Analog Integrated Circuits and Signal Processing (2019) 100:501–512
https://doi.org/10.1007/s10470-019-01446-6

Signal speech reconstruction and noise removal using convolutional denoising audioencoders with neural deep learning

Houda Abouzid[1] · Otman Chakkor[1] · Oscar Gabriel Reyes[2] · Sebastian Ventura[2]

Received: 21 September 2018 / Revised: 15 March 2019 / Accepted: 19 March 2019 / Published online: 27 March 2019

Abstract
Datasets exist in real life in many formats (audio, music, image,...). In our case, we have them from various sources mixed together. Our mixtures represent noisy audio data that need to be extracted (features), compressed and analysed in order to be presented in a standard way. The resulted data will be used for the Blind Source Separation task. In this paper, we deal with two types of autoencoders: convolutional and denoising. The novelty of our work is to reconstruct the audio signal in the output of the neural network after extracting the meaningful features that present the pure and the powerful information. Simulation results show a great performance, yielding of 87% for the reconstructed signals that will be included in the automated system used for real word applications.

Keywords Denoising autoencoder · Convolutional autoencoder · BSS · Keras · Deep learning · Neural network

1 Introduction

Speech signals are often mixed with background noise existing in real life environment. Most existing methods focus on music/voice separation. In this part of work, we focus on noise suppression. The general problem related to this type of task is known by the "Blind Source Separation" or BSS [1]. The idea lies in the fact of reconstructing individual sound signals from a mixture of multiple audio sources. This operation is done without the aid of information about the characteristics of the original signals. Most of studied cases, have considered this problem hard to solve and complexe due to the ignorance of such useful information about the number of original sources in the observed mixtures, their natures and their statistical properties as well. The mathematical science has called this type of hard solving problems as "the ill posed problem". This means that there is no one unique solution, but in the fact, there will be a multitude of different solutions. We talk about *Blind Source Separation* when the estimated sources are recovered by unsupervised methods.

In audio source separation, we suppose that the sources are statistically independent. In our case, we will suppose that the noise represent an other type of sources that need to be separated from the original signals. Many researchers have been attracted to work on this type of problem and have used many methods like the maximum likelihood estimation, negentropy maximization and the Principle Component Analysis known by PCA [2, 3]. All those methods are generated from the famous approach called Independent Component Analysis (ICA). The large body of applied methods that have been used for solving the BSS problem are divided into two types of machine learning paradigms: unsupervised learning and supervised learning [4].

This work has been supported by the research group *KDIS* (Knowledge Discovery and Intelligent Systems) during my research period at the *KDIS* laboratory, department of computer science and numerical analysis at the University of Cordoba, Spain.

✉ Houda Abouzid
abzdhouda@gmail.com

Otman Chakkor
o.chakkor@gmail.com

Oscar Gabriel Reyes
ogreyes@uco.es

Sebastian Ventura
sventura@uco.es

[1] Telecommunications Department, ENSATE, Abdelmalek Essaadi University, Tétouan, Morocco

[2] Department of Computer Science and Numerical Analysis, University of Cordoba, Córdoba, Spain

In this paper, we use the Short-Time Fourier Transform (STFT) to extract the features and generate the spectrum matrix in order to represent the mixed source signal as a primary step, and then for the second step, there will be using two deep autoencoders called *Convolutional Denoising Autoencoders* (CDAEs) to represend the spatio-temporal units of this spectrum matrix. The denoising autoencoder is used for the suppression of noise from the mixtures as a first purpose of use and for the compression data as a second goal of this application.

The proposed framework is to combine between two types of autoencoders to generate a new hybrid structure giving it the name of *S-Convolutional Denoising Autoencoders* (SCDAEs). The Short-Time-Fourier Transform (STFT) is calculated on a signal time of mixture audio. The resulted magnitude spectrogram is then passed through the firt convolutional autoencoder used for compression task only , while the output resulted obtained is then paseed through the second autoencoder which is a denoising autoencoder used for a noise removal and compression. This last autoencoder has different parameters and number of layers than the first one.

Our approach is computationally efficient and can be run in real time. Our proposed contribution is using convolutional architechture neural network for signal processing which is useally used for image processing. This work demonstrates the effectiveness of fully connected convolutional model on audio signal generation task.

The rest of this paper is organized as follows: the Sect. 2 brievely describes the state-of-the-art in solving the (BSS) problem, distinguishing them between unsupervised methods and supervised ones.The underlying idea and the methodology is presented in the Sect. 3. The proposed approach is explained in the Sect. 4. The result implementations and their descriptions are shown in the Sect. 5. Finally, we conclude in Sect. 6.

2 Related works

Convolutional Neural Network (CNN) has been so effective and powerful in image precessing domain like: compression, object identification, automatic image colorization of grayscale images [5], etc. Our contribution is dealing with audio data which is not something usually used with CNN. In [6] the author applied a low-latency monaural source separation framework using a Convolutional Neural Network (CNN). The CNN is used to estimate the time-frequency soft masks which are applied for source separation. The performance of the neural network was tested on a database comprising of musical mixtures of three instruments: voice, drums, bass as well as other instruments which vary from song to song.

Audio applications Many applications dealing with the BSS are used [7]. The proposed approach could be used in many real-world applications such as: the compression of audio data, telephony, speech recognition, audio synthesis and many other areas. In addition to having multiple practical applications, our approach can be used in real time and suggests the supprission of noise.

2.1 Blind source separation

As we know, human beings use speech sound to communicate with each other and even when they are surrounded by noise they could simply concentrate on a single voice and ignore the others as a disturbing background noise. This ability to focus on one speech signal and interact easily with the direct conversation is only a gift minded for humans. Unfortunately, when we talk about human–machine interaction we do not speak about this powerful ability because it is absent for all machines and especially for the humanoid robot. As a use case, suppose that a robot is placed in a noisy environment and an order is given to it. In fact, what is going to happen is, that robot will not be able to focus on the direction of the specific speech and will be lost at the end without knowing what to do exactly. In order to solve this kind of problem, the robot must first pass through a separation process of audio mixtures. To capture those speech mixtures arrived from different locations, it uses its microphones placed on its head.

The interest of studying the Blind Source Separation problem has begun since 1980 and still attracting many researchers in this field until nowadays. Due to the great importance of what BSS could solve for many problems, there are some popular applications that are related directly with BSS and as a type of example we mention: the extracting features from different data to get some information as diverse as neuronal activities and brain image, audio separation, communications and telecommunication, video and all what is related with sensor signals. Also in audio analysis applications, the suppress of noise for speech recognition and audio text alignment helps a lot to improve the quality of the separation in a noisy environment [8, 9].

The problem formulation for BSS is explained as:

Supposing the number of channels is denoted as I and which represents the number of observations where $x(t) \in \Re^{I\times 1}$ is the observed I-channel mixture signal.

J is the number of sources denoted where $c_j(t) \in \Re^{J\times 1}$ is the J-channel spatial image of source j. The mixtures and the sources are both related by:

$$x(t) = \sum_{j=1}^{J} c_j(t) + n(t), \tag{1}$$

and $n(t)$ is the noise.

The purpose of the BSS is to estimate the source spatial images $c_j(t)$ from the observed mixture signal $x(t)$.

In this paper we focus on the suppression of the noise n and the reconstruction of the mixtures x.

2.2 Unsupervised learning

Unsupervised leaning are methods where no desired response is associated as an output data. For the separation task, unsupervised leaning refers to methods that learn the structure of the pattern and try to separate the sources mixed together in the mixtures data. This operation is based on some hypothesis that help to solve the BSS problem instead of using the characteristics of the sources or the mixing process that are supposed to be unknow [4]. There are some famous algorithms that are useful for the same goal like the Independent Component Analysis (ICA) [10, 11], Sparse coding [12], Computational Auditory Stream Analysis (CASA) [13], Beamforming [14] and Non-negative matrix factorization (NMF) [15].

In [16], the author proposed a novel framework for unsupervised source separation using a deep autoencoder. Here, he explored the unknown characteristics of the signals in the mixed input sources by extracting the features using the autoencoder implemented by a neural network with multi hidden layers and then he took those coefficients and classified them on clusters for the separation task.

Our proposed model is considered as an unsupervised learning method because we consider that we only have input data as a matrix X and there are no corresponding output variables. Also there is no prior information about the existing sources and the mixture process as well. This is due to the kind of the procedure that we deal with and which is corresponding to the BSS problem.

The major purpose of the unsupervised learning is when the model is trained it tries to learn more about the provided input data. Unlike supervised learning, there is no teacher or expert to select the intesting structure in the data. Here the unsupervised algorithm learns to inherent the structure from the input data.

2.3 Supervised learning

Supervised learning requires the existence of an expert who has knowledge of the environment. Here we should have both variables: the input X and the output Y. The algorithm is used to learn the mapping function from the input to the output with the corresponding equation:

$$Y = f(X) \tag{2}$$

The desired answers provided by the expert describe the function of the network. The role of the learning algorithm is to get the network to fill this equation and then predict the output variables Y when new input data X is provided. So the correct answers are known and the algorithm iteratively makes predictions on the training data and it is corrected by the teacher that supervises the result process. When the performance level achieves a good value the learning stops.

The term supervised learning in signal processing field and BSS task refers to machine learning algorithms that try to solve the separation problem having a known dataset as a training data to make predictions. These predictions represent the estimated signals expected from their original mixtures. Those training data contain the input data as representing mixtures and the output data referring to their responses values [17]. In order to make predictions, a model is required to train the process of estimation. This is done using some weights that are recalculated each time according to the number of epochs of the pattern. When the calculated result of the prediction is so far from the output training data, this process will be iterated each time until getting better accuracy of the model.

Supervised learning includes several algorithms such as: support Vector Machine (SVM) [18], Gaussian mixture model (GMM) [19, 20], the classification task and Regression.

3 Noise removal based on CDAEs

3.1 Data onset detection

Audio Onset Detection plays a major role to find the time-locations of all sonic events in a piece of audio [21, 22].

Supposing we have started the discret time signal represented by $X[n]$, the process with the autoencoder encodes this signal to binary string (see Fig. 1), the decoder will take these binary string and convert it back into the sound signal $Y[n]$. n represents in this exemple the number of samples existing in the input signal.

The goal of using an encoder is to reduce the number of bits to represent the original signal $X[n]$. Because of this process, the memory requierement will be able to store the sound waveform and reduce the amount of data in a file because it will be compressed. The reduction dimension process is done in the encoder part.

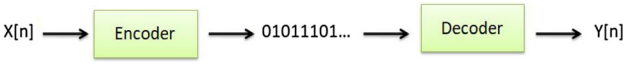

Fig. 1 Lossy compression scheme

Before starting our approach, we must first of all analyze the audio sequence and the first step will be the onset detection. What we do here is to define the beginning of the transient part of an audio [23], or the earliest moment at which a transient can be reliably detected to find the meaningful sound events in audio signals.

Onset detection for audio processing is one of the most fundamental tasks in music information retrieval and it is often used for many different real time applications. Between the most relevent onset detection applications is the automatic score followers [24] that can be proved by introducing (online) onset detectors that search for note onsets in a live performance, while (offline) onset detection is used at the present time to improve digital audio workstations for audio processing. Actually, automatic detection of events in audio signals has a great importance for sound source separation problem, music information retrieval and automatic music transcription as well.

The principle of the onset detection is based on the calculation of signal features that show the presence of transients in the sound signal using either signal analysis techniques or probabilistic models.

The Fig. 2 presents an onset detection backboard architecture that can be included in an audio source separation system before starting the reconstruction phase of mixtures.

In this architecture there are four information steps: spectral novelty function, peaks, spectra and segments. In the first level, *Spectral novelty function* corresponds to the function responsible of the reduction methods used for the implementation knowing that an audio mixture is included as an input signal. The second level corresponds to *Peaks*, here the result of the obtained peaks are generated using a threshold and some algorithms. The third level is dedicated for the *Spectra* which contains the magnitude and phase of the spectrum of the segments defined in the first level according to the *fft* size and window type parameters.

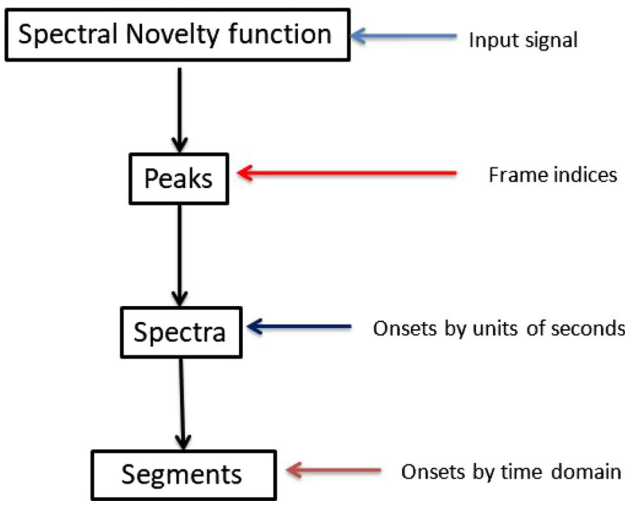

Fig. 2 The architecture of onset detection for an audio signal

After all those steps, comes the final one which is *Segments* and their segmentation results according to some parameters like (time domain, overlap, frame length and number).

Mathematically, we can define a peak picking function that identifies the local maxima of the novelty function above according to some defined threshold. This function calculates the mean value of the detection function within a time interval. And finally a peak is detected if a local maximum satisfied this following condition:

$$\frac{d(n)}{mean\{d(n-M_l),\ldots,d(n+M_u)\}} \geq \lambda \tag{3}$$

where $d(n)$ is the novelty detection function, λ is the threshold and M_l and M_u are the lower and upper limits for the mean calculation.

After analyzing the presented signal , now is time to apply an autoencoder for the audio source mixture.

3.2 Convolutional feedforword neural network

In general, a simple definition of an autoencoder can be defined as a neural network that is trained to reconstruct an output as similar as its input. Since it doesn't need any targets or specific labels, it can be trained in an unsupervised method. Our approach based on autoencoders is considered as an unsupervised learning technique, since it does not need explicit labels to train the model on. All we need to train the model is a raw input data obtained from audio file experiment. The first autoencoder is a feedforward ANN that is trained to reconstruct the input data as much perfect as possible. A simple exemple of an autoencoder is the one who has one input layer directely connected to one hidden layer of nodes, and this last is connected by the same way to one output layer. The output result comming from the hidden layer is the encoding of the input network. Using a deep model with multiple hidden layers proves its optimization for the classification problems and regression tasks [25].

3.3 Underlying idea using a denoising autoencoder

Autoencoders are unsupervised learning algorithms because there is no really need for explicit labels to train the model. The algorithm takes the input data which is in our case represented by audio signals, and try to reconstruct it with only fewer number of bits from the latent space. This operation is done with compression of the data during the time training of the neural network. As the Principle Component Analysis (PCA) first goal is to reduce the

dimension space [26], the autoencoders play the same role. In general, the idea is to project the dataset in a smaller space with removing some unuseful parts. Knowing that PCA uses linear transformation otherwise, the autoencoder uses the non-linear transformation. This is the big difference between them.

A denoising autoencoder (DAE) is a special type of a fully connected feedforward neural networks that takes noisy input signals and outputs their denoised version. DAE are often used in deep learning and even with noisy environnement. They are used to reduce the dimension features in perturbed signals. Their inputs are the spectral frames of the mixed signal and the outputs are the spectral frames of the target sources. A(DAE) try to learn a representation (latent space) that is removed by defining a loss function (the mean squared error). At every iteration of the network, it will compute the loss between the noisy outputed features by the decoder and the denoising features and try then to minimize it.

Knowing that there are many practical applications of autoencoders, the data denoising and dimensionality reduction for data visualization still represent the most important goals. The data projection obtained then is more interesting than PCA or the other basic techniques. Otherwise, what gives the autoencoders this major importance is their capacity to solve the problem of unsupervised learning, i.e. the learning of useful representations without the need for labels.They are also considered as a self-supervised technique, where the targets are generated from the input data.

For this reason, first of all, we will build up a new signal of mixture based on deep learning neural network , the goal of this step is the reconstruction of the input using five layers (the first and the last one has 150 neurons eatch, and there hidden layers having each one of them 120, 70 and 70 neurons succesively). The idea here is to tell to the encoder to mask out or as we say (mathematically) to set to zero some of the inputs passing through the 3 hidden layers, in order to reconstruct a new input empty from noise and also to compress the data set into a low-dimensional space. The Fig. 3 explains this procedure step by step. This method will simplify the computation while using the neural networks strategy.

The goal of the encoder will be, in an other way of explaining, as a filter of the existing noise using the hidden units of the neural networks for the auto encoder. So, we thought to allow one of the neurons of the hidden layers to eliminate or suppress the noise from the input and then the output will be the original input minus what that neuron encoded as a disturbing information.

The architecture of our idea can be seen in the Fig. 3.

Our approach uses the denoising autoencoder with the architecture described in the Fig. 4.

The proposed structure of our denoising autoencoder has five layers defined as folow: the first layer is the input layer

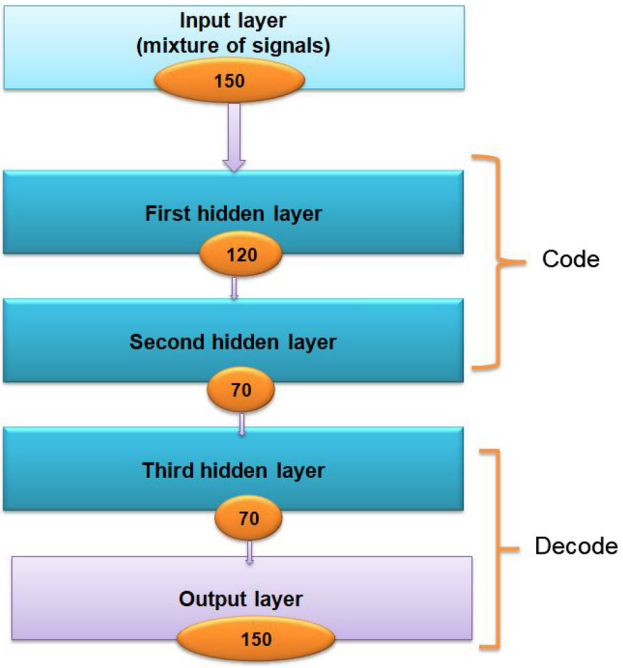

Fig. 3 The structure of the autoencoder using neural network

which contains 150 extracted features of mixed audio signal used as an input data , the two other succesive layers represent the hidden layers for the neural network. The first hidden layer has 120 nodes and the other has 70 nodes. These two hidden layers constitute the encoder. The decoder then , is formed by two last layers which are the third hidden layer contining 70 nodes and by the output layer of 150 features.

To explain what is happened in this architecture, we highlight the most important parts of this structure which are the encoder and the decoder as follow:

- *E*ncoder In this part, the network compresses the input samples into a fewer number of bits. Those compressed bits represent the original input and they are called the "encoding" of the input.
- *D*ecoder Here in this part of the structure, and using only the encoding input, the network tries to reconstruct the input. When the decoder is capable of well reconstructing the input as they have been feded to the encoder, then it is said that the encoder produces the best encodings of the input.

4 The proposed approch based on S-CDAE

In this work we implement a learning approach using two types of autoencoders. This approch learns speech patterns from data mixtures only. From our model, we will recover clean speech generated at the output neural network

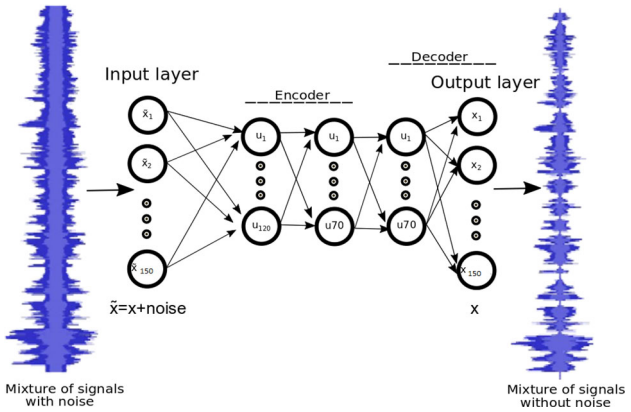

Fig. 4 The proposed architecture of a denoising autoencoder (DAE) using neural network

system. In this part of study, the Convolutional and Denoising autoencoders are selected to process the learning. In most works that used the DAE, authors introduced random corruptions to the input features and feed them to the model training. For us, we will not do this, instead we will try to remove the already existing correptions considered as noisy parts of the audio file from the input original data. Generaly, DAEs have approved their powerful learning on noise suppression and demension reduction representations. For this reason, we will use the DAE to remove the existing noise from the speech mixtures and try to reconstruct again the pure mixtures signals.

In CDAE, the training model is operated with a deep neural network (DNN) structure that allows learning high-level speech model layer by layer. The architecture of the DNN is a fully connected neural network. Otherwise, due to the low-dimensional represented model, the structure is performed to be trained with a small amount of data.

5 Experimental study

5.1 Data set

The data set that has been used for testing the new approach has been taken from the web site of *irisa*[1] where the data set contains three types of stereo mixtures and a live recordings represent a static sources played through loudspeakers in a meeting room, recorded one at a time by a pair of omnidirectional microphones and subsequently added together. The file that we have used is called "test data" and it is made available under the terms of the Creative Commons Attribution-Non Commercial-ShareAlike 2.0 license. The authors are Glen Phillips, Mark Engelberg, Psycho Voyager, Nine Inch Nails and Ali Farka Touré for music source signals and Shoko Araki and Emmanuel Vincent for mixture signals. This data represents the case of under-determined speech and music mixtures which means that the number of original sources is much greater than the number of microphones.

The test data contains three types of stereo mixtures:

- Instantaneous mixtures (static sources scaled by positive gains).
- Synthetic convolutive mixtures (static sources filtered by synthetic room impulse responses simulating a pair of omnidirectional microphones.
- Live recordings (static sources played through loudspeakers in a meeting room, recorded one at a time by a pair of omnidirectional microphones and subsequently added together).

About the room dimensions, they are the same for synthetic convolutive mixtures at *SiSEC2015* and live recordings ($4.45 \times 3.55 \times 2.5$ m). The reverberation time is set to 130 ms or 250 ms and the distance between the two microphones to either is 5 cm or 1 m.

Our datasets have been taken from *irisa* (Institute for Research in Computer Science and Random Systems), created in 1975, which is a joint research unit (UMR 6074) in computer science, automation, signal and image processing and robotics. On the conjunction of these themes, IRISA is positioned as the major research laboratory in Brittany. Also the datasets provided are devided in different categories such as (under-determined, determined and overdetermined) speech and music mixtures and (instantaneous or convolutive mixtures), so the one can choose between these categories according to his preferences. In addition to that, this datasets have been always tested, evaluated and updated before being uploaded in the IRISA web site and being ready for the public use.

5.2 Experimental settings

In the deep learning litterature, the most works that have been published applied the recurrent networks for signal audio processing based on *theano* library. Our approach instead of using this type of neural network for speech signals and this library , we thought of another type which is the convolutional neural network based on *Keras* deep learning library. Keras is a high-level neural networks API that allows easy and fast prototyping. We have also used many other signal processing libraries such as *Librosa* and the library *Pandas* for data analysis. The most important advantages of using *Keras* is that on first hand, it supports both convolutional and recurrent networks or the combination of the two, and it could be running seamlessly on CPU and GPU on the second hand.

[1] http://sisec2008.wiki.irisa.fr/tiki-index.php?page=Under-determined+speech+and+music+mixtures.

For each mixing condition, 6 mixture signals have been generated from different sets of source signals placed at different spatial positions:

- 4 male speech sources
- 4 female speech sources
- 3 male speech sources
- 3 female speech sources
- 3 non-percussive music sources
- 3 music sources including drums

As we are focusing our approch on speech mixtures only, we choose to perform our test on the first foor types of speech mixtures (4 male speech sources, 4 female speech sources, 3 male speech sources, 3 female speech sources).

About the source directions of arrival, they vary between $-60°$ and $+60°$ with a minimal spacing of $15°$ and the distances between the sources and the center of the microphone pair vary between 80 cm and 1.20 m. The data consists of stereo WAV audios [27], which were imported in Python by using the framework Keras (Keras was used to implement the autoencoder) and other signal processing libraries.

5.3 Experiments and discussion

Our data is based on instantaneous underdetermined mixtures. We begin our test with the signal mixture of 4 female speech audio and then we will do the same experiment with the other mentionned data. Since we will repeat all the procedure to all types of data, we will provide the detail simulations and explications for the firt test data (4 femele speech mixture), and the others we will resume the result in a detailed table where we will present the comparaison between the 4 different type mixtures.

So now let's load our fist test with the 4 femele audio into a numpy array of 220,500 samples and 22,050 Hz as a sampling frequency showed in the Fig. 5.

The sampling rate of this signal is equal to 22,050 Hz.

The spectrogram of the audio is presented in the Fig. 6.

The onset frames obtained is as follow (Fig. 7).

After converting the onset frames to units of seconds (see the Fig. 8), it shows that at each detected time there is an onset that uses the frequency representation of the signal according to time. Those obtained onsets could be seen and read obviously from the Fig. 9.

At the beginning, those onsets were as 66 frames (Fig. 7 which could not help us to show them on the signal, so to do this now, we implement them according to time units in the signal of mixture.

The Table 1 shows the indices of those frames.

The following Table 2 shows us those frames after having been converted to time domain.

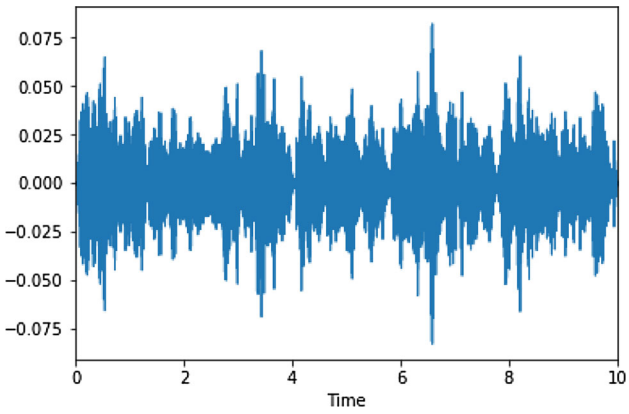

Fig. 5 Audio wave of the sound mixture of 4 females speech

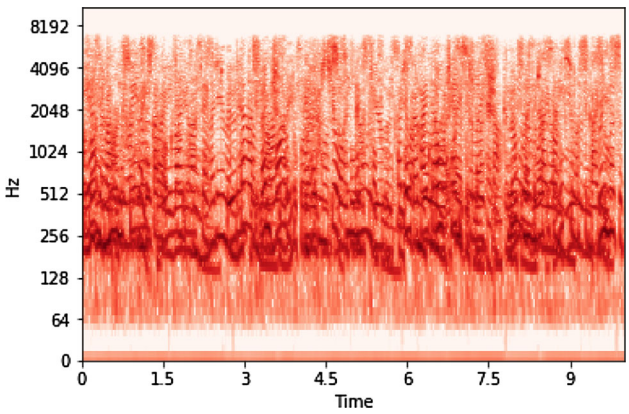

Fig. 6 The spectrogram of the 4 female speech mixture

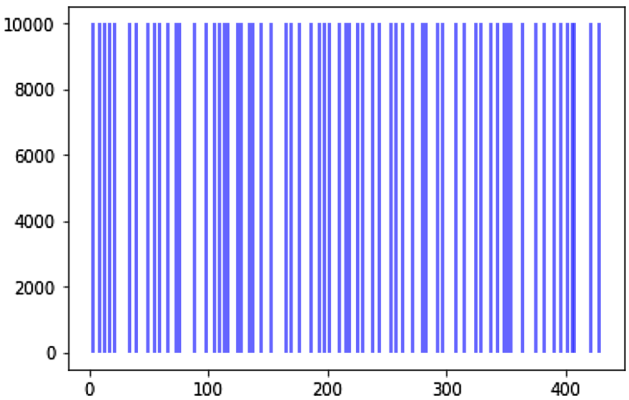

Fig. 7 The onset frames of the 4 female speech mixture

Now let's plot those onsets detection with the time domain waveform that is shown in the Fig. 10.

The idea of our proposed method is to combine between two types of autoencoders. The first autoencoder is a convolutional autoencoder and the second is a denoising autoencoder. The first one, converts the input audio data of matrix of (1, 160,000) dimension to an array , reshape it so it will be of size 200×1, which represent the encoded

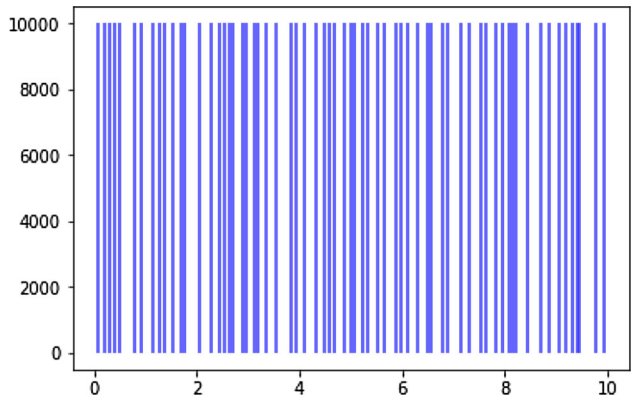

Fig. 8 Onsets time after being converted in time domain

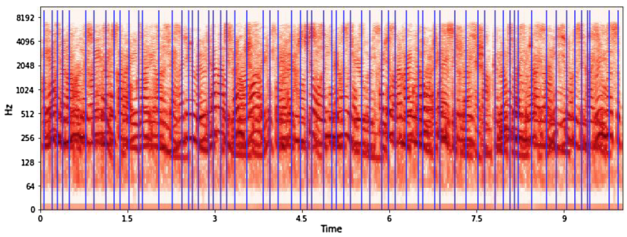

Fig. 9 Onsets time on the top of the spectrogram of the 4 femele sepeech mixture

Table 1 Frame indices for estimated onsets in audio signal

The 66 frame indices obtained for estimated onsets									
3	9	13	17	22	34	40	49	55	59
66	73	76	88	98	105	110	113	117	125
128	134	138	144	153	165	170	176	186	193
198	201	210	216	219	225	230	238	244	253
258	263	271	280	283	292	296	307	315	324
329	337	343	348	351	354	364	375	382	390
396	401	405	407	421	428				

version of the file, then the denoising autoencoder takes those encoded data as a first input of the input layer, reshape it again to 150 dimension and passes it to the succesive hidden layers until the output layer that gives the final version of the encoded denoised data.

The pattern stucture of the first autoencoder is presented in the Table 3.

The model that we choose for the denoising autoencoder is the one with the optimizer SGD and the loss function mean squared error. The iteration of the implemented program is for 50 epochs for each autoencoder.

After those iterations the detail structure of the denoising autoencoder used in this experiment is shown in the Table 4.

Table 2 Converted 66 frames to time units in time domain

The 66 converted frames in time domain				
0.06965986	0.20897959	0.30185941	0.39473923	0.510839
0.78947846	0.92879819	1.13777778	1.27709751	1.36997732
1.53251701	1.69505669	1.76471655	2.04335601	2.27555556
2.43809524	2.55419501	2.62385488	2.71673469	2.90249433
2.9721542	3.11147392	3.20435374	3.34367347	3.55265306
3.83129252	3.94739229	4.08671202	4.31891156	4.48145125
4.59755102	4.66721088	4.87619048	5.0155102	5.08517007
5.2244898	5.34058957	5.52634921	5.66566893	5.87464853
5.9907483	6.10684807	6.29260771	6.5015873	6.57124717
6.78022676	6.87310658	7.12852608	7.31428571	7.52326531
7.63936508	7.82512472	7.96444444	8.08054422	8.15020408
8.21986395	8.45206349	8.70748299	8.87002268	9.05578231
9.19510204	9.31120181	9.40408163	9.45052154	9.77560091
9.93814059				

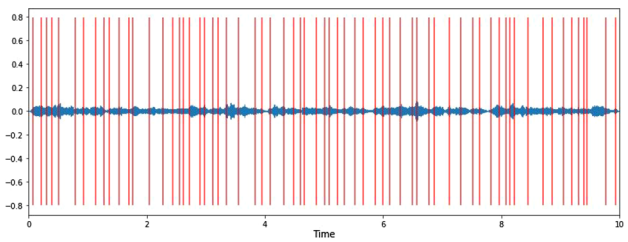

Fig. 10 Waveform of the signal with their detectable onsets in time domain

Table 3 The detail structure of the proposed first autoencoder

DAU model summary		
The input data with 66 frames and 22,050 frequency bins		
Layer (type)	Filters number	Output shape
Input data (InputLayer)	0	(1, 200)
Dense 1 (Dense)	30,150	(1, 150)
Dense 2 (Dense)	30,200	(1, 200)
The output data has a total number of parameters: 60,350		
Non-trainable parameters: 0		

In the first experiment, we have employed the convolutional autoencoder that give us the following results of the signal: the original audio mixture mask constructed by the convolutional autoencoder in the firt layer is shown in the Fig. 11.

The encoding mask after the compressing of the original file in the hidden layres in the middle of the implementation program is presented in the Fig. 12.

Table 4 The detail structure of the proposed denoising autoencoder

DAU model summary		
The input data with 66 frames and 22,050 frequency bins		
Layer (type)	Filters number	Output shape
Input data (InputLayer)	0	(1, 150)
Dense 3 (Dense)	240	(1, 120)
Dense 4 (Dense)	8470	(1, 70)
Dense 5 (Dense)	4970	(1, 70)
Output data (Dense)	71	(1, 150)
The output data has a total number of parameters: 13,751		
Non-trainable parameters: 0		

The reconstructed compressing data audio is implemented in the decoded part of the autoencoder and genereted at the output layer is shown in the Fig. 13.

It is clearly remarkable that the reconstructed mask is similar to the original one after using the first convolutional autoencoder. A way to be sure of that, we implement in the Fig. 14 the two signals in one common figure to show the big similarities between them .

This first autoencoder has achieved a very big and interessting accuracy equal to 98%.

Now let's plot the loss function between the training and validation data to visualise the model performance of this model. The Fig. 15 demonstrates the resulted loss experiment. The signal result given after using the autoencoder is shown in the Fig. 15.

From the above plot, we can derive some intuition that this model is overfitting at some epochs while being in synchronised for most of the time. So, now is the time to definitely try to improve the performance of this model by introducing some complexity into it so that the loss can reduce more and this step will be using another autoencoder of type denoising one with other different parameters and the result that could be taken from this procedure will be described in the last of this paper. So, As usual we begin

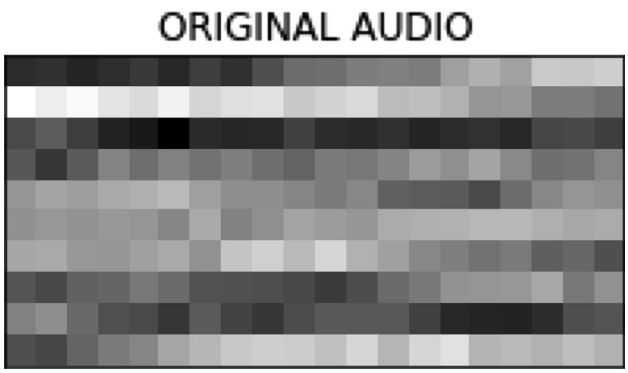

Fig. 11 Original mask of mixture audio file

Fig. 12 The encoding audio mask of the encoded part of the first autoencoder

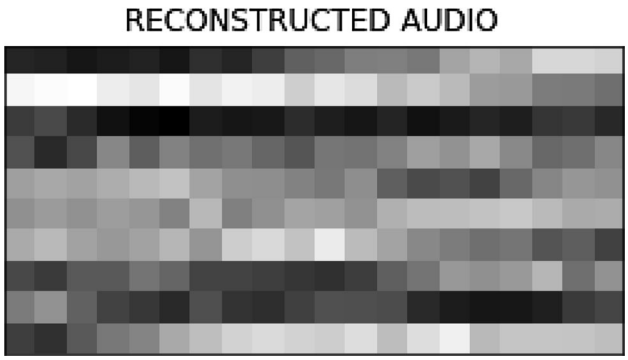

Fig. 13 The obtained reconstructed mask of audio mixture

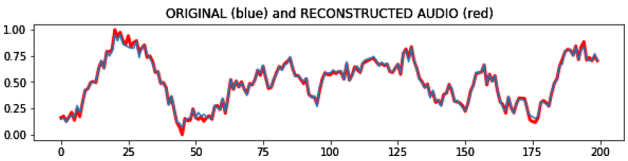

Fig. 14 The original signal VS the reconstructed signal

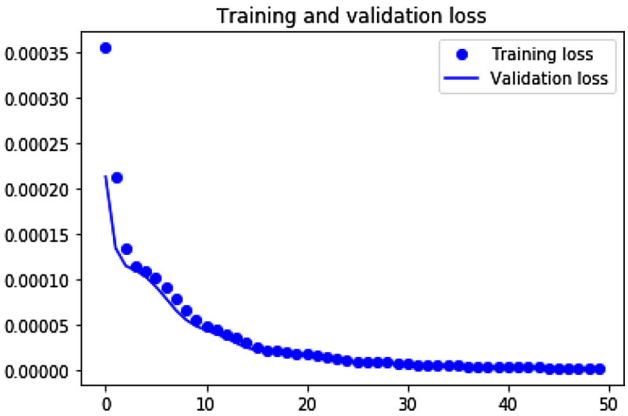

Fig. 15 Training and validation loss of the first autoencoder

by ploting the original mixture of audio signal after using the first input layer of the second denoising autoencoder. The input data feeded to the first layer of the second autoencoder is the output resulted of the first autoencoder. The resulted signal is shown in the Fig. 16.

The estimated encoding denoised signal is presented in the Fig. 17.

It is clear that after using the second denoising autoencoder , the noise representing with some useless samples has been delayed from the signal and we still have only the meaningful information. Also, we can remark that the denoising autoencoder has removed the nil samples that

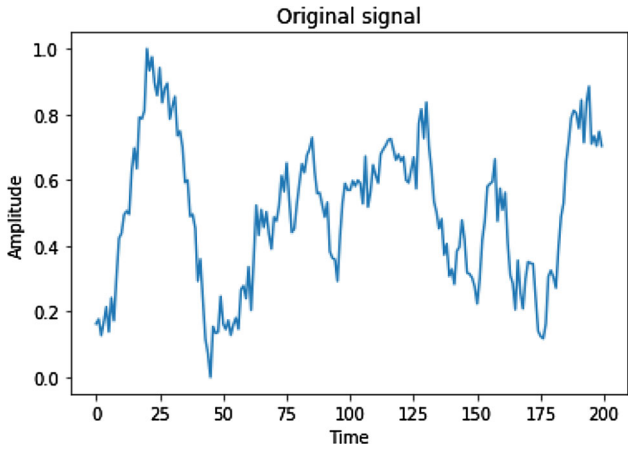

Fig. 16 Original signal used in the second denoising autoencoder

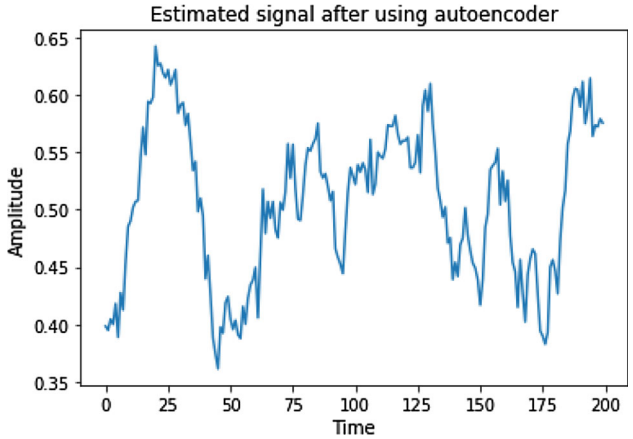

Fig. 17 Estimated denoised signal used in the second denoising autoencoder

may represent the silence in the audio file. To compare between the original signal of mixture and the new output obtained after this second operation of filtration, the Fig. 18 shows this difference.

The Fig. 18 shows a lot of similarities between the resulted two signals whitch means that the final signal has been well reconstructed. It is clear that there is the scale ambiguity seen in the blue signal which is generated

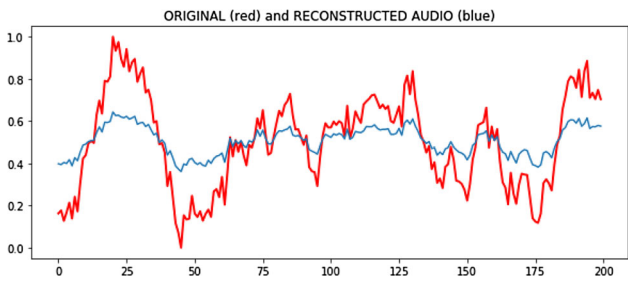

Fig. 18 Original signal VS the reconstructed signal after using two autoencoders

usually after the separation process of the BSS problem. In order to make sure about the performance of the approach, let's again visualize the loss function of our denoising autoencoder and compare it with the previous first loss. The Fig. 19 shows this result.

Finally, we can notice that the validation loss and the training loss both are in sync. It shows that our second model is not overfitting and this is because the validation loss is decreasing and not increasing, and there is a small rarely gap between training and validation loss. Also the total number of the trainable parameters have been reduced from 60,350 to 13,751, which is a very good thing to mention in reduction techniques. Therefore, we can demonstrate that our model's generalization capability is good. Although, this second model has given us the accuracy equal to 87%, which is less compared with the accuracy given by the first autoencoder, but we can ensure that we succeed with this beautiful combinaison of autoencoders to combine between two important things: the compression and the denoising of data with minimum loss as much possible as it is can be.

The experiment continues with the other mensioned data file of different types of mixtures, we follow the same procedure with all the rest of data. The result obtained is resumed in the following table (see Table 5)

We should notice first, that the number of total parameters obtained after the model training of the first and the second autoencoder is the same for all the data set used for this experiment.

As we can notice clearly, the comparaison accuracy between the different data file tested in this experiment study is not the same according to the first and the second autoencoder. The accuracy decreases from 93% for the convolutional autoencoder for the first 4 femele instantaneous mixture to 67% for the 3 male instantaneous mixture. And the same for the denoinsing autoencoder, it decreases from 87 to 72%.

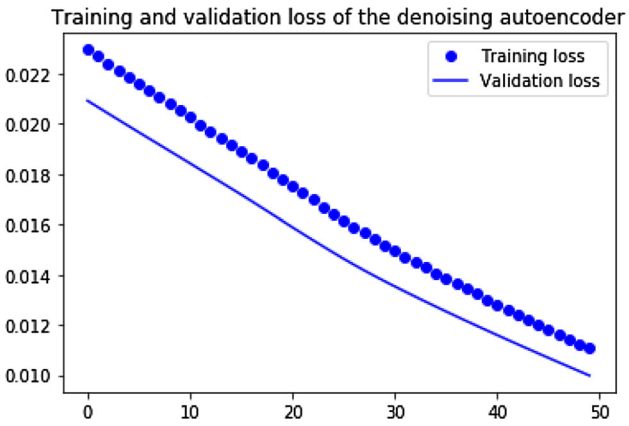

Fig. 19 Training and validation loss of the denoing autoencoder

Table 5 The accuracy comparaison between all data files used in the experiment

Data file	1st autoencoder accuraccy	2nd autoencoder accuracy
4 female inst. speech mixture	0.93	0.87
4 male inst. speech mixture	0.81	0.83
3 female inst. speech mixture	0.69	0.78
3 male inst. speech mixture	0.67	0.72

We conclude that the nature of the data can make a big difference for the accuracy computation result. Also we conclude that as much the number of the original sources is greater as much we obtain good performance accuracy. We have chosen to provide the result simulations figures of the female speech because it the one which has the greater performance accuracy that can be seen obviously in the Table 5. We have of course conducted over the others datasets the same experiments which were conducted on the 4 females speech instantaneous mixture.

6 Conclusion

To separate individual sources from a mixture of multiple sources in the presence of the noise is a very challenging and complex process to perform. A mixture between two different autoencoders for the audio source separation has been presented.

The code coefficients in the hidden layers of the autoencoder is used as a feature compressing filters for distinguishing audio sources after this step. An application of a denoising autoencoder to audio source separation is proposed to denoise corrupted versions of their inputs. The numpy arrays obtained by the last autoencoder analysis are used for the initial representation of the mixed source signal. The fundamental contribution of the proposed method is that the characteristics of the unknown sources are extracted from the mixed signals.

Acknowledgements Drs Reyes and Ventura want to acknowledge the economical support of the Spanish Ministry of Economy and Competitiveness and the Fund of Regional Development (Project TIN2017-83445-P).

References

1. Li, Y., Wang, F., Chen, Y., Cichocki, A., & Sejnowski, T. (2017). The effects of audiovisual inputs on solving the cocktail party problem in the human brain: An fmri study. *Cerebral Cortex*, *28*, 1–15.
2. Févotte, C., & Cardoso, J.-F. (2005). Maximum likelihood approach for blind audio source separation using time-frequency gaussian source models. In *IEEE workshop on applications of signal processing to audio and acoustics* (pp. 78–81). IEEE.
3. Duong, N. Q. K., Vincent, E., & Gribonval, R. (2010). Under-determined reverberant audio source separation using a full-rank spatial covariance model. *IEEE Transactions on Audio, Speech, and Language Processing*, *18*(7), 1830–1840.
4. Romano, J. M. T., Romis, A., Cavalcante, C. C., & Suyama, R. (2016). *Unsupervised signal processing: Channel equalization and source separation*. Boca Raton: CRC Press.
5. Zhang, R., Zhu, J.-Y., Isola, P., Geng, X., Lin, A. S., Yu, T., et al. (2017). Real-time user-guided image colorization with learned deep priors. arXiv preprint arXiv:1705.02999.
6. Chandna, P., Miron, M., Janer, J., & Gómez, E. (2017). Monoaural audio source separation using deep convolutional neural networks. In *International conference on latent variable analysis and signal separation* (pp. 258–266). Springer.
7. Dubey, N., & Mehra, R. (2015). Blind audio source separation (bass): An unsupervised approach. *International Journal of Electrical and Electronics Engineering*, *2*, 29–33.
8. Zhao, M., Wang, D., Zhang, Z., & Zhang, X. (2015). Music removal by convolutional denoising autoencoder in speech recognition. In *2015 Asia-Pacific signal and information processing association annual summit and conference (APSIPA)* (pp. 338–341). IEEE.
9. Katsamanis, A., Black, M., Georgiou, P. G., Goldstein, L., & Narayanan, S. (2011). Sailalign: Robust long speech-text alignment. In *Proceedings of workshop on new tools and methods for very-large scale phonetics research*.
10. Houda, A., & Otman, C. (2015). Blind audio source separation: State-of-art. *International Journal of Computer Applications*, *130*(4), 1–6.
11. Houda, A., & Otman, C. (2017). A novel method based on gaussianity and sparsity for signal separation algorithms. *International Journal of Electrical and Computer Engineering (IJECE)*, *7*(4), 1906–1914.
12. Kim, E., Hannan, D., & Kenyon, G. (2017). Deep sparse coding for invariant multimodal halle berry neurons. arXiv preprint arXiv:1711.07998.
13. Middlebrooks, J. C., & Simon, J. Z. (2017). Ear and brain mechanisms for parsing the auditory scene. In *The Auditory System at the Cocktail Party* (pp. 1–6). Springer.
14. Saruwatari, H., Kurita, S., Takeda, K., Itakura, F., Nishikawa, T., & Shikano, K. (2003). Blind source separation combining independent component analysis and beamforming. *EURASIP Journal on Advances in Signal Processing*, *2003*(11), 569270.
15. Leglaive, S., Badeau, R., & Richard, G. (2017). Separating time-frequency sources from time-domain convolutive mixtures using non-negative matrix factorization. In *2017 IEEE workshop on applications of signal processing to audio and acoustics (WASPAA)* (pp 264–268). IEEE.
16. Jang, G., Kim, H.-G., & Oh, Y.-H. (2014). Audio source separation using a deep autoencoder. arXiv preprint arXiv:1412.7193.
17. Wang, D., & Chen, J. (2018). Supervised speech separation based on deep learning: An overview. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *975*, 8887.
18. Abouzid, H, & Chakkor, O. (2017). Blind source separation approach for audio signals based on support vector machine classification. In *Proceedings of the 2nd international conference on computing and wireless communication systems* (p. 39). ACM.

19. Reynolds, D. A., Quatieri, T. F., & Dunn, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, *10*(1–3), 19–41.
20. Pawar, R. V., Jalnekar, R. M., & Chitode, J. S. (2018). Review of various stages in speaker recognition system, performance measures and recognition toolkits. *Analog Integrated Circuits and Signal Processing*, *94*(2), 247–257.
21. Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. B. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, *13*(5), 1035–1047.
22. Mary, L. (2011). *Extraction and representation of prosody for speaker, speech and language recognition*. Berlin: Springer.
23. Degara-Quintela, N., Pena, A., Sobreira-Seoane, M., & Torres-Guijarro, S. Knowledge-based onset detection in musical applications.
24. Dannenberg, R. B. (1984). An on-line algorithm for real-time accompaniment. In *ICMC* (Vol. 84, pp. 193–198).
25. Sarroff, A. M, & Casey, M. A. (2014). Musical audio synthesis using autoencoding neural nets. In *ICMC*.
26. Abouzid, H., & Chakkor, O. (2018). Dimension reduction techniques for signal separation algorithms. In *International conference on big data, cloud and applications* (pp. 326–340). Springer.
27. Liutkus, A., Stöter, F.-R., Rafii, Z., Kitamura, D., Rivet, B., Ito, N., et al. (2017). The 2016 signal separation evaluation campaign. In *International conference on latent variable analysis and signal separation* (pp. 323–332). Springer.

Houda Abouzid was born in Mai 1989 in Tetouan, Morocco, she is a Ph.D. student at Abdelmalek Essaadi university, National School of Applied Sciences, Tetouan. She received her degree of Engineering of telecommunication systems and networks in 2013 at the same school. Since 2014, she has provided several vocational training courses on matlab programming, networks and embedded systems. She is also a teacher vacant at the National School of Applied Sciences (ENSA-Te). She is the president and member of creation of a telecom generation club at ENSA-Te since 2016, providing different educational courses and workshops dedicated to all PhD students of Abdelmalek Essaadi university. She is interested on signal processing, Neural network and machine learning techniques.

Dr. Otman Chakkor was born in November 1977 in Tangier, Morocco. He is a Professor (Head of Telecommunication Departement) at Abdelmalek Essaadi University, in National School of Applied Sciences ENSA in Tetuan, Morocco and member of the RSAID Laboratory. He received his Ph.D.-Thesis from the University of Granada (Spain), Department of Architecture and Technologies of Computers; Area: Signal processing, He obtained his degree of Electric Engineering IEEA from Abdelmalek Essaadi University (Morocco) in 1999.

Dr. Oscar Gabriel Reyes received the B.S. and M.Sc. degrees in Computer Science from the University of Holguin (Cuba), in 2008 and 2011, respectively. He obtained the Ph.D. degree in Computer Science from the University of Cordoba (Spain) in 2016. He is currently a researcher at the Knowledge Discovery and Intelligent Systems Research Laboratory of University of Cordoba, Spain. Dr. Reyes has published several papers in high-impact journals, and he has also been engaged in 2 national research projects. His current research interests are in the fields of data mining, deep learning, and their applications.

Dr. Sebastian Ventura is currently a Full Professor in the Department of Computer Science and Numerical Analysis at the University of Cordoba, where he heads the Knowledge Discovery and Intelligent Systems Research Laboratory. He received his B.Sc. and Ph.D. degrees in sciences from the University of Cordoba, Spain, in 1989 and 1996, respectively. He has published more than 150 papers in journals and scientific conferences, and he has edited three books and several special issues in international journals. He has also been engaged in 12 research projects (being the coordinator of four of them) supported by the Spanish and Andalusian governments and the European Union. His main research interests are in the fields of soft-computing, machine learning, data mining, and their applications. Dr. Ventura is a senior member of the IEEE Computer, the IEEE Computational Intelligence and the IEEE Systems, Man and Cybernetics Societies, as well as the Association of Computing Machinery (ACM).