Signal Speech Decompression using Convolutional Denoising Audioencoders for Unsupervised Machine Learning Algorithms

Houda Abouzid[1] **and Otman Chakkor**[2]

[1]Department of telecommunications and computer science
Abdelmalek Essaadi University
National School of Applied Sciences
Tetouan, Morocco
abzdhouda@gmail.com

[2]Department of telecommunications and computer science
Abdelmalek Essaadi University
National School of Applied Sciences
Tetouan, Morocco
otman.chakkor@uae.ac.ma

ABSTRACT

In signal procesing, audio data compression refers to the encoding information using less bits than the original representation. This method will be identified in our theoretic approach and applied for the blind source separation (BSS) problem. In this paper, we will mix and match between two types of autoencoders which are Convolutional and Denoising autoencoders. The implementation uses Keras as a principal library of neural deep learning, in order to use this resulted signals after being analysed in blind source speech separation system. We suggest the mixture of those two types of autoencoders for unsupervised learning model that reconstructs audio mixture of speech signal based on Neural Network and deep learning.
Simulation results have proven that this mixture of autoencoders can make BSS easier to study and yield more performance for the signals to be included in the automated system. The advantage of this work is the originality of the solution, given an accuracy of 81% applied on real speech signals.

Keywords: BSS, Denoising autoencoder, Convolutional autoencoder, unsupervised learning, Deep Neural Network (DNN), compression, audio signal.

Computing Classification System (CCS): Computer systems organization $\rightarrow$ Neural networks, Information systems $\rightarrow$ Data compression, Hardware $\rightarrow$ Signal processing systems.
2000 Mathematics Subject Classification: 68P05, 68P30, 68T10.

1 Introduction

Blind source separation or (BSS) means the tasks of estimating individual sound signals from a mixture of multiple audio sources, this operation is done without the aid of information about the characteristic of the original sources. In the majority of the studied cases, this problem is considered complex and hard to solve due to the ignorance of such useful information such as the number of existing signals in the observed mixture , their natures, the statistical properties, etc. We speak about *blind source separation* when the estimated sources are recovered by unsupervised methods.
Audio source separation is a solution to recover a set of sound signals from their observed mixtures keeping only one clue which is supposing that the sources are statistically independent. This interesting technique has always drawn attention of many researchers to study this problem and especially during the last decades (Makino, 2018). Many of them have found different solutions according to the chosen situation of environment (noisy or without noise) and to the conditions of the experiment. The most popular approaches that have been used recently include informax, maximum likelihood estimation, negentropy maximization, and nonlinear PCA (Principal Component Analysis)(Abouzid and Chakkor, 2018). Those approaches are used in the famous method called as ICA (Independent Component Analysis)(Houda and Otman, 2017).

Different methods have been used to solve the source separation problem and most of them must apply the appropriate data pre-processing steps and features extraction techniques , so that the dataset will be amenable for machine learning. When we automate the behavior of the machine by creating and developing models, we enable the user to implement machine learning solutions with ease and frees up the scientific researchers in signal processing field. The most applicable machine learning algorithms are focusing on complex problems dealing with BSS task. Those approches are divided into unsupervised learning or supervised learning (Romano, Attux, Cavalcante and Suyama, 2016).

The first method used for finding the roots of equations with the presence of noise measurements was the Stochastic Approximation (SA) based on a simultaneous perturbations (Spall et al., 1992). This algorithm has broven to be efficient more than the approximation of Kiefer-Wolfowitz in solving many problems. An other work related to control systems, the one cited in (Preitl, Precup, Fodor and Bede, 2006) that proposes a new design method for Iterative Feedback Tuning algorithms employed in the design of a class of fuzzy control systems with Mamdani-type PI-fuzzy controllers. The objectif of combining IFT algorithms with fuzzy control is to have at the end of the operation a low cost fuzzy control solutions with good performance. The algorithm converges well with a stable analysis apprach Popov's hyperstability theory. This method is applied to servo systems like mechatronics systems and embedded systems. Another problem that arises in non parametric identification that deal with non linear systems with chaotic behavio (García, Luviano-Juárez, Chairez, Poznyak and Poznyak, 2011). In this paper, a Projectional Dynamic Neural Network (PDNN) theory is applied to accomplish the identification mission. This approach has also proven its efficiency with acceptable conver-

gence with the identified and measured variables and even with some noise perturbation. In (Yacoub, Bambang, Harsoyo and Sarwono, 2014), the authors presented their proposed structure by combining two filters a finite-impulse response filter and a functional-link neural network (CFFLNN) to improve the attenuation performance that appears in real word applications for Active Noise Control (ANC). In this work, an adaptative learning algorithm has been developed to identify the ANC secondary-path and for the active noise control process. The algorithm is tested on digital signal processors DSP TMS320C6713 DSK and performed for real-time experiments in ANC system.
In our work, we combine between two types of autoencoders, the first one is a convolutional autoencoder used for compression task only , while the second one is a denoising autoencoder used for compression data as well with different parametres and number of layers and also used for the suppression of noise that will be taking from the mixture speech signal. Our approach is computationally efficient and can be run in real time. Our proposed neural network structure is using convolutional architechture for signal processing. In this paper, we are truying to demonstrate the effectiveness of feedforward convolutional denoising model using the autoencoders on speech signal mixture.
This paper is organized as follows: The second section shows the most relevant related works as a part of the state of the art of some existing techniques solving the (BSS) problem. We are interested more on those which are unsupervised methods. The methodology is then described in section 3. The main result implementations and their explications are presented in the section 4. In last section, we conclude and give some perspectives.

2 Comparison with State-of-the-Art

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 which 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*, *Pandas* and many others. The most important advantages of using *Keras* is that on first hand, it supports both convolutional and recurrent networks or the combinaison of the two, and it could be running seamlessly on CPU and GPU on the second hand.
Audio applications: many applications could apply our approach such as in compression audio data, telephony, speech recognition, audio synthesis and speech enhancement, etc.

2.1 Problem formulation for BSS

The Blind source separation formulation is described as follow: The number of channels is denoted as I and J is the number of sources, $x(t) \in \Re^{I \times 1}$ is the observed I-channel mixture signal, $c_j(t) \in \Re^{I \times 1}$ is the I-channel spatial image of source j. The mixtures and the the

sources are both related by:

$$x(t) = \sum_{j=1}^{J} c_j(t) + n(t), \tag{2.1}$$

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 $x(t)$.

2.2 Unsupervised methods

Unsupervised leaning are methods which separate and learn the structure of sources in the mixtures based on some hypothesis that help to solve the BSS problem instead of using their sophisticated characteristics that are supposed to be unknown, or human auditory perception as well (Romano et al., 2016). So the idea here is to seek to describe axes of variation or clusters of behavioral patterns with as few a priori assumptions as possible. In unsupervised learning there are no labeled outputs. There are some famous algorithms that are used for this purpose like the Independent Component Analysis (ICA)(Houda and Otman, 2015), Sparse coding(Kim, Hannan and Kenyon, 2017), Computational Auditory Stream Analysis (CASA)(Middlebrooks and Simon, 2017), Beamforming(Saruwatari, Kurita, Takeda, Itakura, Nishikawa and Shikano, 2003) and Non-negative matrix factorization (NMF)(Leglaive, Badeau and Richard, 2017).

For exemple, in (Jang, Kim and Oh, 2014), the author proposed a new framework for unsupervised source separation using a deep autoencoder. The author explored the unknown characteristics of the signals in the mixed input sources by extracting the features using the auto encoder implemented by a neural network with multi hidden layers and then he took those coefficients and classified them on clusters for the separation task.

In (Makhzani, 2018), the author studied the learning of unsupervised representations using autoencoders by exploiting the sparsity property to represent the autoencoders and then proposed sparse autoencoders that can learn sparse representations of the image datasets. Aften that, the author proposed generative autoencoders that use a generative adversarial network (GAN)[1] to correspond the distribution of the latent code of the autoencoder with a pre-defined prior.

The most relevant algorithms for unsupervised learning are: k-means clustering, principal component analysis and autoencoders. In this paper, we are dealing with the autoencoders. Since there are no labels provided, there is no specific way to compare model performance in most unsupervised learning methods. In our case, we will use the accurracy function to compute the performance of the proposed architecture. The main goal of using the autoencoder is first, to do the analysis for the provided data and then identify its structure. Secondly, for dimensionality reduction to make further data processing much less intensive, and eliminate redundant

[1] Generative Adversarial Networks

features.

2.3 Supervised methods

In Supervised methods, as the name indicates a presence of supervisor as a teacher. Basically the supervised methods are algorithms that learn from labeled data which means some data is already tagged with correct answer. After analyzing and understanding the data, the algorithm should be able to determine which label will be given to new data based on pattern and associating the patterns to the unlabeled new data. So when the machine is provided with new set of examples(data), the supervised learning algorithm will produces an correct outcome from labeled data. . In Source separation task, the term supervised learning refers to machine learning algorithms that try to solve the separation problem having a prior knowledge of what the output values should be. The major purpose is to learn a function that can make best approximate predictions. These predictions represent the estimated signals that should be similar as much as possible to the original signals. The training data contains the input data as representing mixtures and the output data referring to their responses values (Wang and Chen, 2018).
To construct predictions, a model is fundamental to train the process of estimation. This is done using some weights that are recalculated each time when the result of the prediction is so far from the output training data or wrong. This process will be iterated each time until getting better accuracy of the model. The supervised learning is divided into two main categories: classification and regression. Supervised learning methods include several algorithms taht are used for many tasks like : Classification , Support Vector Machine (SVM)(Abouzid and Chakkor, 2017), logistic regression, naive bayes, artificial neural networks, random forests, Gaussian mixture model (GMM)(Reynolds, Quatieri and Dunn, 2000) and Anomaly detection.
The fig 1 resume the main differences between the unsupervised and supervised learnings.

2.4 Blind source separation

Speech is the most basic and easier tool of communication for interaction between human beings, and nowadays it has become also a tool for human-machine interaction. As we are always surrounded by noise and other sounds mixed and coming together from different directions , the human mind has always the ability to focus on one single channel and ignore the others as a disturbing background noise. Unfortunately, this powerful ability is absent for machines or humanoid robot. For exemple if one put a robot in a noisy environment as what is happen in the cocktail party problem (Li, Wang, Chen, Cichocki and Sejnowski, 2017), and try to give it an order or just speaks with it, this robot will not be able to interpret what was saying to it and then it is not going to do this order. To solve this issue, a machine must first of all separate the arrived signals comming to it michrophones as a mixture of those signals with their delays and reverberation due to the walls in the place.
The great importance that has been givin to study and solve the source separation problem has begun since 1980 and it is still attracting many researchers in different fields of science until nowadays. While being an interesting problem in itself, the BSS is being considered as a

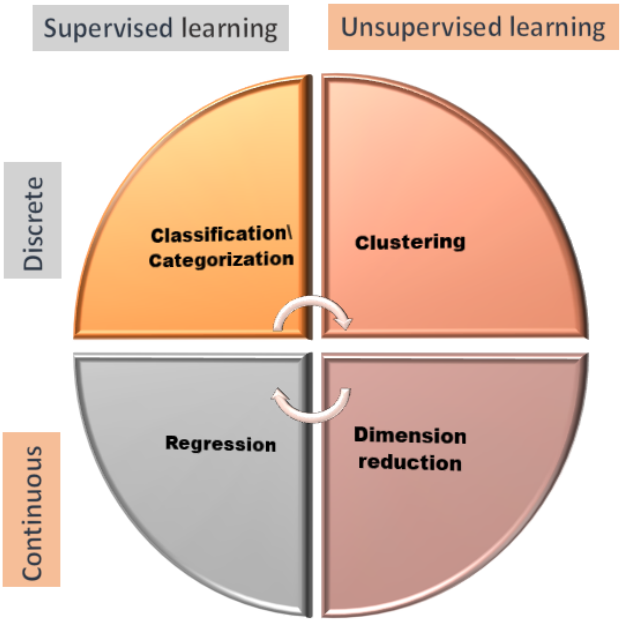

Figure 1: Comparaison between unsupervised and supervised learning

intermediary step for other applications such as automatic speech recognition and frequency features estimation (Chandna, Miron, Janer and Gómez, 2017) that can be used for exemple in the analyse study of neuronal activities and brain image. In addition to that, between the most fundamental application that mention the BSS problem is the removing noise from mixtures that helps a lot to improve the validation quality of the separation in a noisy environment (**?**).

3 Unsupervised learning based on autoencoder

3.1 Deep autoencoder for BSS

An autoencoder is an unsupervised learning algorithm that uses backpropagation, that try to reconstruct the output values to be equal to the inputs. I.e., it uses $y(i) = x(i)$. Deep autoencoders are networks able to compress and decompress the input data, and reconstruct again the approximation of the input as much perfect as possible, thus is, after reducing the dimensionality in the hidden layers.This is will be achieved by using a multilayer neural network, where the hidden layers decrease in size. Appliying a deep structure with multiple hidden layers proves its optimization for the classification problems and regression tasks (Sarroff and Casey, 2014).

In general, an autoencoder has three layers: an input layer, a hidden (encoding) layer, and a decoding layer. The network architecture is trained to generate the output that should be the maximum similar to the input, which forces the hidden layer to try to learn good representations of the inputs.
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

raw input data obtained from audio file experiment. Autoencoders are quite similar to PCA but they are much more flexible than PCA. This last is used only in linear transformation, however the autoencoder can be used on both linear and non-linear transformations.

First of all, and before starting the implementation we should analyze the audio sequence and the first step will be the onset detection and localization.

3.2 Onset detection pre-processing

Onset detection is a primary step to do for analyzing and indexing audio signals. The usual way to detect onsets is to look for ”transient” regions in the signal (Degara-Quintela, Pena, Sobreira-Seoane and Torres-Guijarro, n.d.). Which means the change in the short-time spectrum of the signal or in the statistical properties. The process of the onset detection is to determine the beginning of the transient part of the audio to find the meaningful sound events.
The automatic detection of events of audio signals include many applications like the automatic score followers(Dannenberg, 1984), content delivery, compression, indexing and retrieval. The onset detection property can be applied on different signals (speech, music, environmental sounds) (Bello, Daudet, Abdallah, Duxbury, Davies and Sandler, 2005). To make clear the idea about onset detection, we should define some new notions.

- **Transient**: For acoustic signals, the transient useally corresponds to the time period during which the excitation, is applied and then damped, leaving only the slow decay at the resonance frequencies of the body.
- **Onset**:is a notion which is always related to the transient. It can be define as a single instant chosen to mark the temporally extended transient. In most cases, it will coincide with the beginning of the transient, or the earliest moment at which the transient can be reliably detected.

Actually, automatic detection of events in audio signals has achieved a remarquable importance for sound source separation problem, music information retrieval and automatic music transcription as well.

Basicly, to detect the onsets a calculation of signal features should be done to 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.

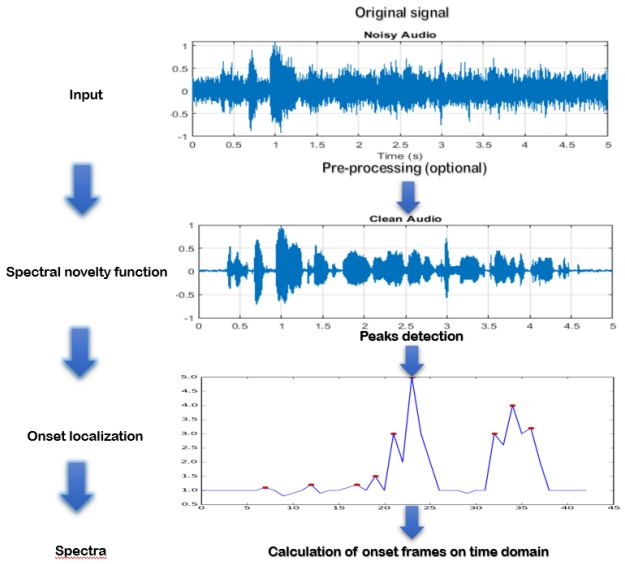

Figure 2: The architecture of onset detection for an audio signal

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:

$$\frac{d(n)}{mean\{d(n-M_l),...,d(n+M_u)\}} \geq \lambda \tag{3.1}$$

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.
Knowing that there are many practical application 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 160000 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 purpose of the encoder will be the filtering of the existing noise using the hidden units of the neural networks for the auto encoder. Before using the mixture of the two autoencoders, we begin by the onset detection operation to detect the transient of the signal and then transform the obtained onset frames to time domain. The idea behind the proposed architecture is 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.

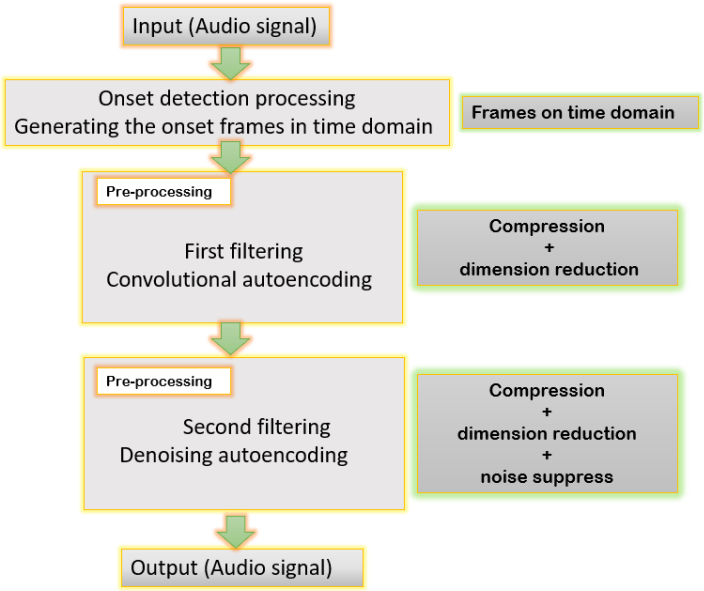

Figure 3: The structure of the auto encoder using neural network

3.3 Underlying idea using a denoising autoencoder

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.

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

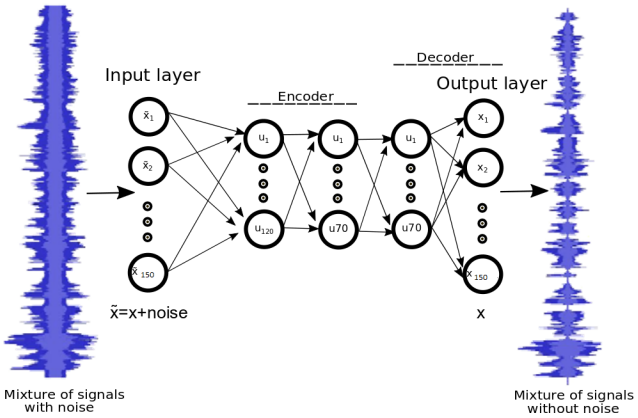

Figure 4: The proposed architecture of a denoising auto-encoder (DAE) using neural network: this figure has been taken from the second part of this work in (Abouzid, Chakkor, Reyes and Ventura, 2019)

The model first process if to detect the beginning of the transient part of the audio, then performs the pre-processing as a first stage of the training using the fist autoencoder which is the convolutional autoencoder (CAE). The CAE decompresses the data and reconstruct it with a very good accuracy. Next, the resulted signal will be used as an input for the second autoencoder. The denoising autoencoder takes this data and uses to the model to finetuned it by the backpropagation algorithm when it reaches the output layer. The hall algorithm stops when the trained model reaches convergence. Here we proposed the epochs equal to 50. This model is implemented using keras and python code.
The proposed structure of our denoising autoencoder has five layers defined as folow: The first layer is the input layer which contains 160000 samples 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 160000 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:

• Encoder : in this part, the network compresses the input samples into a fewer number of bits. Those compressed bits represent the original input and called the ”encoding” of the input.
• Decoder : 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 Main Result: experimental study

4.1 Data set

The data set that has been used for testing the new approach has been taken from the web site of *inria*[2] 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. About the room dimensions, they are the same for synthetic convolutive mixtures at *SiSEC2015* and live recordings (4.45 x 3.55 x 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.

About the reason of choosing our source of dataset from *inria* especially, is that it is the French Institute for Research in Computer Science and Automation since 1967 and it is the Public Scientific and Technical Research Establishment (EPST) under the double supervision of the French Ministry of National Education, Advanced Instruction and Research and the Ministry of Economy, Finance and Industry. Also the dataset provided is devided in different categories such as (Under-determined, determined and overdetermined) speech and music mixtures, so the one can choose between these categories according to his preferences. In addition to that, this dataset has been always tested, evaluated and updated before being uploaded in the inria web site and be ready for the public use.

4.2 Experiment settings

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

For our test, we choose the mixtures of 4 female speech sources.
About the source directions of arrival, they vary between -60 degrees and +60 degrees with a minimal spacing of 15 degrees 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 (Liutkus, Stöter, Rafii, Kitamura, Rivet, Ito, Ono and Fontecave, 2017), which were imported in Python by using the framework Keras (Keras was used to implement the autoencoder) and other signal processing libraries.

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

4.3 Experiments and discussion

After loading the audio file witch contains the audio mixtures of 4 females speaking at the same time into a numpy array of 160000 samples, we obtain the Fig. 5.
The sampling rate of this signal is equal to 22050 Hz.

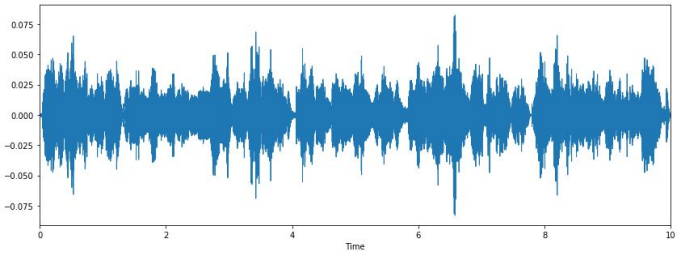

Figure 5: Audio wave of the sound mixture of 4 females speach

Now let's plot the onsets on the top of the spectrogram that we have obtained of our mixture audio. After converting the onsets to units of seconds, it shows that at each detected time

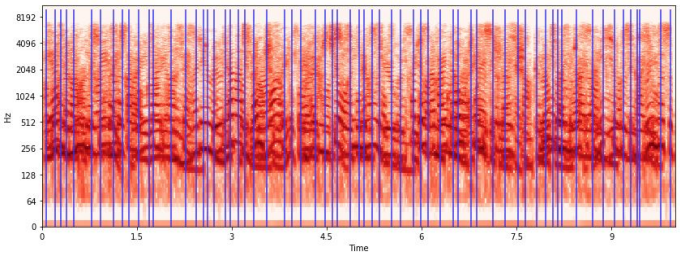

Figure 6: Onsets on the top of the spectrogram of audio signal

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. 6.

At the beginning, those onsets were as 66 frames 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 Tab. 1 shows the indices of those frames:

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				

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

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				

Now let's plot those onsets detection with the time domain waveform that is shown in the Fig. 7.

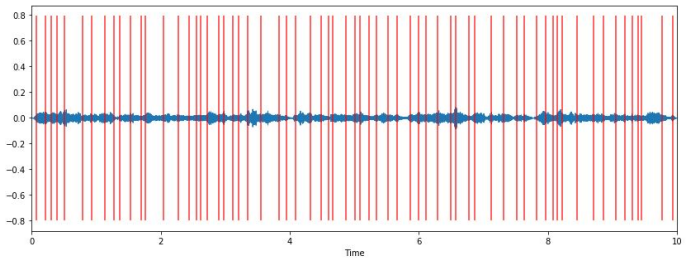

Figure 7: Waveform of the signal with their detectable onsets in time domain

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 (1,160000) dimension to an array , reshape it so it will be of size 200×1, which represent the encoded 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 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 Tab. 3. In the first experiment, we have employed the convolutional autoencoder

Table 3: The detail structure of the proposed denoising autoencoder

DAU model summary		
The input data with 66 frames and 22050 frequency bins		
Layer (type)	Filters number	Output shape
Input data (InputLayer)	0	(1, 150)
Dense 1 (Dense)	18120	(1, 120)
Dense 2 (Dense)	8470	(1, 70)
Dense 3 (Dense)	4970	(1, 70)
Output data (Dense)	10650	(1, 150)
The output data has a total number of parameters : 42,210		

that gives 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. 8. The encoding mask after

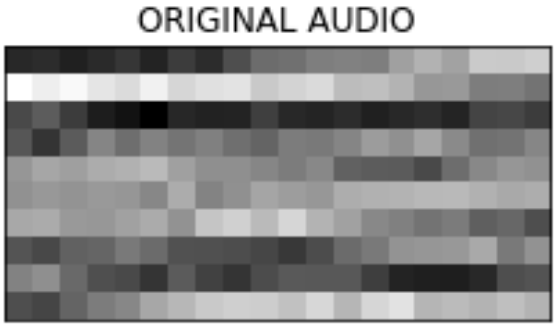

Figure 8: Original mask of mixture audio file

the compressing of the original file in the hidden layers in the middle of the implementation program is presented in the Fig. 9: The reconstructed compressing data audio is implemented

Figure 9: The encoding audio mask of the encoded part of the first autoencoder

in the decoded part of the autoencoder and generated at the output layer is shown in the Fig. 10.

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. 11 the two signals in one common figure to show the big similarities between them .

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

Now let's plot the loss function between the training and validation data to visualize the model performance of this model. The Fig. 12 demonstrates the resulted loss experiment. The signal

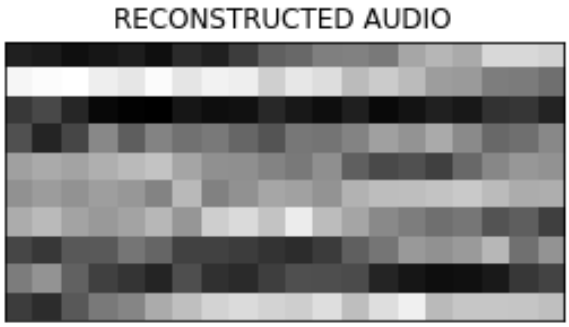

Figure 10: The obtained reconstructed mask of audio mixture

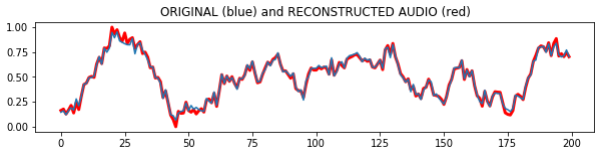

Figure 11: The original signal VS the reconstructed signal

result given after using the autoencoder is shown in the Fig. 12. From the above plot, we can

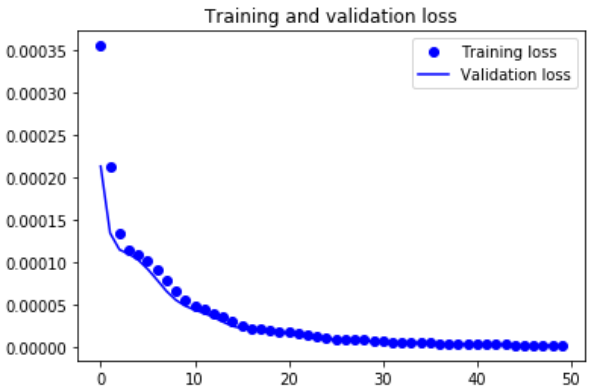

Figure 12: Training and validation loss of the first autoencoder

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 by ploting the original mixture of audio signal after using the first input layer of the second denoising autoencoder. The resulted signal is shown in the Fig. 13.

The Fig. 14 shows the original signals and its encoded version in the encoding part of the denoising autoencoder.

The estimated encoding denoised signal is presented in the Fig. 15.

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

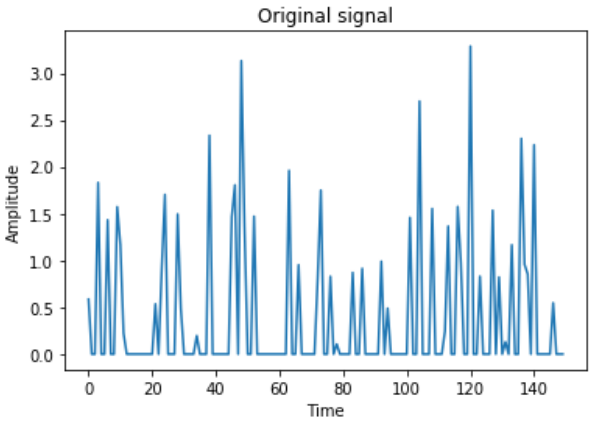

Figure 13: Original signal used in the second denoising autoencoder

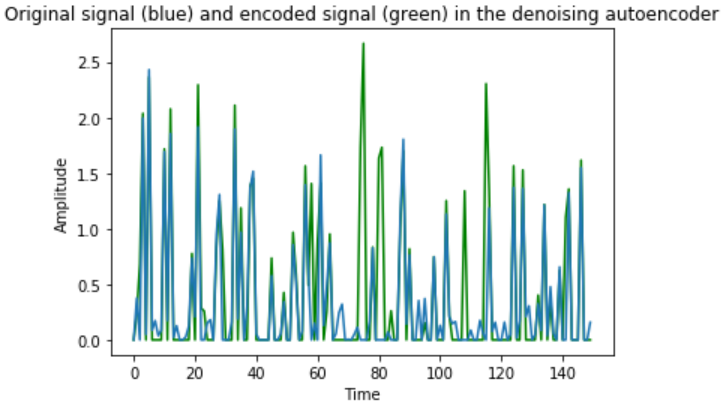

Figure 14: Original signal (blue) and its encoded data (green) in the denoising autoencoder

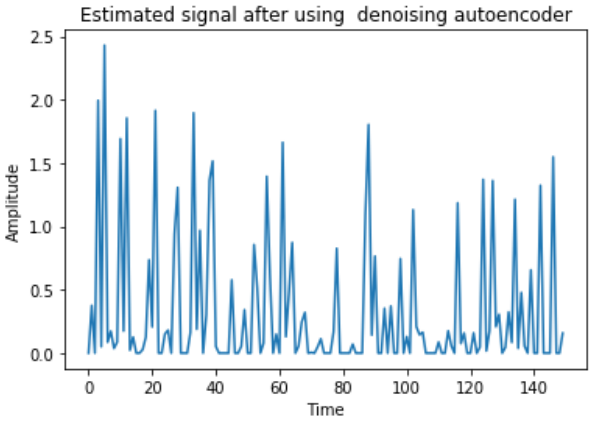

Figure 15: Estimated denoised signal used in the second denoising autoencoder

information. Also, we can remark that the denoising autoencoder has removed the nil samples that 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. 16 shows this difference:

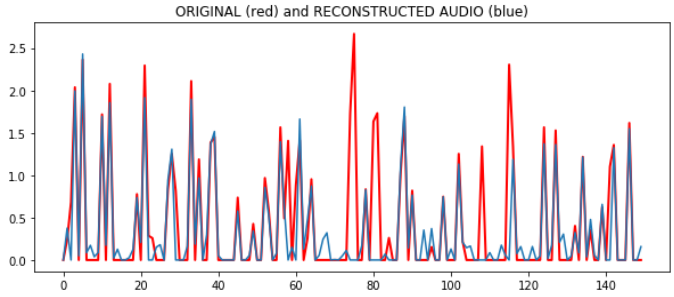

Figure 16: Original signal VS the reconstructed signal after using two autoencoders

The Fig. 16 shows a lot of similarities between the resulted two signals whitch means that the final signal has been reconstructed well and to make sure let's again visualize the loss function of our denoising autoencoder and compare it with the previous first loss. The Fig. 17 shows this result. Finally, we can notice that the validation loss and the training loss both are

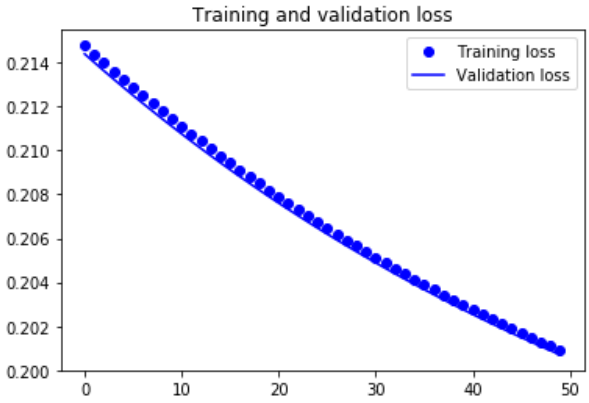

Figure 17: Training and validation loss of the denoing autoencoder

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. Therefore, we can demonstrate that our model's generalization capability is good. Although, this second model has given us the accuracy equal to 81%, 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.

5 Conclusion and futur work

Human and machines work collaboration is going to shape our current and future where both of the entities will be contributing to the larger cause of serving the new technology to be

performed on it best way. In this presented work, we have applied a mixture between two different autoencoders for speech signals.
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 and compress the signal for the second time. The numpy arrays obtained by the last autoencoder analysis is 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.
In the futur work we will study another type of autoencoder for unsupervised learning that uses sparse data. The sparse autoencoder is also an algorithm that automatically learns features from unlabeled data. Imposing the sparsity constraint on the hidden units, we will show that the autoencoder could discover interesting structure in the data, even if the number of hidden units is large.

References

Abouzid, H. and Chakkor, O. 2017. Blind source separation approach for audio signals based on support vector machine classification, *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems*, ACM, p. 39.

Abouzid, H. and Chakkor, O. 2018. Dimension reduction techniques for signal separation algorithms, *International Conference on Big Data, Cloud and Applications*, Springer, pp. 326–340.

Abouzid, H., Chakkor, O., Reyes, O. G. and Ventura, S. 2019. Signal speech reconstruction and noise removal using convolutional denoising audioencoders with neural deep learning, *Analog Integrated Circuits and Signal Processing* pp. 1–12.

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M. and Sandler, M. B. 2005. A tutorial on onset detection in music signals, *IEEE Transactions on speech and audio processing* **13**(5): 1035–1047.

Chandna, P., Miron, M., Janer, J. and Gómez, E. 2017. Monoaural audio source separation using deep convolutional neural networks, *International Conference on Latent Variable Analysis and Signal Separation*, Springer, pp. 258–266.

Dannenberg, R. B. 1984. An on-line algorithm for real-time accompaniment, *ICMC*, Vol. 84, pp. 193–198.

Degara-Quintela, N., Pena, A., Sobreira-Seoane, M. and Torres-Guijarro, S. n.d.. Knowledge-based onset detection in musical applications.

García, A., Luviano-Juárez, A., Chairez, I., Poznyak, A. and Poznyak, T. 2011. Projectional dynamic neural network identifier for chaotic systems: Application to chua's circuit, *Int. J. Artif. Intell* **6**(11): 1–18.

Houda, A. and Otman, C. 2015. Blind audio source separation: State-of-art, *International Journal of Computer Applications* **130**(4): 1–6.

Houda, A. and Otman, C. 2017. A novel method based on gaussianity and sparsity for signal separation algorithms., *International Journal of Electrical & Computer Engineering (2088-8708)* **7**(4).

Jang, G., Kim, H.-G. and Oh, Y.-H. 2014. Audio source separation using a deep autoencoder, *arXiv preprint arXiv:1412.7193* .

Kim, E., Hannan, D. and Kenyon, G. 2017. Deep sparse coding for invariant multimodal halle berry neurons, *arXiv preprint arXiv:1711.07998* .

Leglaive, S., Badeau, R. and Richard, G. 2017. Separating time-frequency sources from time-domain convolutive mixtures using non-negative matrix factorization, *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2017 IEEE Workshop on*, IEEE, pp. 264–268.

Li, Y., Wang, F., Chen, Y., Cichocki, A. and Sejnowski, T. 2017. The effects of audiovisual inputs on solving the cocktail party problem in the human brain: An fmri study, *Cerebral Cortex* pp. 1–15.

Liutkus, A., Stöter, F.-R., Rafii, Z., Kitamura, D., Rivet, B., Ito, N., Ono, N. and Fontecave, J. 2017. The 2016 signal separation evaluation campaign, *International Conference on Latent Variable Analysis and Signal Separation*, Springer, pp. 323–332.

Makhzani, A. 2018. *Unsupervised representation learning with autoencoders*, PhD thesis.

Makino, S. 2018. *Audio Source Separation*, Springer.

Middlebrooks, J. C. and Simon, J. Z. 2017. Ear and brain mechanisms for parsing the auditory scene, *The Auditory System at the Cocktail Party*, Springer, pp. 1–6.

Preitl, S., Precup, R.-E., Fodor, J. and Bede, B. 2006. Iterative feedback tuning in fuzzy control systems. theory and applications, *Acta Polytechnica Hungarica* **3**(3): 81–96.

Reynolds, D. A., Quatieri, T. F. and Dunn, R. B. 2000. Speaker verification using adapted gaussian mixture models, *Digital Signal Processing* **10**(1-3): 19–41.

Romano, J. M. T., Attux, R., Cavalcante, C. C. and Suyama, R. 2016. *Unsupervised signal processing: channel equalization and source separation*, CRC Press.

Sarroff, A. M. and Casey, M. A. 2014. Musical audio synthesis using autoencoding neural nets, *ICMC*.

Saruwatari, H., Kurita, S., Takeda, K., Itakura, F., Nishikawa, T. and Shikano, K. 2003. Blind source separation combining independent component analysis and beamforming, *EURASIP Journal on Advances in Signal Processing* **2003**(11): 569270.

Spall, J. C. et al. 1992. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation, *IEEE transactions on automatic control* **37**(3): 332–341.

Wang, D. and Chen, J. 2018. Supervised speech separation based on deep learning: An overview, *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **26**(10): 1702–1726.

Yacoub, R. R., Bambang, R. T., Harsoyo, A. and Sarwono, J. 2014. Dsp implementation of combined fir-functional link neural network for active noise control, *International Journal of Artificial Intelligence* **12**(1): 36–47.