma) and a sinusoidal carrier tone. This response can be defined as:

$$h_i(t) = \begin{cases} a \cdot t^{n-1} e^{-2\pi B_i t} cos(2\pi\omega_i t) + \phi & , t \geq 0 \\ 0 & , t < 0 \end{cases} \tag{3}$$

   where ω_i is the central frequency of the filter and ϕ its phase. Gain is controlled by the constant a, and n is the order of the filter. B_i is a decay parameter that determines the bandwidth of the band-pass filter.
4. Cochleagram (CO): this mapping models the frequency selectivity property of the human cochlea [38]. To extract a cochleagram, it is first necessary to filter the original signal with a gammatone filter bank (see, Equation (3) above). The filtered signal must then be divided into overlapping windows. For each window and every frequency, the energy of the signal is calculated.

Each of the four spectrograms is then mapped to a gray-scale image using a linear transformation that maps the minimum value to 0 and the maximum value to 255, with the value of each pixel rounded to the closest smaller integer.

3. Convolutional Neural Networks (CNNs)

Aside from the input and output layers, CNNs are composed of one or more of the following specialized hidden layers: convolutional (CONV), activation (ACT), pooling (POOL), and fully-connected (FC), or classification layer. The CONV layers pull out features from the input volume and work by convolving a local region of the input volume (the receptive field) to filters of the same size. Once the convolution is computed, these filters slide into the next receptive field, where once again, the convolution between the new receptive field and the same filter is computed. This process is iterated over the entire input image, whereupon it produces the input for the next layer, a non-linear ACT layer, which improves the learning capabilities and classification performance of the network. Typical activation functions include (i) the non-saturating Rectified Linear Activation Function (ReLU) function $f(x) = \tan\text{h}(x)$, (ii) the saturating hyperbolic tangent $f(x) = \tan\text{h}(x)$, $f(x) = |\tan\text{h}(x)|$, and (iii) the sigmoid function $f(x) = (1 + e^{-x})^{-1}$. Pool layers are often interspersed between CONV layers and perform non-linear downsampling operations (max or average pool) that serve to reduce the spatial size of the representation, which in turn has the benefit of reducing the number of parameters, the possibility of overfitting, and the computational complexity of the CNN. FC layers typically make up the last hidden layers and have FC neurons to all the activations in the previous layer. SoftMax is generally used as the activation function for the output CLASS layer, which performs the final classification (also typically using the SoftMax function).

In this study, five CNNs pre-trained on ImageNet [39] or Places365 [40] are adapted to the problem of environmental sound classification as defined in the datasets used in this work. The architecture of the following pre-trained CNNs remains unaltered except for the last three layers, which are replaced by an FC layer, an ACT layer using SoftMax, and a CLASS layer also using SoftMax:

1. AlexNet [41] was the first neural network to win (and by a large margin) the ILSVRC 2012 competition. AlexNet has a structure composed of five CONV blocks followed by three FC layers. The dimension of the hidden layers in the network is gradually

reduced with max-pooling layers. The architecture of AlexNet is simple since every hidden layer has only one input layer and one output layer.

2. GoogleNet [42] is the winner of ILSVRC 2014 challenge. The architecture of GoogleNet involves twenty-two layers and five POOL layers. GoogleNet was unique in its introduction of a novel Inception module, which is a subnetwork made up of parallel convolutional filters. Because the output of these filters is concatenated, the number of learnable parameters is significantly reduced. This study uses two pre-trained GoogleNets: the first is trained on the ImageNet database [39], and the second is trained on the Places365 [40] datasets.
3. VGGNet [43] is a CNN that took second place in ILSVRC 2014. Because VGGNet includes 16 (VGG-16) or 19 (VGG-19) CONV/FC layers, it is considered extremely deep. All the CONV layers are homogeneous. Unlike AlexNet [41], which applies a POOL layer after every CONV layer, VGGNet is composed of relatively tiny 3 × 3 convolutional filters with a POOL layer applied every two to three CONV layers. Both VGG-16 and VGG-19 are used in this study, and both are pre-trained on the ImageNet database [39].
4. ResNet [44] was the winner of ILSVRC 2015 and is much deeper than VGGNet. ResNet is distinguished by introducing a novel network-in-network architecture composed of residual (RES) layers. ResNet is also unique in applying global average pooling layers at the end of the network rather than the more typical set of FC layers. These architectural advances produce a model that is eight times deeper than VGGNet yet significantly smaller in size. Both ResNet50 (a 50-layer residual network) and ResNet101 (the deeper variant of ResNet50) are investigated in this study. Both CNNs have an input size of 224 × 224 pixels.
5. InceptionV3 [45] advances GoogleNet by making the auxiliary classifiers perform as regulators rather than as classifiers. This is accomplished by factorizing 7 × 7 convolutions into two or three consecutive layers of 3 × 3 convolutions and applying the RMSProp Optimizer. InceptionV3 accepts images of size 299 × 299 pixels.

In Table 1, we report the specifications of the different network architectures that we used in our paper.

Table 1. Specifications of the different network architectures.

Networks	Layers	Parameters	Size	Input Size
AlexNet	8	61 million	227 Mb	227 × 227
GoogleNet	22	7 million	27 Mb	224 × 224
VGG16	16	138 million	515 Mb	224 × 224
VGG19	19	147 million	535 Mb	224 × 224
ResNet50	50	25.6 million	96 Mb	224 × 224
ResNet101	101	44.6 million	167 Mb	224 × 224
Inception	48	23.9 million	89 Mb	224 × 224

4. Data Augmentation Methods

Below is a description of the augmentation protocols that were combined and tested in this study. For each data augmentation method used to train a CNN, both the original and the artificially generated patterns were included in the training set.

4.1. Standard Signal Augmentation (SGN)

Standard signal augmentation (SGN) is the application of the built-in data augmentation methods for audio signals available in MATLAB (see samples in Figure 2). For each training signal, ten new ones are generated by applying the following labeled transformations with 50% probability:

1. SpeedupFactorRange scales the speed of the signal by a random number in the range of [0.8, 1.2];

2. SemitoneShiftRange shifts the pitch of the signal by a random number in the range of [−2, 2] semitones;
3. VolumeGainRange increases or decreases the gain of the signal by a random number in the range of [−3, 3] dB;
4. SNR injects random noise into the signal in the range of [0, 10] dB;
5. TimeShiftRange shifts the time of the signal in the range of [−0.005, 0.005] seconds.

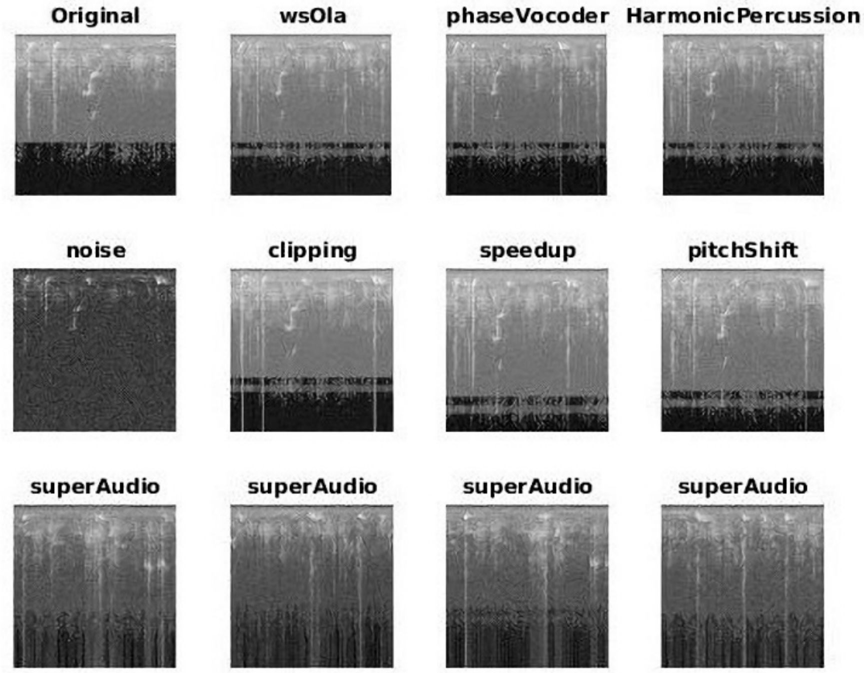

Figure 2. Audio augmented samples, where the last row shows the iterative applications of multiple random augmentations (SGN, SSA, SSiA, and TSM). The horizontal axis in each image represents time, and the vertical axis represents frequency.

4.2. Short Signal Augmentation (SSA)

Short signal augmentation (SSA) works directly on the raw audio signals (see Figure 2 for some samples). For every original image, the following ten augmentations are applied to produce ten new images:

1. applyWowResampling implements wow resampling, a variant of pitch shift that changes the intensity in time. The formula for the wow transformation is:

$$F(s) = s + a_m \frac{\sin(2\pi f_m s)}{2\pi f_m} \tag{4}$$

   where s is the input signal. In this study, $a_m = 3$ and $f_m = 2$;
2. applyNoise is the insertion of white noise so that the ratio between the signal and the noise X dB (here, X = 10);
3. applyClipping normalizes the audio signal by leaving 10% of the samples out of [−1, 1], with the out-of-range samples (x) clipped to sign (x)
4. applySpeedUp not only increases but also decreases the speed of the audio signal; in this study, the speed was augmented by 15%;
5. applyHarmonicDistortion is the repeated application of quadratic distortion to the signal; in this study, the following distortion was applied five consecutive times:

$$s_{out} = \sin\left(2\pi s_{in}\right); \tag{5}$$

6. applyGain increases the gain by a specific number of decibels (set here to 10 dB);

7. applyRandTimeShift randomly divides each audio signal into two signals and swaps them by mounting them back into a randomly shifted signal. If we call $s_{in}(t)$ the value of the input audio signal at time t, T is the length of the signal and t^* is a random time between 0 and T:
$$s_{out}(t) = s_{in}(\mathrm{mod}(t^* + t, T)) \tag{6}$$
8. applyDynamicRangeCompressor applies dynamic range compression (DRC) [46] to a sample audio signal. DRC boosts the lower intensities of an audio signal and attenuates the higher intensities by applying an increasing piecewise linear function. DRC, in other words, compresses an audio signal's dynamic range;
9. applyPitchShift shifts the pitch of an audio signal by a specific number of semitones. We chose to increase it by two semitones;
10. We use applyPitchShift again to decrease the pitch of the audio signal by two semitones.

4.3. Super Signal Augmentation (SSiA)

With this protocol, twenty-nine new images are generated from every original image. The following five augmentations are applied to every sample, with the parameters of the augmentations randomized to generate the new images:

1. applyWowResampling, as in SSA;
2. applySpeedUp, as in SSA; but, in this case, the speed is either increased or decreased by a random number of percentage points in the range $[-5,\ 5]$;
3. applyGain, as in SSA, but the gain factor is sampled randomly in the range of $[-0.5,\ 0.5]$;
4. applyRandTimeShift, as in SSA;
5. applyPitchShift, as in SSA, but the pitch is shifted in the range of $[-0.5,\ 0.5]$.

Small parameters are selected because applying multiple transformations to the input introduces changes that are too large. The difference between the protocols in SSiA and SSA is that the SSiA protocols create a large number of images through multiple small transformations (See Figure 2). Conversely, the images created by SSA protocols are generated with only one large transformation.

4.4. Time-Scale Modification (TSM)

This protocol applies the five algorithms contained in the time-scale modification (TSM) toolbox [47]. TSM methods (samples in Figure 2) are commonly used in music production software to change the speed of signals without changing their pitch. Since two different constant stretching factors (0.5 and 1.8) were used for each TSM method, this augmentation approach produced ten new images. For a detailed description of the TSM toolbox, see [48]. A brief description of the five TSM algorithms follows:

1. OverLap Add (OLA): this algorithm is the simplest TSM method. It covers the input signals with overlapping windows of size H_a and maps them into overlapping windows of size H_s. The number H_α depends on the implementation of the algorithm, while the ratio $\alpha = {}^{H_s}/_{H_a}$ is the speed-up factor, which can be optionally set. The settings investigated in this study were 0.8 and 1.5. These same values were used for each TSM method;
2. Waveform Similarity OverLap Add (WSOLA): this is a modification of OLA where the overlap of the windows is not fixed but has some tolerance to better represent the output signal in cases where there is a difference of phase;
3. Phase Vocoder addresses the same phase problem as WSOLA. However, it exploits the dual approach by matching the windows in the frequency domain: first, the Fourier transforms of the signal are calculated; second, the frequencies are matched, and the signal is pulled back into the time domain;

4. Phase Vocoder with identity phase locking: this TSM method is a modification of Phase Vocoder where the frequencies are matched as if they were not independent of each other. This modification was introduced by Laroche and Dolson [49];
5. Harmonic-Percussive Source Separation (HPSS): this augmentation technique decomposes an audio signal into its harmonic sound components, which form structures in the time direction, and its percussive sounds, which yield structures in the frequency direction. After decomposing the signal in this way, the phase vocoder is applied with the identity phase locking to the harmonic component, and OLA is applied to the percussive component. Finally, these two components are merged to form a new signal.

4.5. Short Spectrogram Augmentation (SSpA)

Short spectrogram augmentation (see samples in Figure 3): works directly on spectrograms and generates five transformed versions of each original

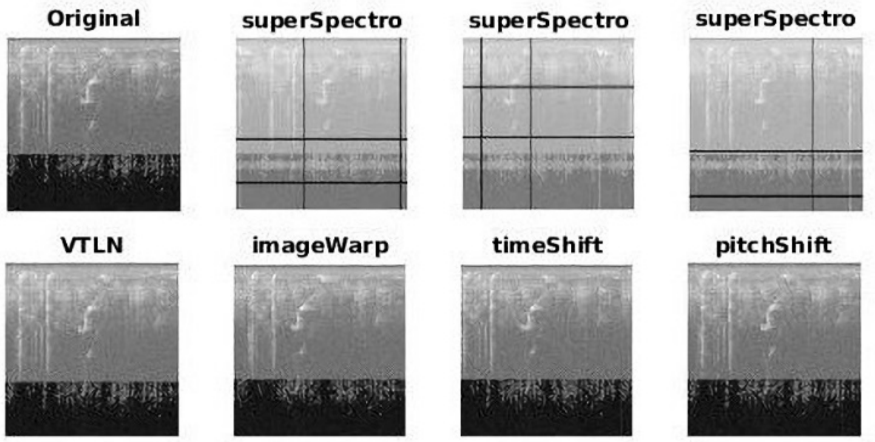

Figure 3. Spectrogram augmented samples (SSpa and SuSA). The horizontal axis in each image represents time, and the vertical axis represents frequency.

1. applySpectrogramRandomShifts randomly applies pitch shift and time shift;
2. VocalTractLengthNormalization (apply VTLN) creates a new image by applying VTLP [29], which divides a given spectrogram into 10 unique temporal slices. Once so divided, each slice passes through the following transformation:

$$G(f) = \begin{cases} \alpha f & , 0 \leq f < f_0 \\ \frac{f_{max} - \alpha f_0}{f_{max} - f_0}(f - f_0) + \alpha f_0 & , f_0 \leq f \leq f_{max} \end{cases} , \tag{7}$$

where f_0, f_{max} are the basic and maximum frequency, and $\alpha \in [a, b]$ is randomly chosen. In this study, a and b are set to 0.9 and 1.1, respectively;
3. applyRandTimeShift does what its name indicates by randomly picking the shift value T in $[1, \mathrm{M}]$, where M is the horizontal size of the input spectrogram. A given spectrogram is cut into two different images: S_1 and S_2, the first taken before and second after time T. The new image is generated by inverting the order of S_1 and S_2;
4. applyRandomImageWarp creates a new image by applying thin-spline image warping (TPS-Warp) [50] to a given spectrogram. TPS-Warp is a perturbation method applied to the original image by randomly changing the position of a subset S of the input pixels. It adapts pixels that do not belong to S via linear interpolation. In this study, the spectrogram is changed on the horizontal axis only. Also, a frequency-time mask is applied by setting to zero the values of two rows and one column of the spectrogram. In this study, the width of the rows is set to 5 pixels and the width of the column to 15 pixels;

5. applyNoiseS applies pointwise random noise to spectrograms. The value of a pixel is multiplied by a uniform random variable of average one and variance one, with a probability 0.3

4.6. Super Spectro Augmentation (SuSA)

In this protocol, 29 new images are generated from each original sample. The following five augmentation methods are applied to each signal, with parameters randomized to produce different samples (see samples in Figure 3):

1. applySpectrogramRandomShifts as in SSpA, but with the time shift equal to zero and random pitch shift in the range $[-1,\ 1]$;
2. applyVTLN as in SSpA;
3. applyRandTimeShift as in SSpA;
4. applyFrequencyMasking sets to zero at most two random columns (which represent times) and at most two random rows (which represent frequencies);
5. applyNoiseS applies pointwise random noise to spectrograms. The value of a pixel is multiplied by a uniform random variable in $[0.3,\ 1.7]$ with a probability of 0.1.

5. Experimental Results

5.1. Datasets

The approach presented here was tested on three sound datasets (see Figure 4 for some examples):

- BIRDZ [34]: a control audio dataset, where recordings in the wilds were downloaded from the Xeno-canto Archive (http://www.xeno-canto.org/ accessed on 6 January 2021). BIRDZ contains 2762 bird acoustic events with 339 detected “unknown” events that are either noise or other vocalizations aside from the eleven labeled North American bird species. Many spectrogram types (constant frequency, broadband with varying frequency components, broadband pulses, frequency modulated whistles, and strong harmonics) are included;
- CAT [28,33]: a balanced dataset of 300 samples of 10 classes of cat vocalizations collected from Kaggle, YouTube, and Flickr. The average duration of each sound is ~ 4 s;
- ESC-50 [4]: an environmental sound classification dataset with 2000 samples evenly divided into 50 classes and five folds; each fold contains eight samples. The five folds with class names in parentheses are *Animals* (Dog, Rooster, Pig, Cow, Frog, Cat, Hen, Flying Insects, Sheep, and Crow), *Natural soundscapes and water sounds* (Rain, Sea waves, Crackling fire, Crickets, Chirping birds, Water drops, Wind, Pouring water, Toilet flush, and Thunderstorm), *Human non-speech* (Crying baby, Sneezing, Clapping, Breathing, Coughing, Footsteps, Laughing, Brushing teeth, Snoring, and Drinking/sipping), *Interior and domestic sounds* (Door knock, Mouse click, Keyboard typing, Door, wood creaks, Can opening, Washing machine, Vacuum cleaner, Clock alarm, Clock tick, and Glass breaking), and *Exterior and urban noises* (Helicopter, Chainsaw, Siren, Car horn, Engine, Train, Church bells, Airplane, Fireworks, and Hand saw).

The data augmentation techniques explored in this study are assessed on each dataset using the same testing protocols described in the original papers. The recognition rate (the average accuracy across all folds) is used as the performance indicator. Also, it should be noted that many papers [16,27,46,48,51–53] report classification results on the datasets listed above that are superior to human performance.

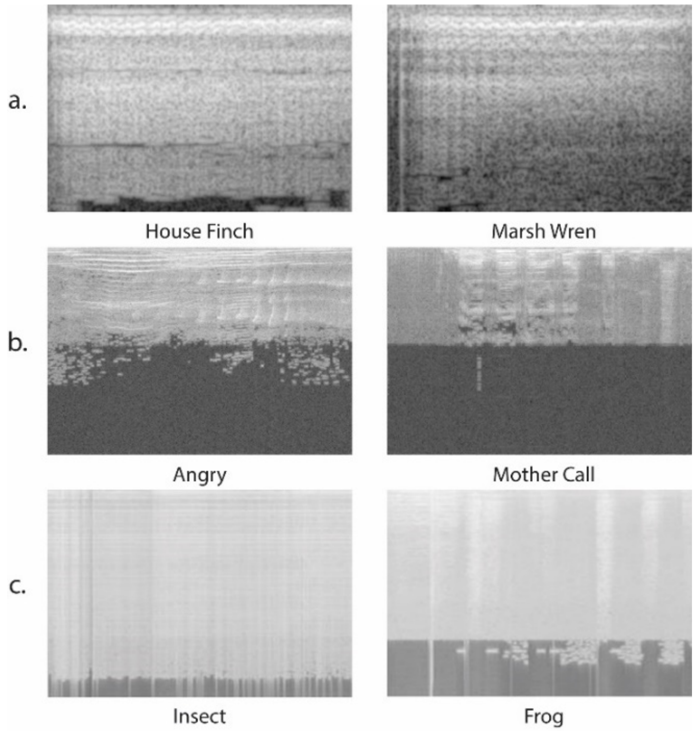

Figure 4. Spectrogram samples from (**a**) BIRDZ, (**b**) CAT, and (**c**) ESC-50 illustrating the applications of identifying bird songs, cat calls, and environmental noise. The horizontal axis in each image represents time, and the vertical axis represents frequency. Note that the frequency scale has been reversed in this figure.

5.2. Results

In Tables 2–5, the accuracy obtained by some of the data augmentation protocols is reported and compared with the baseline that skips the augmentation step (NoAUG). The CNNs were trained for 30 epochs with a learning rate of 0.0001, except for the last fully connected layer that has a learning rate of 0.001 and a batch size of 60. The one exception is the CNN labeled 'VGG16BatchSize', the standard VGG16 with a fixed batch size of 30. For NoAUG, the batch size was set to 30. Also, seven fusions are reported in Tables 2–5. We combined the results of the CNNs in an ensemble using the sum rule. The sum rule consists of averaging all the output probability vectors of the stand-alone CNNs in the ensemble to create a new probability vector that is used for classification. The rationale behind fusion, as Hansen [31] notes, is that "the collective decision produced by the ensemble is less likely to be in error than the decision made by any of the individual networks." The labels used in the tables and a brief description of the seven ensembles follow:

1. GoogleGoogle365: sum rule of GoogleNet and GoogleNetPlaces365 trained with each of the data augmentation protocols;
2. FusionLocal: sum rule of CNNs where each one is trained with a different data augmentation method;
3. FusionShort: sum rule of all CNNs trained with SGN, SSA, and SSpA;
4. FusionShortSuper: sum rule of all CNNs trained with SGN, SSA, SSpA, SSiA, and SuSA;
5. FusionSuper: sum rule of all CNNs trained with SGN, SSiA, SuSA, and TSM;
6. FusionSuperVGG16: sum rule of VGG16 trained with SGN, SSiA, SuSA, and TSM;
7. FusionALL: sum rule of all CNNs trained with SGN, SSA, SSpA, SSiA, SuSA, and TSM.

Table 2. Performance on the CAT (mean accuracy over ten-fold cross-validation). Values of interest are in bold.

CAT	NoAUG	SGN	SSA	SSpA	SSiA	SuSA	TSM
AlexNet	83.73	85.76	86.10	83.39	87.12	86.78	87.12
GoogleNet	82.98	86.10	87.80	83.39	86.78	85.08	87.80
VGG16	84.07	87.12	88.47	85.76	87.80	87.80	88.47
VGG19	83.05	85.42	87.80	84.75	86.10	86.10	89.15
ResNet50	79.32	81.36	85.42	76.95	85.08	82.03	87.12
ResNet101	80.34	84.75	85.42	75.59	82.03	73.56	86.78
Inception	79.66	82.71	—	66.44	—	84.07	86.10
GoogleNetPlaces365	85.15	86.44	85.76	83.73	86.10	86.10	88.47
VGG16BatchSize	—	86.10	88.14	86.78	89.49	86.10	**89.15**
FusionLocal	88.14	88.47	89.83	86.78	89.83	89.83	**90.51**
FusionShort				88.47			
FusionShortSuper				89.83			
FusionSuper				90.17			
FusionALL				89.83			
FusionSuperVGG16				89.83			

Table 3. Performance on the BIRDZ (mean accuracy over ten-split training/test set).

BIRDZ	NoAUG	SGN	SSA	SSpA	SSiA	SuSA	TSM
AlexNet	94.48	94.96	95.40	94.02	95.05	95.76	88.51
GoogleNet	92.41	94.66	94.84	91.48	93.85	95.85	82.91
VGG16	95.30	95.59	95.60	94.69	95.44	96.18	94.63
VGG19	95.19	95.77	87.15 *	94.50	95.44	96.04	94.88
ResNet50	90.02	94.02	93.22	90.48	92.95	94.16	91.75
ResNet101	89.64	94.00	92.76	88.36	92.84	94.20	90.62
Inception	87.23	93.84	92.48	83.81	92.30	94.01	90.52
GoogleNetPlaces365	92.94	94.81	95.10	92.43	94.76	95.80	86.91
VGG16BatchSize	—	95.84	—	94.91	95.81	**96.31**	94.78
FusionLocal	95.81	96.32	96.24	95.76	96.39	**96.89**	95.27
FusionShort				96.47			
FusionShortSuper				96.79			
FusionSuper				**96.90**			
FusionALL				96.89			
FusionSuperVGG16				96.78			

The * indicates a fold that failed to converge so producing a random performance in that fold. Values of interest are in bold.

Table 4. Performance on the ESC-50 (mean accuracy over the five-fold cross-validation). Values of interest are in bold.

ESC-50	NoAUG	SGN	SSA	SSpA	SSiA	SuSA	TSM
AlexNet	60.80	72.75	73.85	65.75	73.30	64.65	70.95
GoogleNet	60.00	72.30	73.70	67.85	73.20	71.70	73.55
VGG16	71.60	79.40	**80.90**	75.95	79.35	77.85	79.05
VGG19	71.30	78.95	78.80	74.10	78.00	76.40	77.45
ResNet50	62.90	76.65	75.95	70.65	77.20	73.95	77.40
ResNet101	59.10	75.25	75.65	70.05	77.50	72.30	74.85
Inception	51.10	71.60	74.70	63.45	75.55	71.10	70.65
GoogleNetPlaces365	63.60	75.15	76.10	71.35	74.00	71.60	73.55
VGG16BatchSize	—	79.40	80.50	73.45	79.35	77.85	80.00
FusionLocal	75.95	84.75	85.30	80.25	85.25	82.25	85.30
FusionShort				86.45			
FusionShortSuper				87.15			
FusionSuper				**87.55**			
FusionALL				87.30			
FusionSuperVGG16				85.75			

Table 5. Performance using different methods for representing the signal as an image.

	CAT			BIRD			ESC-50		
	GA	**MEL**	**CO**	**GA**	**MEL**	**CO**	**GA**	**MEL**	**CO**
AlexNet	82.03	83.73	79.32	91.85	91.43	87.54	73.95	73.50	65.50
GoogleNet	74.07	84.07	77.97	90.71	88.96	86.95	73.75	73.25	66.15
VGG16	83.39	86.10	80.00	92.65	93.17	88.82	77.60	**79.20**	66.75
VGG19	85.76	83.73	77.97	92.93	93.22	89.07	76.40	77.55	65.85
ResNet50	82.03	83.05	75.93	90.87	90.74	86.98	75.80	76.05	67.75
ResNet101	82.71	82.37	79.32	91.15	91.00	87.28	75.00	74.80	64.90
Inception	79.66	84.75	77.63	89.53	89.86	87.35	73.95	72.55	67.50
GoogleNetPlaces365	83.05	82.71	77.63	90.88	88.31	86.75	73.60	75.50	68.70
VGG16BatchSize	85.42	**87.80**	81.02	93.09	**93.22**	89.43	77.80	78.95	67.50
FusionLocal	87.46	**88.47**	82.37	93.76	**93.97**	90.57	81.90	**83.80**	73.25

VGG16 can fail to converge; when this happens, VGG16 undergoes a second training. VGG16 can also produce a numeric problem by assigning the same scores to all patterns. In this case, all scores are considered zeros. Other numeric problems in the fusions by sum rule can occur. To avoid such issues, all scores that do not produce not-a-number value are treated as zero.

In Tables 2–4, the DGT spectrogram is used for representing the signal as an image. Any cell with '—' means that the given CNN was not trained successfully (mainly due to memory problems with the graphical processing units (GPUs). We also tested three additional methods GA, MEL, and CO to represent a signal as an image, coupled with SGN only, as reported in Table 5.

In Table 6, our best ensembles FusionGlobal and FusionGlobal-CO are compared with the state of the art. FusionGlobal is built with the CNNs belonging to Fusion Super and those reported in Table 5. FusionGlobal-CO is constructed similarly to FusionGlobal but without considering the CNNs trained using CO as a signal representation approach. The performance reported in [11] in Table 5 is different from that reported in the original paper since, for a fair comparison with this work, we ran the method without considering the supervised data augmentation approaches.

Table 6. Comparison of our best sound classification ensemble with state of the art.

Descriptor	BIRDZ	CAT	ESC-50
[11]	96.45	89.15	85.85
FusionGlobal	96.82	90.51	88.65
FusionGlobal-CO	97.00	90.51	88.55
[2]	95.1	—	—
[34]	93.6	—	—
[33]	—	87.7	—
[28]	—	91.1	—
[28]—CNN	—	90.8	—
[51]	96.7	—	—
[20]	—	—	94.10
[54]	—	—	89.50
[21]	—	—	87.10
[19]	—	—	88.50
[27]	—	—	84.90
[16]	—	—	86.50
[17]	—	—	83.50
[52]	—	—	83.50
[18]	—	—	81.95
Human Accuracy [4]	—	—	81.30

The following conclusions can be drawn from the reported results:

1. There is no single data augmentation protocol that outperforms all the others across all the tests. TSM performs best on CAT and ESC-50 but works poorly on BIRDZ. Data augmentation at the spectrogram level works poorly on ESC-50 as well as on two other datasets. SGN and data augmentation at the signal level work well across all the datasets. On average, the best data augmentation approach is SSA. Although it produces a performance that is close to SSiA, the training time for SSA is shorter;
2. The best stand-alone CNNs are VGG16 and VGG19;
3. DGT works better than the other signal representations;
4. Combining different CNNs enhances performance across all the tested datasets;
5. For the ensemble FusionLocal, data augmentation is marginally beneficial on CAT and BIRDZ but produces excellent results on ESC-50. Compared to the stand-alone CNNs, data augmentation improves results on all three datasets. Of note, an ensemble of VGG16 (FusionSuperVGG16) outperforms the stand-alone VGG16;
6. The performance of the ensemble of CNNs trained with different augmentation policies (FusionALL) can be further improved by adding to the ensemble those networks trained using different signal representations (FusionGlobal). However, this performance improvement required considerable computation time, mainly during the training step;
7. The approach in [20] manages to outperform our results, but the authors pretrained their networks on AudioSet; hence the comparison with our approach is not fair.

The methods reported in Table 6 were based solely on deep learning approaches. As mentioned in the introduction, several papers have proposed sound classification methods based on texture features. It is also possible to construct ensembles that combine deep learning with texture methods. To examine the potential of combining ensembles trained on texture descriptors with deep learning approaches, the following two fusion rules were examined:

a. Sum rule between FusionGlobal and the ensemble of texture features proposed in [55] (extracted from DGT images) obtains an accuracy of 98.51% (higher than that obtained by FusionGlobal). In BIRDZ, the ensemble of texture features obtains an accuracy of 96.87%, which is close to that obtained by our deep learning approach. The ensemble with texture descriptors works poorly on ESC-50, producing an accuracy of only 70.6%. As a result, it is not advantageous to combine the texture approach with FusionGlobal;

b. The sum rule1 between FusionGlobal and [34] obtains an excellent accuracy of 98.96% compared to [34], which achieves an accuracy of 93.6%. This means that the features extracted in [34] and by the deep learning approach access different information.

In terms of computation time, the most expensive activity is the conversion of audio signals into spectrograms since the conversion runs on a CPU and not on a GPU. In Table 7, computation time is reported for different CNNs and signal representations on a machine equipped with an i7-7700HQ 2.80 GHz processor, 16 GB RAM, and a GTX 1080 GPU. This test was run on an audio file of length 1.27 s with a sample rate of 32 kHz. It is interesting to note that FusionGlobal takes less than three seconds using a laptop. However, a speed-up is possible: audio files can be classified simultaneously with DGT since it can be parallelized.

Table 7. Computation time (in seconds) between CNNs and signal representations.

Signal Representation	Computation Time	CNN	Computation Time
DGT	1.29	AlexNet	0.01
GA	0.02	GoogleNet	0.03
MEL	0.01	VGG16	0.01
CO	0.08	VGG19	0.01
		ResNet50	0.02
		ResNet101	0.03
		Inception	0.03

In Table 8, we compare some of the advantages and disadvantages of our system with audio approaches proposed in the literature whose backbone is a neural network trained on image-based audio features.

Table 8. Comparisons among the different approaches of audio classification.

Paper	Method	Advantages
Ours	Ensemble of networks with different data augmentation and different spectrograms.	Great performances over a variety of datasets and easy to implement but requires large computational power.
[20]	Multi-stage sequential learning with knowledge transfer from a large audio dataset.	State of the art on ESC-50. However, it requires large computational power, and the model is pretrained on a large audio dataset.
[54]	Pretrained model with multi-channel features.	Simple model with performances similar to ours on ESC-50.
[21]	CNN model with potential compression.	Performance close to the state of the art on ESC-50 and suitable for mobile devices.
[16]	CNN with filter banks learned using convolutional Restricted Boltzmann Machines + fusion with gammatone spectral coefficient and Mel energies.	Performance close to ours on ESC-50 but unsuitable for mobile devices.
[56]	CNN with mix-up and data augmentation.	Very light network for mobile devices that performs very well, but not as much as the state-of-the art models.

6. Conclusions

In this paper, we presented the largest study conducted so far that investigates ensembles of CNNs using different data augmentation techniques for audio classification. Several data augmentation approaches designed for audio signals were tested and compared with each other and with a baseline approach that did not include data augmentation. Data augmentation methods were applied to the raw audio signals and their visual representations using different spectrograms. CNNs were trained on different sets of data augmentation approaches and fused via the sum rule.

Experimental results clearly demonstrate that ensembles composed of fine-tuned CNNs with different architectures maximized performance on the tested three audio classification problems, with some of the ensembles obtaining results comparable with the systems, including on the ESC-50 dataset. To the best of our knowledge, this is the largest, most exhaustive study of CNN ensembles applied to the task of audio classification. Our best ensemble, FusionGlobal is composed of 63 networks with nine different architectures and seven different trainings for every network. Four of the networks were obtained using different data augmentation strategies and three different methods to create a spectrogram. It is worth noting that the same ensemble is competitive with the state of the art on the three datasets. To be more precise, we reach 96.82% on the BIRDZ dataset, 90.51% on the CAT dataset, and 88.65% on ESC-50.

This work can be expanded further by investigating which augmentation methods (spectrogram augmentation vs. signal augmentation) work best for classifying different kinds of sounds. We also plan to apply transfer learning using spectrograms instead of natural images. A systematic selection of augmentation approaches, e.g., by iteratively evaluating an increasing subset of augmentation techniques (as is typical when evaluating different features), would require an enormous amount of time and computation power. An expert-based approach that utilizes the knowledge of environmental scientists would be the best way of handling this challenge.

This study could also be expanded by including more datasets, which would provide a more comprehensive validation of the proposed fusions. Furthermore, there is a need to investigate the impact on performance when different CNN topologies and parameter settings in the retuning step are combined with different types of data augmentation.

The technique proposed here is not exclusively applicable to environmental sound classification but can be applied to many related audio tasks. For example, this technique should work well for real-time audio event detection and speech recognition. An important application area related to the ESC-50 dataset for noise identification would be to apply our approach to the more refined problem of fault diagnosis of drilling machines and other machinery [57,58]. Most studies in this area have focused on extracting handcrafted texture features from visual representations of sound [58]. But recently, there has been some interesting work that has successfully used deep learning to solve this problem [57]. Investigating the performance of our proposed method to tasks such as these would involve combining a different set of network architectures, but the main idea of ensembling different models obtained from many audio augmentation protocols should increase classification performance on these novel tasks and would be worth investigating in the future. Finally, we plan on exploring some possibilities presented in [59].

Author Contributions: Conceptualization, L.N. and G.M.; software, L.N. and G.M.; resources, S.B., M.P. and L.N.; writing—Original draft preparation, S.B., M.P., G.M. and L.N.; writing—Review and editing, S.B., G.M., L.N. and M.P.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: MATLAB source code for this project is located at https://github.com/LorisNanni.

Acknowledgments: The authors thank NVIDIA Corporation for supporting this work by donating Titan XP GPU and the Tampere Center for Scientific Computing for generous computational resources.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

List of the methodological acronyms along with a few others used in this study.

ACT	CNN Activation Layer
CNN	Convolutional Neural Networks
CO	Cochleagram
CONV	CNN Convolutional Layer
DGT	Discrete Gabor Transform
FC	CNN Fully-Connected Layer
GA	Gammatone
GLCM	Gray Level Co-occurrence Matrices
GPU	Graphical Processing Unit
HPSS	Harmonic-Percussive Source Separation
LBP	Local Binary Patterns
LPQ	Local Phase Quantization
Mel	Mel-frequency Cepstral Coefficients spectrograms
POOL	CNN Pooling Layer
SGN	Standard Signal Augmentation
SSA	Short Signal Augmentation
SSiA	Super Signal Augmentation
SSpA	Short Spectrogram Augmentation
STFT	Short-Time Fourier Transform
SuSA	Super Spectro Augmentation
SVM	Support Vector Machines
TSM	Time Scale Modification

References

1. Padmanabhan, J.; Premkumar, M.J.J. Machine learning in automatic speech recognition: A survey. *Iete Tech. Rev.* **2015**, *32*, 240–251. [CrossRef]
2. Nanni, L.; Costa, Y.M.; Lucio, D.R.; Silla, C.N., Jr.; Brahnam, S. Combining visual and acoustic features for audio classification tasks. *Pattern Recognit. Lett.* **2017**, *88*, 49–56. [CrossRef]
3. Sahoo, S.K.; Choubisa, T.; Prasanna, S.R.M. Multimodal biometric person authentication: A review. *IETE Tech. Rev.* **2012**, *29*, 54–75. [CrossRef]
4. Piczak, K.J. ESC: Dataset for environmental sound classification. In Proceedings of the 23rd ACM International Conference on Multimedia, Brisbane, Australia, 26–30 October 2015; Association for Computing Machinery: New York, NY, USA, 2015; pp. 1015–1018.
5. Lidy, T.; Rauber, A. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In Proceedings of the ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11–15 September 2005; pp. 34–41.
6. Wyse, L. Audio spectrogram representations for processing with convolutional neural networks. *arXiv* **2017**, arXiv:1706.09559.
7. Rubin, J.; Abreu, R.; Ganguli, A.; Nelaturi, S.; Matei, I.; Sricharan, K. Classifying heart sound recordings using deep convolutional neural networks and mel-frequency cepstral coefficient. In Proceedings of the Computing in Cardiology (CinC), Vancouver, BC, Canada, 11–14 September 2016; pp. 813–816.
8. Ojala, T.; Pietikainen, M.; Maeenpaa, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 971–987. [CrossRef]
9. Brahnam, S.; Jain, L.C.; Lumini, A.; Nanni, L. Introduction to local binary patterns—New variants and new applications. In *Local Binary Patterns—New Variants and New Applications*; Brahnam, S., Ed.; Springer: Berlin, Germany, 2014.
10. Ojansivu, V.; Heikkila, J. Blur insensitive texture classification using local phase quantization. In *International Conference on Image and Signal Processing*; Springer: Berlin, Germany, 2008; pp. 236–243.
11. Nanni, L.; Maguolo, G.; Paci, M. Data augmentation approaches for improving animal audio classification. *arXiv* **2019**, arXiv:1912.07756. [CrossRef]
12. Xie, J.; Hu, K.; Zhu, M.; Yu, J.; Zhu, Q. Investigation of different CNN-based models for improved bird sound classification. *IEEE Access* **2019**, *7*, 175353–175361. [CrossRef]
13. Harjoseputro, Y.; Yuda, I.; Danukusumo, K. MobileNets: Efficient convolutional neural network for identification of protected birds. *Int. J. Adv. Sci. Eng. Inf. Technol.* **2020**, *10*, 2290–2296. [CrossRef]
14. Zor, C.; Awais, M.; Kittler, J.; Bober, M.; Husain, S.; Kong, Q.; Kroos, C. Divergence based weighting for information channels in deep convolutional neural networks for bird audio detection. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 3052–3056.
15. Cao, Z.; Principe, J.C.; Ouyang, B.; Dalgleish, F.; Vuorenkoski, A. Marine animal classification using combined CNN and hand-designed image features. In Proceedings of the OCEANS 2015—MTS/IEEE Washington, Washington, DC, USA, 19–22 October 2015; pp. 1–6.
16. Sailor, H.B.; Agrawal, D.M.; Patil, H.A. Unsupervised filterbank learning using convolutional restricted boltzmann machine for environmental sound classification. *InterSpeech* **2017**, *8*, 9.
17. Li, X.; Chebiyyam, V.; Kirchhoff, K. Multi-stream network with temporal attention for environmental sound classification. *arXiv* **2019**, arXiv:1901.08608.
18. Agrawal, D.M.; Sailor, H.B.; Soni, M.H.; Patil, H.A. Novel TEO-based Gammatone features for environmental sound classification. In Proceedings of the 25th European Signal Processing Conference (EUSIPCO 2017), Kos Island, Greece, 28 August–2 September 2017; pp. 809–1813.
19. Sharma, J.; Granmo, O.-C.; Olsen, M.G. Environment sound classification using multiple feature channels and deep convolutional neural networks. *arXiv* **2019**, arXiv:1908.11219.
20. Kumar, A.; Ithapu, V.K. A sequential self teaching approach for improving generalization in sound event recognition. *arXiv* **2020**, arXiv:2007.00144.
21. Mohaimenuzzaman, M.; Bergmeir, C.; West, I.T.; Meyer, B. Environmental sound classification on the edge: Deep acoustic networks for extremely resource-constrained devices. *arXiv* **2021**, arXiv:2103.03483.
22. Marcus, G. Deep learning: A critical appraisal. *arXiv* **2018**, arXiv:1801.00631.
23. Lasseck, M. Audio-based bird species identification with deep convolutional neural networks. In *CLEF*; Museum für Naturkunde: Berlin, Germany, 2018.
24. Sprengel, E.; Jaggi, M.; Kilcher, Y.; Hofmann, T. Audio based bird species identification using deep learning techniques. In *CLEF*; Museum für Naturkunde: Berlin, Germany, 2016.
25. Wei, S.; Xu, K.; Wang, D.; Liao, F.; Wang, H.; Kong, Q. Sample mixed-based data augmentation for domestic audio tagging. *arXiv* **2018**, arXiv:1808.03883.
26. Inoue, T.; Vinayavekhin, P.; Wang, S.; Wood, D.; Greco, N.; Tachibana, R. Domestic activities classification based on CNN using shuffling and mixing data augmentation, Technical Report. *DCASE 2018 Challenge*. 2018. Available online: https://www.semanticscholar.org/paper/DOMESTIC-ACTIVITIES-CLASSIFICATION-BASED-ON-CNN-AND-Inoue-Vinayavekhin/90f875233e3efebe02feeb10cb551cc69f20ebc7 (accessed on 6 January 2021).

27. Tokozume, Y.; Ushiku, Y.; Harada, T. Learning from between-class examples for deep sound recognition. *arXiv* **2018**, arXiv:1711.10282.
28. Pandeya, Y.R.; Kim, D.; Lee, J. Domestic cat sound classification using learned features from deep neural nets. *Appl. Sci.* **2018**, *8*, 1949. [CrossRef]
29. Jaitly, N.; Hinton, E.S. Vocal tract length perturbation (VTLP) improves speech recognition. In Proceedings of the International Conference on Machine Learning (ICML Work), Atlanta, GA, USA, 16–21 June 2013.
30. Takahashi, N.; Gygli, M.; Pfister, B.; Van Gool, L. Deep convolutional neural networks and data augmentation for acoustic event recognition. *arXiv* **2016**, arXiv:1604.07160.
31. Hansen, L.K.; Salamon, P. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, *12*, 993–1001. [CrossRef]
32. Nanni, L.; Brahnam, S.; Maguolo, G. Data augmentation for building an ensemble of convolutional neural networks. In *Smart Innovation Systems and Technologies*; Chen, Y.-W., Ed.; Springer Nature: Singapore, 2019; pp. 61–70.
33. Pandeya, Y.R.; Lee, J. Domestic cat sound classification using transfer learning. *Int. J. Fuzzy Log. Intell. Syst.* **2018**, *18*, 154–160. [CrossRef]
34. Zhao, Z.; Zhang, S.H.; Xu, Z.Y.; Bellisario, K.; Dai, N.H.; Omrani, H.; Pijanowski, B.C. Automated bird acoustic event detection and robust species classification. *Ecol. Inform.* **2017**, *39*, 99–108. [CrossRef]
35. Prusa, Z.; Søndergaard, P.L.; Balázs, P. The large time frequency analysis toolbox: Wavelets. In *International Symposium on Computer Music Multidisciplinary Research*; Springer: Cham, Switzerland, 2013.
36. Rabiner, L.R.; Schafer, R.W. *Theory and Applications of Digital Speech Processing*; Prentice Hall Press: Hoboken, NJ, USA, 2010.
37. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proceeding IEEE* **1998**, *86*, 2278–2323. [CrossRef]
38. Lyon, R.F.; Dyer, L.M. Experiments with a computational model of the cochlea. ICASSP '86. *IEEE Int. Conf. Acoust. Speech Signal Process.* **1986**, *11*, 1975–1978.
39. Russakovsky, O.; Deng, J.; Su, H. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
40. Zhou, B.; Khosla, A.; Lapedriza, A.; Torralba, A.; Oliva, A. Places: An image database for deep scene understanding. *arXiv* **2017**, arXiv:1610.02055. [CrossRef]
41. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; Pereira, F., Ed.; Curran Associates Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.
42. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
43. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arxiv* **2014**, arXiv:1409.1556v6.
44. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
45. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
46. Salamon, J.; Bello, J. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283. [CrossRef]
47. Driedger, J.; Müller, M. TSM Toolbox: MATLAB implementations of time-scale modification algorithms. In Proceedings of the 17th International Conference on Digital Audio Effects (DAFx-14), Erlangen, Germany, 1–5 September 2014.
48. Driedger, J.; Müller, M.; Ewert, S. Improving time-scale modification of music signals using harmonic-percussive separation. *IEEE Signal Process. Lett.* **2014**, *21*, 105–109. [CrossRef]
49. Laroche, J.; Dolson, M. Improved phase vocoder time-scale modification of audio. *IEEE Trans. Speech Audio Process.* **1999**, *7*, 323–332. [CrossRef]
50. Bookstein, F.L. Thin-plate splines and decomposition of deformation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1989**, *11*, 567–585. [CrossRef]
51. Zhang, S.-H.; Zhao, Z.; Xu, Z.Y.; Bellisario, K.; Pijanowski, B.C. Automatic bird vocalization identification based on fusion of spectral pattern and texture features. In Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 271–275.
52. Kumar, A.; Khadkevich, M.; Fügen, C. Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes. In Proceedings of the 2018 IEEE International Conference on Acoustics Speech and Signal Processing (IEEE ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 326–330.
53. Nanni, L.; Costa Costa, Y.M.; Alessandra, L.; Kim, M.Y.; Baek, S.R. Combining visual and acoustic features for music genre classification. *Expert Syst. Appl.* **2016**, *45*, 108–117. [CrossRef]
54. Kim, J. Urban sound tagging using multi-channel audio feature with convolutional neural networks. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020, Tokyo, Japan, 2–3 November 2020.
55. Nanni, L.; Costa, Y.M.; Aguiar, R.L.; Silla, C.N., Jr.; Brahnam, S. Ensemble of deep learning visual and acoustic features for music genre classification. *J. New Music. Res.* **2018**, *47*, 383–397. [CrossRef]
56. Huang, J.; Leanos, J.J.A. AclNet: Efficient End-To-End Audio Classification CNN. *arXiv* **2018**, arXiv:1811.06669.

57. Glowacz, A.; Tadeusiewicz, R.; Legutko, S.; Caesarendra, W.; Irfan, M.; Liu, H.; Brumercik, F.; Gutten, M.; Sulowicz, M.; Daviu, J.A.A.; et al. Fault diagnosis of angle grinders and electric impact drills using acoustic signals. *Appl. Acoust.* **2021**, *179*, 108070. [CrossRef]
58. Tran, T.; Lundgren, J. Drill Fault Diagnosis Based on the Scalogram and Mel Spectrogram of Sound Signals Using Artificial Intelligence. *IEEE Access* **2020**, *8*, 203655–203666. [CrossRef]
59. Kwon, H.; Yoon, H.; Park, K.-W. POSTER: Detecting Audio Adversarial Example through Audio Modification. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, Communications Security, London, UK, 11–15 November 2019.