applied sciences

MDPI

Article

Analyzing the Influence of Diverse Background Noises on Voice Transmission: A Deep Learning Approach to Noise Suppression

Alberto Nogales *,†, Javier Caracuel-Cayuela and Álvaro J. García-Tejedor *,†

CEIEC, Universidad Francisco de Vitoria, Ctra. Pozuelo-Majadahonda km. 1800, 28223 Madrid, Spain; javcaracuel1@gmail.com

* Correspondence: alberto.nogales@ceiec.es (A.N.); a.gtejedor@ceiec.es (Á.J.G.-T.)

† These authors contributed equally to this work.

Featured Application: A deep learning application to improve speech clarity in digital audio affected by environmental noises, showing potential for enhancing real-time streaming communication in noisy settings.

Abstract: This paper presents an approach to enhancing the clarity and intelligibility of speech in digital communications compromised by various background noises. Utilizing deep learning techniques, specifically a Variational Autoencoder (VAE) with 2D convolutional filters, we aim to suppress background noise in audio signals. Our method focuses on four simulated environmental noise scenarios: storms, wind, traffic, and aircraft. The training dataset has been obtained from public sources (TED-LIUM 3 dataset, which includes audio recordings from the popular TED-TALK series) combined with these background noises. The audio signals were transformed into 2D power spectrograms, upon which our VAE model was trained to filter out the noise and reconstruct clean audio. Our results demonstrate that the model outperforms existing state-of-the-art solutions in noise suppression. Although differences in noise types were observed, it was challenging to definitively conclude which background noise most adversely affects speech quality. The results have been assessed with objective (mathematical metrics) and subjective (listening to a set of audios by humans) methods. Notably, wind noise showed the smallest deviation between the noisy and cleaned audio, perceived subjectively as the most improved scenario. Future work should involve refining the phase calculation of the cleaned audio and creating a more balanced dataset to minimize differences in audio quality across scenarios. Additionally, practical applications of the model in real-time streaming audio are envisaged. This research contributes significantly to the field of audio signal processing by offering a deep learning solution tailored to various noise conditions, enhancing digital communication quality.

Keywords: speech enhancement; noise suppression; deep learning; variational autoencoders

Citation: Nogales, A.; Caracuel-Cayuela, J.; García-Tejedor, Á.J. Analyzing the Influence of Diverse Background Noises on Voice Transmission: A Deep Learning Approach to Noise Suppression. *Appl. Sci.* **2024**, *14*, 740. https://doi.org/10.3390/app14020740

Academic Editors: Isabel Barbancho and Lorenzo J. Tardón

Received: 27 November 2023
Revised: 9 January 2024
Accepted: 11 January 2024
Published: 15 January 2024

1. Introduction

Although the quality used to store audio allows them to be very faithful to the originals, some information may be lost during transmission. In particular, we call noise a signal or set of signals that distort the wave that transmits the original sound. Within this concept, there can be artificial noises (those generated by the means of communication, such as interferences) or natural noises (those generated by the environment where the communication takes place). When the transmission is in noisy scenarios, ambient sounds may affect the intelligibility of the received message. This issue underscores the importance of addressing noise problems in audio, as highlighted in [1,2], demonstrating the significant impact of noise on the fidelity of audio transmissions and the need for solutions to enhance the overall quality of communication.

The motivation for this research arises from the critical need to enhance the clarity and intelligibility of speech in various communication settings, where background noise often compromises the quality of the transmitted audio. While existing noise reduction techniques have made strides in mitigating this issue, our work aims to develop a deep learning model specifically tailored to suppress background noise across a range of simulated scenarios. However, our objectives extend beyond mere noise elimination. We also seek to quantitatively assess the relative difficulty of filtering out different types of background noises present in the same audio fragment. By doing so, we aim to identify which types of noise have a more detrimental impact on speech quality, thereby creating a guide for future research and technological development in the field of audio signal processing.

The real problem comes when the noise is equal to or stronger than the signal and causes its complete distortion. This fact opens the possibility of using techniques that can eliminate background noise for safer and more reliable transmissions, which is desirable in cases such as phone communications, especially in emergencies.

Artificial intelligence has demonstrated its ability to remove noisy information from various formats, such as images or signals [3,4]. Deep learning models have obtained the best results in recent years among all the artificial intelligence techniques. Deep learning was defined by [5] as models composed of multiple processing layers that learn representations of data with various levels of abstraction. These models have shown an exemplary performance in speech enhancement [6].

Audio signals can be analyzed either in the time domain or the frequency domain, with the latter often represented visually as images. Our approach capitalizes on the image-processing capabilities of convolutional layers in deep learning neural networks, specifically for tasks like cleaning and restoration. We have trained a deep learning model that can remove four types of background noise with pairs of original audio and audio mixed with background noise. Original audio signals have been transformed into a 2D representation by converting them into a power spectrogram. Following this transformation, we employ a two-dimensional Variational Autoencoder (VAE) to remove noise from the signal and reconstruct the clean audio. As no dataset solves the presented use cases, we have created our own. To carry out the model development and training, we have used a dataset with recordings of TED talks (TED-LIUM 3 dataset) as the expected output and the same dataset mixed with four different background noises (aircrafts, rain and thunderstorms, wind, and traffic) obtained from different specialized websites as the input to the network.

The results show that our model performs better than other solutions in the state of the art. Regarding the proposed scenarios, although there are differences between them, we cannot assure that one of the background noises influences more than others. The only thing confirmed by the results is that the audios with wind are perceived with better improvement, but this perception is a bit tricky as the differences between the clean audios and the audios with this background noise are smaller.

The paper is structured as follows. Section 2 compiles some works framed in speech enhancements, audio denoising, etc. Section 3 formally describes the dataset used to train the model and the deep learning models used in the research. Section 4 compiles the different results that have been obtained and their interpretation. Finally, Section 5 gives some conclusions and future works.

2. State of the Art

Building on the foundational need to improve speech communication in noisy environments, as outlined in our introduction, a variety of models have been developed to address challenges in speech enhancement and background noise reduction. This section aims to review these related works, highlighting their contributions and limitations, to contextualize our own approach, which extends beyond mere noise removal to a nuanced understanding of how different types of noise uniquely impact speech quality.

Some of these works apply techniques not encompassed in the deep learning field. Ref. [7] reduces the presence of background music over conversational audios. It uses

trigonometric transformation and wavelet denoising techniques. In [8], the audio denoising is made with speech audios and noises like buzzing equipment or background noise from the street. Spectrograms form the training dataset and use a block thresholding estimation procedure. The same authors present, in [9], a similar approach to suppressing the harmonic noise of music by using its spectrograms. The method applies block attenuation techniques. Finally, ref. [10] implements a process to denoise speech audios containing slight background noises (which do not work with loud ones). It works with raw audio and applies Wiener filters, a well-known method of the previous age of audio denoising.

The present work focuses on speech enhancement but uses deep learning techniques. The following studies make use of these methods in similar use cases. In [11], an autoencoder with a bottleneck that uses Recurrent Neural Networks (RNNs) addresses speech enhancement in recordings made with mobile phones. In [12], a U-Net model is applied directly to the audio waveform to restore audios in cases of background noise and loss of signal. The results are evaluated with a loss metric and by people listening to the audios. The use of Generative Adversarial Networks (GANs) for speech enhancement can be found in [13]. In this case, they used raw audios of 10 use cases (eight of them are genuine cases and two of them artificially created). Another interesting work is [14], which separates the different waves of raw audios with the voices of women and men mixed with songs. In this case, deep autoencoders are used to achieve the issue. To achieve a similar objective, ref. [15] addresses the field of deep learning-based speech enhancement techniques, focusing on their real-time applications. Evaluating three popular models in terms of signal processing metrics, such as a signal-to-interference ratio, response time, and memory usage, the research offers valuable insights into the online viability of these methods.

Another distinguishing feature is the use of power log spectrograms to train the deep learning model, as in [16], that evaluate twelve deep learning models for single-channel ego-noise reduction on drones, exploring various domains and architectures. The findings highlight the superiority of U-Net models in the time–frequency complex domain, achieving substantial improvements in speech enhancement measures across low Signal-to-Noise ratio scenarios. In [17], a based convolutional system is developed to enhance spectral information by simultaneously utilizing multiple bandwidth spectrograms, specifically augmenting wider bandwidth (16 ms and 8 ms) spectrograms as auxiliary information. Experimental results on the VB dataset demonstrate that incorporating different bandwidth spectrograms provides supplementary information, resulting in an over 0.1 improvement, with the embedding dimension influencing the fusion strategy in the encoder. In [18], a Long Short-Term memory neural network (LSTM) is trained with frequency spectrograms to improve the speech quality in audios with background noises, reverberations, and a poor communication environment. Another work that uses power spectrograms is [19], which represents cases of passing cars and café babble noise. It uses a convolutional neural network in the approach. Similarly, in [20], the authors use voice sound with background noise characteristics of conversations represented as spectrograms. Then, a deep neural network of four layers is used to eliminate the noise in the spectrograms. Finally, ref. [21] presents a method for enhancing laser-detected speech signals by optimizing spectrograms to mitigate a low signal-to-noise ratio and non-stationary noise challenges. The approach incorporates GAN with spatial attention and integrates short-time objective intelligibility (STOI) into the loss function, successfully improving speech quality under severe noise interference conditions.

This research stands out in the field due to several key innovations. Primarily, it targets evaluating the impact of four distinct real-world background noises on speech audios. To achieve this, we have developed a unique deep learning model designed to effectively eliminate these noises from a series of speech recordings. Our choice of a variational autoencoder for this purpose distinguishes our approach. Moreover, our study is the first of its kind to incorporate a subjective evaluation method. This involves a panel assessing the clarity of speech audios post noise removal. While other studies rely solely on mathematical metrics, our approach adds a crucial human dimension to the assessment.

This step is often overlooked in other research due to the technical challenges and potential information loss when converting spectrograms back into raw audio, possibly leading to subpar results that others might choose not to report. Our willingness to undertake this complex transformation and evaluation process further underlines the novelty and depth of our work.

3. Materials and Methods

In this paper, we have trained a deep learning model, specifically a Variational Autoencoder, with pairs of power logs spectrograms of voice audios mixed with four different background noises: aircrafts, rain and thunderstorms, wind, and traffic jams. The research followed a four-step process that is described in the following. First, the creation of the dataset by mixing speech audios with background noise. Second, the transformation of the audios into a visual representation, a spectrogram, that is more efficient to manage as the model. Third, the training of the model with the dataset. Finally, the objective and subjective evaluations. The workflow followed from working with the raw data to obtaining the trained model is represented in Figure 1.

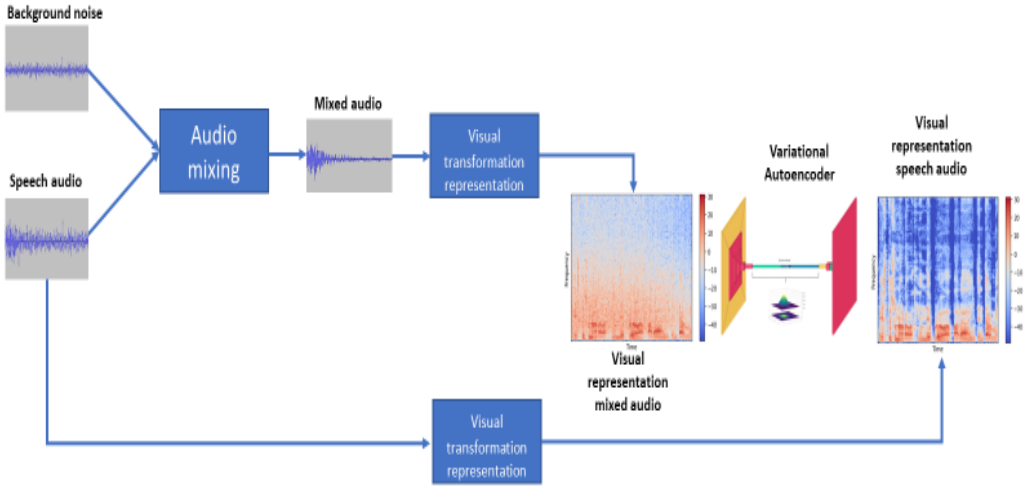

Figure 1. Workflow for training a model for background noise suppression.

3.1. Datasets of Speech Audios and Background Noises

3.1.1. Audio Features Decision

The first step consists of selecting the audio file format and the signal processing characteristics: sampling rate, bit-depth, and the number of channels. As a file format, we use WAV or WAVE (apocope of Waveform), launched in 1991 by Microsoft based on the RIFF (Resource Interchange File Format) specification. It has the advantage of being in an uncompressed format, in which the user hears what is stored. It also supports different quality characteristics such as sampling frequency and bit depth [22], which are the most important features of audio management [23]. We have decided to use 22.5 kHz, a bit-depth of 16 bits, and only one audio channel corresponding to monaural sound reproduction. These characteristics are sufficient for the problem we want to solve.

During the second step, we need to create the pairs of audios. As no dataset solves the presented use cases, we have created our own. For that, we need speech audios and background noises that represent the four use cases.

For the former, we are using the TED-LIUM 3 dataset, which includes audio recordings of the well-known TED-TALK series. The dataset contains 2351 audio files in a NIST Sphere format (SPH), which is very common in audio speech files as it includes the audio alongside a transcription of the speech. It uses a sampling rate of 16 kHz and a bit-depth of 16 bits. The total length of the dataset reaches more than 452 h. In the audios, we can find a diverse range of people with different types of tones and quality recordings under other

circumstances, which augments the problem's difficulty. The dataset is freely available to download from Open Speech and Language resources.

For the latter, we have obtained the different background noises from two websites. They used a WAV format, a sampling rate of 16 kHz, and a bit-depth of 16. The number of audios per use case differs, as the availability was not the same. This problem will be, lately, solved during the preprocessing stage. Also, the length of the audio varies depending on the use case. Table 1 summarizes this information for the four use cases.

Table 1. Background noise use cases dataset.

Use Case	Number of Audios	Length per Audio
Aircrafts	80	Around 5 s
Storm lights and rain	18	Around 5 min
Wind	522	From 3 s to 5 min
Traffic jams	512	Around 3 s

3.1.2. Audio Preprocessing

The first problem arose with the TED LIUM dataset as it has an SPH format. We transformed it into WAV format using a Python script using the SoX tool [24] to convert the audio.

Next, we will describe all the modifications done to obtain the final dataset of the pairs of audios. First, we eliminated the first and last 15 s of the speech audios to avoid moments without speech and irrelevant sounds like applause or music. Then, we generated 60,000 chunks with a length of 3 s with a balance of 25% in-stances for each use case. At this stage, we also set the value of the sampling rate, bit-depth, and the number of channels, which were set at 22.5 kHz, 16 bits, and an audio-mono channel, respectively. In the case of finding audio with different value ranges, the corresponding transformation applied min–max normalization. This method allows all audio values to be in the same range between 0.0 and 1.0, as shown in Equation (1).

$$z = \frac{x - minval}{maxval - minval} \quad (1)$$

Given the complexity of the problem, the neural model requires many audios for proper training. In our case, the availability of audios was scarce, so it was necessary to apply data augmentation techniques and create synthetic data. Once we had the audios chunked in pieces of 3 s with standardized features, we mixed the speech audios with the different background noises randomly, but ensured the number of audios for the four use cases. In the cases where the background noise did not have a length of 3 s, different audio of the same case was chosen until the speech audio length was reached.

3.1.3. Visual Representation of Audios

Audio signals can be processed as a time-domain or a frequency-domain representation. The former uses the signal as raw audio. The latter uses images of visual representations of the audio. Due to the high complexity of the problem, where the original speech audios and its version with background noise do not match any of the values (increasing the difficulty of a reconstruction problem), we have decided to use the second option as images perform well in auto-encoders. We used autoencoders with convolutional filters of 2 dimensions which have demonstrated better performances in these models than those of 1 dimension [25].

Many audio representations in the frequency domain are log power spectrograms (LPS), Mel spectrograms, or Mel-frequency cepstral coefficients spectrograms (MFCCS). In this case, we used LPS because it includes an audio feature called power representing wave decibels at a particular moment. LPS consists of an image representing the audio information. In this case, it captures more information than other standard spectrums [26].

Spectrograms are created using a Python script and a library called librosa, [27], which specializes in music and audio management. The script implements a Short-Term Fourier Transformation (STFT) that converts the audio from the time domain to the frequency domain. Each spectrogram was created using a sliding window of 23 widths and 11.6 steps in milliseconds. The output spectrograms had a size of 256 × 256 × 1 for each instance. Finally, the dataset contained 60,000 spectrograms, with 15,000 belonging to each use case.

3.2. Using a Variational Autoencoder for Noise Suppression

As noted in the state-of-the-art section, autoencoders perform well to do speech enhancement, audio denoising, etc. We have selected a variational autoencoder as the DL model to perform this task because of its ability to represent the input data more accurately using a latent space. With this representation, the output images have higher definition and, therefore, the quality when transforming them backwards into raw audio will be better. In the following, we formalize some deep learning concepts to better understand this type of architecture.

3.2.1. Convolutional Operator

This operator makes it possible to find image features like edges, which is why their main application is image classification [28]. In particular, it led to the creation of the 2-dimensional Convolutional Neural Networks (2D CNNs), used in Deep Learning since 1999 [29]. Convolutional operators allow finding a feature in one part of an image that can later be found in another. As we work with images of power log spectrograms, the convolutional operators of the Variational Autoencoder will find feature representations from one spectrogram to another.

3.2.2. Autoencoders

These two-part models first use a multilayer encoder network to represent high-dimensional structures in a low-dimensional space. Then, there is a decoder network to convert the data from this space into high-dimensional structures with some relation to the first one [30]. This architecture works as follows. The input data go through the different convolutional layers of the encoder, obtaining a small piece of data with the main features. These data are distributed in the bottleneck, creating a representation called latent space. Finally, the feature representation goes through the decoder to obtain an output like the input data.

3.2.3. Variational Autoencoders

This Autoencoder solves the problem that classical ones have with latent space. Instead of placing a single point in the latent space, this case provides a distribution. This latent space can also be better organized by adding a regularization to the loss function, [31]. In our case, it creates a distribution using the input data's mean and variance. The following equation represents the distribution of this space.

$$f(x_1, \ldots, x_k) = \frac{e^{-\frac{1}{2}(\vec{x}-\vec{\mu})^T \sum^{-1} (\vec{x}-\vec{\mu})}}{\sqrt{(2\pi)^k |\sum|}} \tag{2}$$

where $\vec{\mu}$ represents the mean vector of the different distributions in the latent space and $|\sum|$ is the covariance matrix of the distributions. The last two parameters are calculated using Equations (3) and (4) applied to a space of 2 dimensions.

$$\vec{\mu} = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix} \tag{3}$$

$$\begin{pmatrix} {\sigma_x}^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & {\sigma_y}^2 \end{pmatrix} \tag{4}$$

To sample a point into the latent space, we have used the formula of sampling points, depicted in Equation (5). In this case, we have modified it to get negative values, so we have a more expansive dimensional space to represent the information where ε is a sampled point from a standard normal distribution.

$$z = \vec{\mu} + \frac{e^{\log(\sigma^2)}}{2}\varepsilon \tag{5}$$

3.3. Training Phase

To train the model, we have split the dataset into a percentage of 80 for training/validation and 20 for the test. The final dataset has an amount of 60,000 LPS that represent audios of 3 s without overlapping. The model was trained with pairs of audios with mixed background noise and the original version of these audios.

The hyperparameters during training were found using a grid search strategy. This method guides the training in finding the best hyperparameter setting for the model by using different combinations of the values [32]. The different hyperparameters and values used in the grid search have been compiled in Table 2.

Table 2. Hyperparameters and values used in the grid search.

Hyperparameter	Values
Latent space	128, 200, 400, 800, 1024, and 2048 neurons
Dense layer	100, 200, 256, 1024, 2048, and 4096 neurons
Convolutional blocks	3, 4, 5, and 6
Skip connections	3, 4, and 5
Learning rate	0.01, 0.001, and 0.0001

Optimizer and loss function hyperparameters have not been changed during the process. The optimizer is an adaptive learning rate optimization algorithm. In particular, we have used Adam as it was explicitly designed for training deep neural networks, [33]. All the efforts to create a variational bottleneck do not make sense unless the network knows how to learn about the input data representations in the latent space. Therefore, the loss function needs a change to distribute the information in the latent space precisely and accurately. The loss function uses the Root Mean Squared Error (RMSE) and the Kullback–Leibler divergence $D_{KL}(P||Q)$ (KL). This last metric will allow the network to check if the distributions are placed correctly in the latent space. RMSE and KL have been formalized in Equations (6) and (7).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{6}$$

$$D_{KL}(N(\mu_i, \sigma_i)||N(0,1)) = \frac{1}{2}\sum_i \left(1 + \ln\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right) \tag{7}$$

where N(μ,σ) is a normal distribution having μ as the mean, σ as the standard deviation of the output data obtained in training, and N(0,1) is the standard normal distribution.

Finally, Equation (8) shows the modified loss function we have used.

$$loss = RMSE + \lambda \cdot D_{KL}(N(\mu_i, \sigma_i)||N(0,1)) \tag{8}$$

In this case, λ is a hyperparameter that weights the KL metric. After fine-tuning, the selected value for this hyperparameter was 100,000, indicating that the RMSE between the input and output signals was not a primary concern in our optimization process. This decision was based on the understanding that RMSE is a relatively poor metric for evaluating the noise level in an audio signal, especially when the audio is represented as an

image in the form of a power spectrogram. In such a representation, even a minor deviation in a single pixel can significantly impact the RMSE, but this does not necessarily translate to a perceptible error when the image is converted back to an audio signal. Therefore, our model's loss function is designed to prioritize other aspects over RMSE for a more accurate and meaningful evaluation of audio quality.

The model comprises 306,762,465 hyperparameters, and we used an AMD Threadripper 2950X with 128 Gigabytes of DDR4 RAM and a GeForce RTX 2080 TI GPU. The training lasted more than 22 h. The final metric at the end of the training phase was 0.0153.

3.4. Proposed Method

The proposed solution is a two-dimensional VAE that is trained with pairs of spectrograms of speech audios with real background noise and the original speech audios. After some convolutional operation, the encoder reduces the size to a minimal piece of information that represents its main features. These pieces of information are then placed in a latent space, which is a distribution of the encoded data obtained at the convolutional stage. The mean and variance of each input are used to fit the distribution in the latent space, this feature being the main difference from a standard encoder. For a VAE, each input represents a distribution in latent space rather than a single point. Information is recovered and upsampled from the latent space until the output has the expected dimensions. Finally, this output is compared to the spectral representation of the original audio without background. This comparison evaluates how well the background noise has been suppressed and lets the model adjust its hyperparameters and learn the information. After training with the whole dataset that comprises audios from the four use cases, the model can remove the real background noise.

The input layer of the model has a size of $256 \times 256 \times 1$, representing the audio of one channel, which is connected to the following layer. The input layer aims to feed the audio to the convolutional blocks. These blocks are responsible for reducing the dimensionality of the information. The convolutional stage consists of 6 convolutional blocks, where each of them obtains a feature map of the previous data. Each block has two convolutional layers with two 2-dimensional filters of size 3×3, stride one, and max-pooling of 2×2 (except in the last block). Convolutional layers in the same block have the same number of neurons using the Rectified Liner Unit (ReLU) as an activation function. The number of neurons from one block to another is the following: 32, 64, 128, 256, 512, and 1024. At the end of this convolutional stage, the input data are reduced to $8 \times 8 \times 1024$. This small piece of information represents the input data with all the main features. Finally, a flattened layer is applied to process information as a one-dimensional vector of size, 65,536.

At this point, the information goes through the bottleneck, which in the case of VAEs builds a latent space of its representation. The bottleneck entrance has a dense layer of 2048 neurons that introduces the information from the last convolutional block. Then, we have two other dense layers that manage the distribution's mean and variance, having 2048 neurons. A lambda layer is applied to choose the point representing the input data in the latent space. After getting this point, we can position it in a multivariate space and recreate the last convolution dimensions before the bottleneck, allowing the decoder to do the inverse process. To start this process, we use a dense layer that converts the point into information that can be processed in the de-convolutional stage.

The stage of dimensionality upsampling, or deconvolution, now begins. This stage starts with a reshaped layer so the information can be introduced in the two dimensions' deconvolutional blocks. This process has five deconvolution blocks. Each block comprises two convolution layers, with an upsampling layer in the first position. Again, the number of neurons within the layers of the same block is the same, and all use the ReLU activation function. The number of neurons per layer of each block is 512, 256, 128, 64, and 32. These blocks increase the dimensionality of the feature map. In this deconvolution process, starting from the tiny feature map in the bottleneck stage, new audio spectrograms are created using only the essence of the original audio. In each convolution block, concatenate

layers have been used with the even blocks of the encoder to speed up the training and not lose the substantial relationship between the data. Lastly, a final convolution layer with one neuron is used to achieve the same final audio size as the input. The last layer uses the hyperbolic tangent (tanh) activation. A representation of the model is in Figure 2.

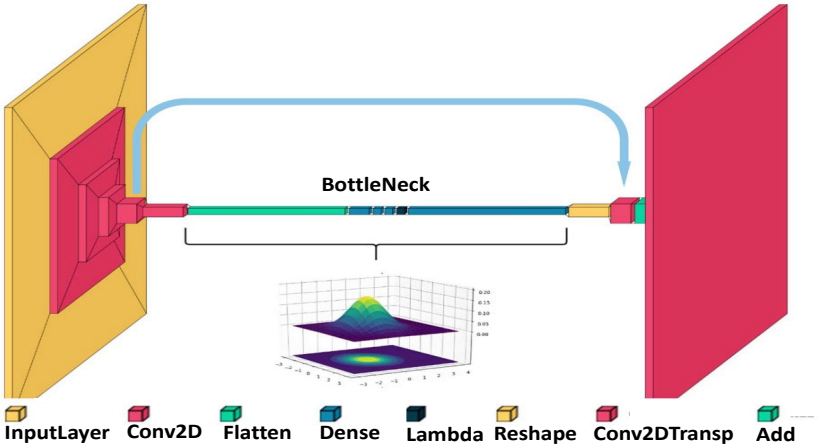

Figure 2. Architecture of the model.

Finally, the output data are compared to the spectral representation of the original audio without background noise. This comparison evaluates how well the background noise has been suppressed and lets the model adjust its hyperparameters and learn the information. After feeding the whole dataset that comprises audios from the four use cases, the model is trained so it can remove real background noise.

4. Results and Evaluation

This section compiles the results obtained by evaluating the model for the four different use cases with objective and subjective tests. The first ones compare our model with others in the state-of-the-art using mathematical metrics. The second type of tests is based on the subjective perception of a group of people who will evaluate how good the model is at eliminating noise.

4.1. Objective Evaluations

4.1.1. Evaluation of Noise Reduction Performance

We will assess the performance of our approach in eliminating background noise by comparing it with two baseline methods. This evaluation involves comparing the noisy audio input to the model with the model's output, which is a denoised signal. The two baseline models are a classical method using Wiener filters and a more recent technique known as Deep Audio Priors Design (DAP). Wiener filters were used by [34] to reduce the noise in audios utilizing the frequency domain. DAP Design is an update of U-Net that uses dilated convolutions and dense connections [35].

The evaluation is made with a set of audios that comprises 400 instances, 100 for each use case. Each audio has a length of 3 s, with a sample rate of 22 kHz and a bid-depth of 16.

To make an accurate comparison, we use two metrics: MSE and a Signal-to-Noise Ratio (NSR). The first one measures the Euclidean distance between two images which are the spectrograms of an audio with background noise and the same audio after being processed by the denoising model, [36]. The larger the value of MSE, the greater the difference between the noise signal and the cleaned signal, from which it can be inferred that the model eliminates background noise better. The metric is depicted in Equation (9) where y_i is the value of a particular position in the noisy audio and $\hat{y}_i$ is the value of the denoising audio in the same place.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{9}$$

The second metric measures the difference in decibels (dB) between the same two signals once they have been converted back to the audio domain. SNR values range between −35 and 35 dB, which is the theoretical magnitude of noise. The closer the value is to zero (either positive or negative), the better the performance of the model [37]. Equation (10) mathematically formalizes this metric. In the Equation, P_{signal_noise} refers to the audio with background noise and P_{noise} only to the background noise. By subtracting the noise from the noise with the audio, we obtain the audio or signal, allowing us to calculate the signal-to-noise ratio.

$$SNR_{dB} = 10\log_{10}\left(\frac{P_{signal\_noise} - P_{noise}}{P_{noise}}\right) \tag{10}$$

In Table 3, we compile both metrics and compare our model with the two proposed baselines.

Table 3. Comparison against baselines.

Use Case		MSE	SNR
Aircrafts	Wiener	0.0006 ± 0.0035	16.378 ± 7.4762
	DAP Design	0.0761 ± 0.1621	-
	Our model	**0.0182 ± 0.0234**	**1.3702 ± 0.8983**
Storms and rain	Wiener	0.0007 ± 0.0064	16.908 ± 6.8374
	DAP Design	0.0877 ± 0.0879	-
	Our model	**0.0202 ± 0.0758**	**−1.2728 ± 3.7621**
Traffic jams	Wiener	0.0006 ± 0.0036	17.328 ± 13.1234
	DAP Design	0.0693 ± 0.3245	-
	Our model	**0.0154 ± 0.2522**	**0.4991 ± 1.3253**
Wind	Wiener	0.0004 ± 0.0003	17.505 ± 9.6523
	DAP Design	0.0687 ± 0.0319	-
	Our model	**0.0182 ± 0.9325**	**−0.1870 ± 3.2167**

It is important to note that MSE is not an accurate metric for assessing audio quality as it does not always align well with the human perception of sound quality that involves subjective factors such as psychoacoustics, human hearing sensitivity, and preferences. When starting with an image that is essentially a Fourier transform of an audio signal, converting it back into an audio signal using the inverse transform does not necessarily reflect how well the audio will turn out. Even a minor change in a single pixel can drastically alter the sound obtained upon reversing the transformation. This fact is due to the use of a parameter of the original audio called the phase of the wave. This parameter is lost when transforming the audio into a spectrogram and a change in its value can make the audio obtained after transforming the spectrogram back into something inaudible. In our case, we have used the Inverse STF, and we have used the phase of the audio with noise that is used as an input in the model. Although the results are not perfect, the evaluation demonstrates that they are good enough, but should be improved in the future.

Therefore, MSE values do not precisely represent differences in the quality of the resulting sound. However, we can assert that if the MSE is very low, the two images are highly similar at the pixel level. This implies that the spectrogram obtained after the noise-cleaning process is highly like the original (noisy) one that was fed into the model, meaning that the cleaning process worked poorly. Therefore, while MSE may not be a perfect indicator of audio quality, it does serve as a useful metric for evaluating how closely the processed image resembles the original one in terms of their pixel-level similarity. SNR works with the converted signal and reflects whether the amount of noise in the signal is

high or low. A combination of both metrics provides a more comprehensive evaluation of the model's performance in both the image and audio domains.

Based on the results shown in Table 3, our model outperforms in all scenarios, reducing both the image noise and the actual noise in the subsequently converted signal. In terms of MSE, Wiener filters perform poorly across all four use cases, as the audios being compared are nearly identical. The DAP Design shows better results, but it is important to note that this method removes all information, including speech, resulting in silent audio. Obviously, in this case, the MSE value should be high, as noisy audio differs significantly from silent audio. MSE can still be calculated when comparing images, even if the audio signal itself is inaudible. Regarding the Signal-to-Noise Ratio (SNR), our model significantly outperforms Wiener filters. On the other hand, DAP Design's approach leads to silent audio, rendering the SNR value empty.

When we look at the results for different situations, it seems that storm and rain noises are easier to eliminate when evaluated using both MSE and SNR metrics. When considering the standard deviation, instances involving wind noise tend to yield poorer results in some examples. This variability in performance across different types of environmental noise underscores the complexity of the problem and the need for a more nuanced approach to noise reduction in audio signals.

4.1.2. Evaluation of Background Noise Suppression Based on Noise Type

This evaluation quantifies the differences between the spectrogram of the original audio, sourced from the TED-LIUM 3 dataset before adding background noise, and two other spectrograms: one corresponding to the audio with background noise and the other to the denoised audio. The relationship between these values serves as a measure of the level of noise suppression relative to the original clean signal. In other words, it helps us identify which type of background noise has been most effectively suppressed. By referencing both values to the same baseline—the original TED talk audio that is free of noise—we obtain a common metric for all scenarios. The difference between these metrics indicates the effectiveness of noise reduction in each case. Specifically, the better the second measurement (the difference between the restored audio spectrogram and the original audio spectrogram) is compared to the first (the difference between the noisy audio spectrogram and the original audio spectrogram), the more effectively the background noise has been eliminated in that particular use case.

We will use two metrics applied to the spectrograms: again, the MSE and the Structural Similarity Index Measure (SSIM). The SSIM was introduced by [38] and measures the perceptual similarity between images regardless of which is of better quality. It considers three image features, luminance (l), contrast (c), and structure (s), that are weighted through three constants: α, β, and γ. SSIM is calculated using Equation (11):

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \quad (11)$$

This evaluation has used 100 samples for each scenario from the previous evaluation. So, 400 instances have been evaluated in total. Regarding the scikit-image library, α, β, and γ values by default are 0.01, 0.03, and 1.5, respectively [39].

For both metrics, the same analysis can be applied. The suppression of wind noise yields the poorest performance. Both in absolute terms and as a percentage, the difference in the MSE and SSIM values is the smallest. In the case of the MSE, which measures the pixel-to-pixel similarity between spectrograms, the restored audio does resemble the original more than the noisy audio does, but by a smaller percentage (37.2%) compared to other scenarios. It is worth noting that these are the cases that most closely resemble the original audio, meaning that the distortion introduced by the noise is the lowest among all of the use cases. Therefore, the margin for improvement is smaller, as will be reflected in the subjective assessment where noise cleanliness depends on auditory perception. In the other scenarios, although the MSE of the restored audio is still high, the reduction compared to the audio with background noise is much greater. Specifically, in the case

of thunderstorms and rainfall, the quality of the work is significantly higher. This is also noteworthy considering that these types of background noise most severely affect the intelligibility of the original spoken segments. A similar analysis and conclusion apply when considering the SSIM metric instead of the MSE. In this case, storms are the best, wind is the worst, and traffic interchanges its position with aircrafts. The results of Table 4 confirm what the comparison with the baselines describes in Table 3.

Table 4. Comparison between spectrograms.

Use Case		MSE	SSIM
Aircrafts	Noisy audio	1628 ± 1246	0.6173 ± 0.1257
	Restored audio	680 ± 323	0.6257 ± 0.1012
	Absolute difference	**948**	**−0.0084**
	% reduction	**58.2%**	**−1.4%**
Storms and rain	Noisy audio	1867 ± 1100	0.5780 ± 0.0848
	Restored audio	698 ± 267	0.6075 ± 0.0754
	Absolute difference	**1169**	**−0.0375**
	% reduction	**62.6%**	**−5.1%**
Traffic jams	Noisy Audio	1329 ± 1139	0.6282 ± 0.1127
	Restored audio	609 ± 349	0.6437 ± 0.0891
	Absolute difference	**720**	**−0.0155**
	% reduction	**54.2%**	**−2.5%**
Wind	Noisy audio	683 ± 541	0.7287 ± 0.1107
	Restored audio	429 ± 174	0.7311 ±0.0769
	Absolute difference	**209**	**−0.0024**
	% reduction	**37.2%**	**−0.3%**

4.2. Subjective Tests

We present an evaluation based on listening to audios. In this case, we have created a set of audios that comprises the four use cases that have been listened to by a group of people to evaluate the amount of background noise that has been suppressed. The dataset consists of 22 different audios; for each audio, the volunteers must listen to the audio with background noise and the same audio after being processed by our model. For each of the four use cases, there were four different audios, totaling 20 audios. The two remaining audios correspond to control audios where no background noise was eliminated and have been used to check that the surveyed people were performing the test well.

During the process, volunteers could listen to each pair of audios as often as possible. Then, they had to choose between four different options: "No or practically no noise eliminated", "Some noise eliminated", "Much noise eliminated", or "All or almost all noise eliminated". The evaluation was delivered through a Google Form and 61 people answered it. All the information has been compiled in Figure 3. The top pie chart represents the average results of the evaluation of the 20 valid audios. The other pie charts depict the evaluation results for each of the use cases.

Looking at the figure above, we can conclude that most people think that the model could suppress much or almost all or the noise (90%). Only in a few cases (10%), did the evaluation result in no noise eliminated. If we look at the use cases separately, there are no big differences in the obtaining of good evaluations (from much noise to all noise) with percentages of around 90%. It should be highlighted that the wind scenario is the one with more cases of removing all the noise, which corroborates the objective evaluations obtained with the mathematical metrics (the margin for improvement is less than in the other cases). If we look at the scenarios considering that as much noise as possible is eliminated, storms/wind is the one that performs the best, but the differences are not remarkable. Another interesting point is that wind/storms, rain, and traffic are the scenarios that have reported cases where no noise was eliminated.

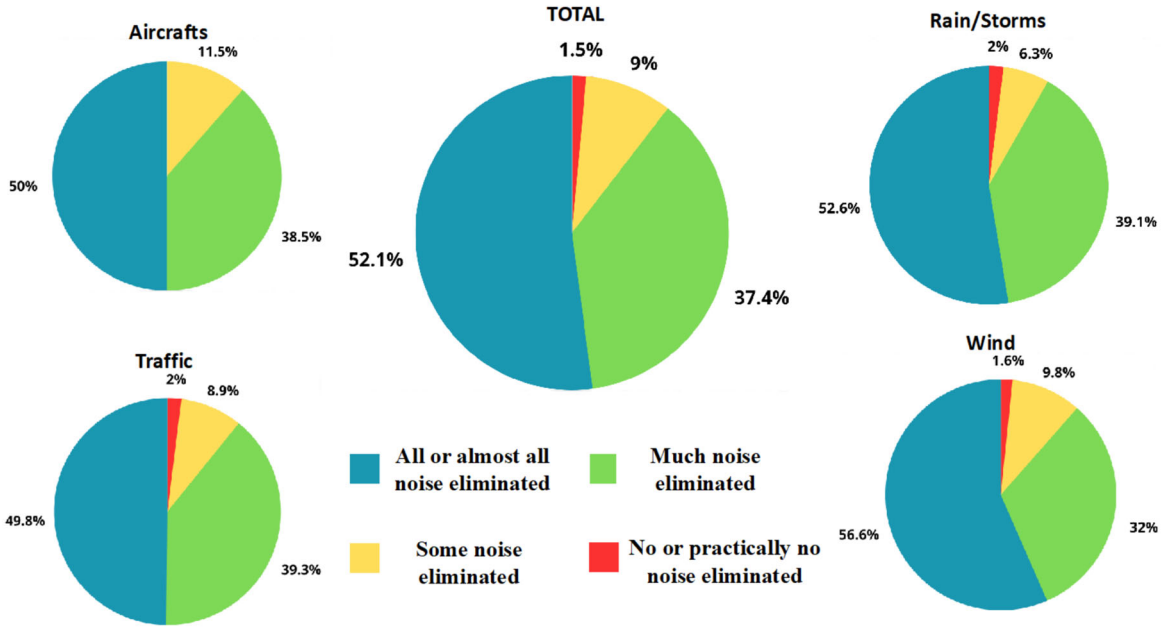

Figure 3. Human evaluation of noise suppression by listening to a set of audios.

5. Conclusions and Future Work

Audio signals can be processed either in the time domain, treating the raw signal, or in the frequency domain. In the latter case, visual representations of the audio (images) are used. This research proposes a model for cleaning background noise in audio signals (human speech) using a VAE composed of 2D convolutional filters applied to a two-dimensional representation of the audio signal, namely power log spectrograms. In the research, we propose four different scenarios that simulate environmental noise produced by storms, wind, traffic jams, and aircrafts. The whole workflow of the research comprises different stages. First, we created an ad hoc dataset by mixing speech audios with background noises, representing the four use cases. Then, we used this dataset to train a VAE with pairs of audios with background noise and only the speech audio using spectrograms. To measure if we had trained the model accurately, we used objective and subjective measures. Objective measures allowed us to measure mathematically, which is the scenario where the background noise is more difficult to suppress and whether the proposed model overcomes other models proposed before. The subjective evaluation allowed us to confirm the previous results based on the auditive perception of some people. Subjective evaluations are normally not performed in the works compiled in the state-of-the-art as there is a need to transform the spectrograms into audios so they can be listened to by the surveyed people. This transformation is not easy to apply as it depends on the phase that corresponds to the original speech's clean audio. In our case, we have applied the IFTF and although it does not achieve perfect results, it does obtain perfectly evaluable audios by using the phases of the audios that are the inputs for the model.

The mathematical metrics produced by our model confirm that it performs better than the selected baselines in all cases. If we look at the results between the use cases, we can see that storms and rains are easier to eliminate. Looking at the standard deviation, we can see that wind cases have the worse results in a few examples. In this case, it should be highlighted that the scenario of background wind is the one with the smallest differences between the noisy audio and the cleaned one. This is confirmed in the objective evaluation where surveyed people evaluate this scenario as the one with the best performance.

As future works, the main need is to obtain a method that could calculate, in a more precise way, the phase of the cleaned audio. Also, there is a need to obtain a more balanced dataset where the differences between the audio with background noise and the audio after

using the model are smaller between scenarios. From a practical perspective, the model could be integrated into applications, so the model works with streaming audio.

6. Patents

The Spanish Patent and Trademark Office (OEPM) has processed the patent application related to the work presented in this article, assigning it the number P202330047 and the filing date of 24 January 2023.

Author Contributions: Conceptualization, A.N. and Á.J.G.-T.; methodology, A.N.; software, J.C.-C.; validation, A.N., J.C.-C. and Á.J.G.-T.; formal analysis, A.N., J.C.-C. and Á.J.G.-T.; investigation, A.N. and Á.J.G.-T.; resources, Á.J.G.-T.; data curation, J.C.-C.; writing—original draft preparation, A.N.; writing—review and editing, A.N. and Á.J.G.-T.; supervision, Á.J.G.-T. and A.N.; project administration, Á.J.G.-T.; funding acquisition, Á.J.G.-T. 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: Publicly available datasets were analyzed in this study. TED-LIUM dataset data can be found here: https://www.openslr.org/51/ (accessed on 26 November 2023). Background noises can be obtained from https://www.videvo.net/es/efectos-de-sonido/viento/ (accessed on 26 November 2023) and https://zenodo.org/record/4279220#.YpdZVKhByUk (accessed on 26 November 2023).

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

References

1. Zhang, H.; Wang, D. A Deep Learning Approach to Multi-Channel and Multi-Microphone Acoustic Echo Cancellation. In Proceedings of the Interspeech 2021, Brno, Czech Republic, 30 August–3 September 2021; pp. 1139–1143.
2. Guo, G.; Yu, Y.; de Lamare, R.C.; Zheng, Z.; Lu, L.; Cai, Q. Proximal normalized subband adaptive filtering for acoustic echo cancellation. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 2174–2188.
3. Liu, B.; Liu, J. Overview of image denoising based on deep learning. *J. Phys. Conf. Ser.* **2019**, *1176*, 022010.
4. Zie, J.; Colonna, J.G.; Zhang, J. Bioacoustic signal denoising: A review. *Artif. Intell. Rev.* **2021**, *54*, 3575–3597.
5. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
6. Yuliani, A.R.; Amri, M.F.; Suryawati, E.; Ramdan, A.; Pardede, H.F. Speech Enhancement Using Deep Learning Methods: A Review. *J. Elektron. Dan Telekomun.* **2021**, *21*, 19–26.
7. Hammam, H.; Elazm, A.A.; Elhalawany, M.E.; El-Samie, A.; Fathi, E. Blind separation of audio signals using trigonometric transforms and wavelet denoising. *Int. J. Speech Technol.* **2010**, *13*, 1–12. [CrossRef]
8. Yu, G.; Mallat, S.; Bacry, E. Audio denoising by time-frequency block thresholding. *IEEE Trans. Signal Process.* **2008**, *56*, 1830–1839. [CrossRef]
9. Yu, G.; Bacry, E.; Mallat, S. Audio signal denoising with complex wavelets and adaptive block attenuation. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07 2007, Honolulu, HI, USA, 15–20 April 2007; Volume 3, pp. III-869–III-872.
10. Ng, L.C.; Burnett, G.C.; Holzrichter, J.F.; Gable, T.J. Denoising of human speech using combined acoustic and EM sensor signal processing. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Proceedings (Cat. No. 00CH37100), Istanbul, Turkey, 5–9 June 2000; Volume 1, pp. 229–232.
11. Tan, K.; Zhang, X.; Wang, D. Deep learning based real-time speech enhancement for dual-microphone mobile phones. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 1853–1863. [CrossRef]
12. Nogales, A.; Donaher, S.; García-Tejedor, Á. A deep learning framework for audio restoration using Convolutional/Deconvolutional Deep Autoencoders. *Expert Syst. Appl.* **2023**, *230*, 120586. [CrossRef]
13. Pascual, S.; Bonafonte, A.; Serra, J. SEGAN: Speech enhancement generative adversarial network. *arXiv* **2017**, arXiv:1703.09452.
14. Abouzid, H.; Chakkor, O.; Reyes, O.G.; Ventura, S. Signal speech reconstruction and noise removal using convolutional denoising audioencoders with neural deep learning. *Analog. Integr. Circuits Signal Process.* **2019**, *100*, 501–512.
15. Kantamaneni, S.; Charles, A.; Babu, T.R. Speech enhancement with noise estimation and filtration using deep learning models. *Theor. Comput. Sci.* **2023**, *941*, 14–28. [CrossRef]
16. Mukhutdinov, D.; Alex, A.; Cavallaro, A.; Wang, L. Deep learning models for single-channel speech enhancement on drones. *IEEE Access* **2023**, *11*, 22993–23007.

17. Shi, H.; Shu, Y.; Wang, L.; Dang, J.; Kawahara, T. Fusing multiple bandwidth spectrograms for improving speech enhancement. In Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Chiang Mai, Thailand, 7–10 November 2022; pp. 1938–1943.
18. Pandey, A.; Wang, D. On cross-corpus generalization of deep learning-based speech enhancement. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2020**, *28*, 2489–2499. [PubMed]
19. Roy, S.K.; Nicolson, A.; Paliwal, K.K. A Deep Learning-Based Kalman Filter for Speech Enhancement. In Proceedings of the Interspeech 2020, Shanghai, China, 25–29 October 2020; pp. 2692–2696.
20. Nossier, S.A.; Wall, J.; Moniri, M.; Glackin, C.; Cannings, N. An experimental analysis of deep learning architectures for supervised speech enhancement. *Electronics* **2020**, *10*, 17.
21. Luo, H.; Lu, S.; Wei, Q.; Fu, Y.; Tian, J. Spectrogram-based speech enhancement by spatial attention generative adversarial networks. In Proceedings of the 14th International Conference on Digital Image Processing (ICDIP 2022), Wuhan, China, 20–23 May 2022; pp. 773–779.
22. Siegert, I.; Lotz, A.F.; Duong, L.L.; Wendemuth, A. Measuring the impact of audio compression on the spectral quality of speech data. *Stud. Zur Sprachkommun. Elektron. Sprachsignalverarbeitung* **2016**, *2016*, 229–236.
23. Kanetada, N.; Yamamoto, R.; Mizumachi, M. Evaluation of Sound Quality of High Resolution Audio. *Jpn. J. Inst. Ind. Appl. Eng.* **2013**, *1*, 52–57. [CrossRef]
24. Sourceforge SoX-Sound. Available online: http://sox.sourceforge.net/ (accessed on 26 November 2023).
25. Wu, N.; Wang, X.; Lin, B.; Zhang, K. A CNN-based end-to-end learning framework toward intelligent communication systems. *IEEE Access* **2019**, *7*, 110197–110204. [CrossRef]
26. Repp, A.; Szapudi, I. Precision prediction of the log power spectrum. *Mon. Not. R. Astron. Soc. Lett.* **2017**, *464*, L21–L25. [CrossRef]
27. McFee, B.; Raffel, C.; Liang, D.; Ellis, D.P.; McVicar, M.; Battenberg, E.; Nieto, O. librosa: Audio and music signal analysis in python. In Proceedings of the 14th Python in Science Conference, Austin, TX, USA, 6–12 July 2015; Volume 8, pp. 18–25.
28. LeCun, Y.; Boser, B.; Denker, J.; Henderson, D.; Howard, R.; Hubbard, W.; Jackel, L. Handwritten digit recognition with a back-propagation network. *Adv. Neural Inf. Process. Syst.* **1989**, *2*.
29. LeCun, Y.; Haffner, P.; Bottou, L.; Bengio, Y. Object Recognition with Gradient-Based Learning. In *Shape, Contour and Grouping in Computer Vision*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1681. [CrossRef]
30. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [CrossRef] [PubMed]
31. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. *arXiv* **2013**, arXiv:1312.6114.
32. Bergstra, J.S.; Bardenet, R.; Bengio, Y.; K'egl, B. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*; NIPS: New Orleans, LA, USA, 2011; pp. 2546–2554.
33. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
34. Lim, J.S.; Oppenheim, A.V. Enhancement and bandwidth compression of noisy speech. *Proc. IEEE* **1979**, *67*, 1586–1604. [CrossRef]
35. Narayanaswamy, V.S.; Thiagarajan, J.J.; Spanias, A. On the Design of Deep Priors for Unsupervised Audio Restoration. *arXiv* **2021**, arXiv:2104.07161.
36. Sammut, C.; Webb, G.I. (Eds.) *Encyclopedia of Machine Learning*; Springer Science & Business Media: New York, NY, USA, 2011.
37. Elkum, N.; Shoukri, M.M. Signal-to-noise ratio (SNR) as a measure of reproducibility: Design, estimation, and application. *Health Serv. Outcomes Res. Methodol.* **2008**, *8*, 119–133. [CrossRef]
38. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
39. Scikit-Image Library. Available online: https://scikit-image.org/ (accessed on 12 December 2023).