Machine Learning & Big Data Processing

Voice isolation

Moulin Léo
Rouvroy Alexis
Schoone Rudy
Zhang Yudong

2017-2018

1 Introduction

The "cocktail party" effect is a fascinating phenomenon. How does the brain can focus on a particular audio stream in a noisy environment is a question that has been studied in neurosciences for many years. Recently, machine learning algorithms have become the state-of-art in solving this kind of problems.

This goal of this project consists of implementing a machine learning algorithm capable of separating the vocal part and the instrumental part of a song.

A convolutionnal neural network (CNN) has been trained with a supervised learning approach, using binary masks as ground truth. The parameters of the CNN have been derived from experimental results.

2 State of the art

Many algorithms isolating the voice of the accompaniment in a music exist. Their principles is often the same : display the spectrogram, compute a mask by using a Convolutional Neural Network (CNN) and finally apply this mask on the spectrogram. With this process we assign each time-frequency element to its respective source.

The difference between all these methods is the calculation of the algorithm allowing to compute the mask. For our project we used the technique called "U-net" and in this chapter we will rapidly browse some other existing methods.

The first one uses the Deep Neural Network (DNN) and F0 estimation [4]. It begins with training the DNN by using the Stochastic Gradient Descent (SGD) algorithm. We obtain a first estimated mask. We can then apply the YINFFT algorithm that will allow the identification of unvoiced speech regions. We just have to multiply both masks to improve the first estimation.

The second one uses the Deep Recurrent Neural Network (DRNN) [3] that introduces a memory from previous time steps. In this approach the training objectives are to minimize the Main Square Error and the generalized Kullback-Leibler (KL) divergence criteria :

$$J_{MSE} = ||\hat{y}_t - y_t||_2^2 \tag{1}$$

$$J_{KL} = D(\hat{y}_t||y_t) \tag{2}$$

The third one uses the Convolutionnal Denoising Autoencoders (CDAE) [2]. The voice is considered here as a signal and the rest as a noise to be removed, the CDAE is trained to minimize the cost function :

$$C_i = \sum_{n,f}(Z_i(n,f) - S_i(n,f))^2 \tag{3}$$

where Z_i is the actual output of the last layer of the CDAE of source i, S_i is the reference clean output signal for source i, n, and f are the time and frequency indices respectively.

To end this section we will quickly mention an issue of this study area. To create reliable algorithms in machine learning, we need huge datasets. But the amount of existing songs of high quality where we have access to the separated components of the song (voice and accompaniment) is relatively small. It forces us to make a choice between quality and quantity of the elements of the dataset.

3 Description of the algorithm

The basic principle of the algorithm is to first transform an input sound signal into a spectrogram using the Short Time Fourier Transform (STFT).

$$STFT(t, f) = \sum_{n=1}^{N} x(n)w(n-t)e^{-jfn} \tag{4}$$

$$Spectro(t, f) = |STFT(t, f)|^2 \tag{5}$$

The STFT of a signal has 2 components : a magnitude and a phase. As seen in Figure 1, the phase component contains no useful information. The difference between the phase of the unseparated and separated spectrogram is negligible. Thus, we will leave the phase untouched. Only the magnitude will be fed to the neural network.

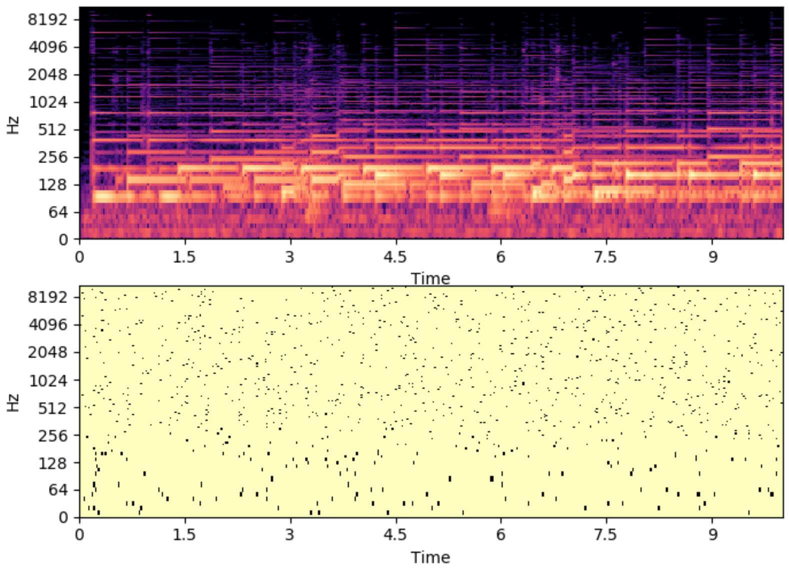


FIGURE 1 – Magnitude and Phase of the STFT of a 10s song

The neural network is trained using the spectrogram as input and the ideal binary mask as output. The binary mask gives information, on each frequency-time pixel, whether the pixel belongs to the voice or the instrumental part. The goal of the neural network is to be able to reconstruct the binary mask given a song spectrogram as testing data, as accurately as possible.

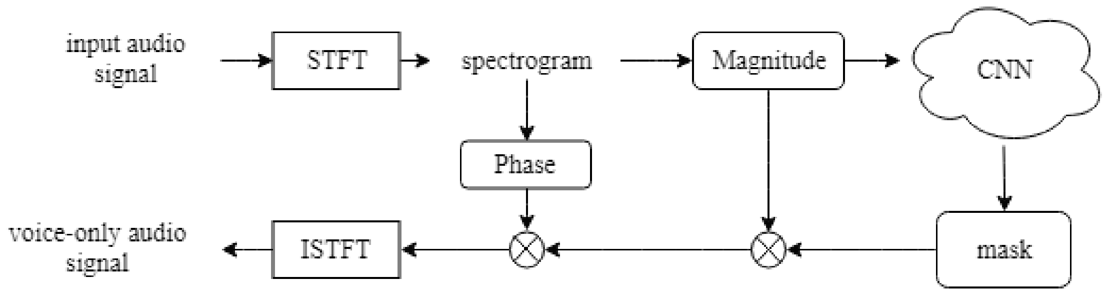


FIGURE 2 – Proposed algorithm

The mask estimated by the CNN is then multiplied element-wise with the original spectrogram to produce the vocal-only spectrogram. The final voice-only audio is constructed by taking the Inverse Short Time Fourier Transform (ISTFT).

The main advantages of working with a time-frequency representation are that the spectrogram can be seen as a 2D gray-level image, the rows being the frequency axis, the columns being the time axis and the intensity being the magnitude (power) of the spectrogram.

Algorithms involving images, for example image classification or image segmentation, can be used the same way with spectrograms.

3.1 Construction of the mask

A first step is to build a binary mask for each training song. This constructed mask will act as a ground truth during the training.

As the MedleyDB dataset [1] comes with several stems (sources) for each song, the vocal stems were first added up and normalized to make a vocal mix, and the instrumental stems were added and normalized to make an instrumental mix.

The ideal binary mask is constructed as :

$$\mathcal{M}_{bin}(i,j) = \begin{cases} 1 & \text{if } |\mathcal{S}_{vocal}(i,j)| \geq |\mathcal{S}_{instr}(i,j)| \\ 0 & \text{else} \end{cases} \tag{6}$$

where $i = 1...N$ is the time axis, $j = 1...M$ is the frequency axis, $\mathcal{S}_{vocal}(i,j)$ is the magnitude of the spectrogram of the vocal-only mix and $\mathcal{S}_{instru}(i,j)$ is the magnitude of the spectrogram of the instrumental mix.

A continuous mask was also considered, as in practice a time-frequency pixel is not all black (only instrument) or all white (only vocal). It can contain both components at the same time.

$$\mathcal{M}_{cont}(i,j) = \frac{|\mathcal{S}_{vocal}(i,j)|}{|\mathcal{S}_{instr}(i,j)| + |\mathcal{S}_{vocal}(i,j)|} \tag{7}$$

By experiments, we found that using a continuous mask doesn't improve the performance. Plus, we tested the 2 different masks on different songs and it turns out that the sound quality of the reconstructed vocal is similar with both masks. In the end, we opted for a binary mask, which is also easier to use with the neural network.

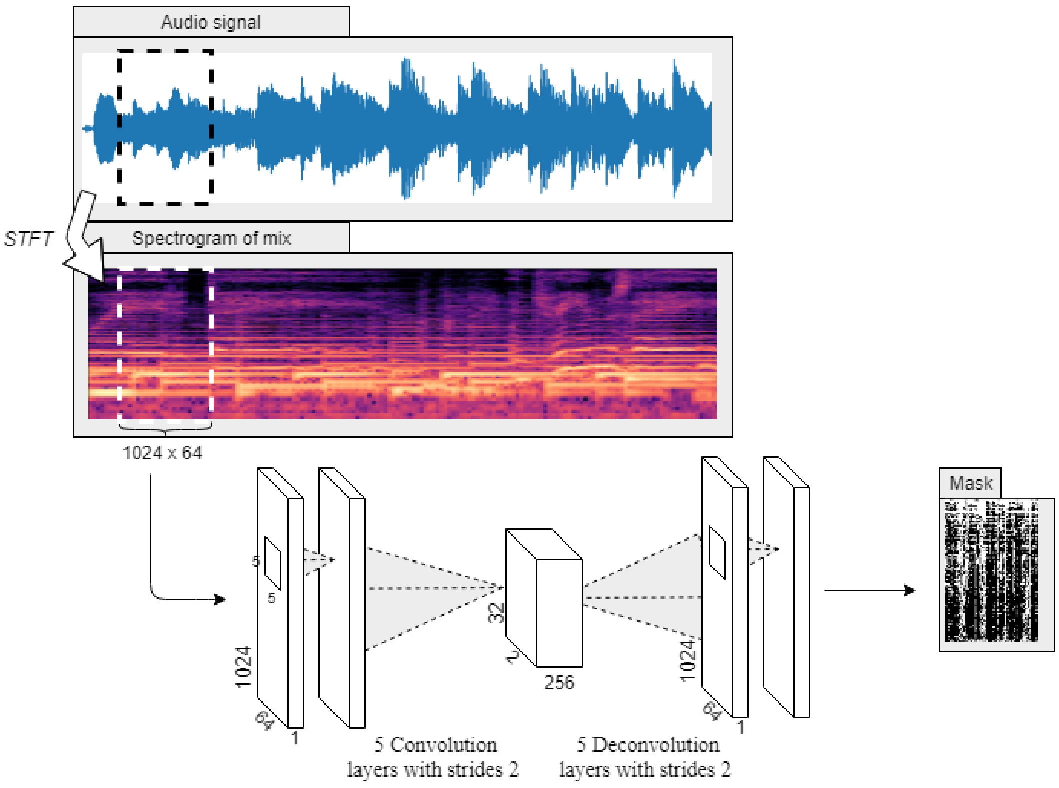
3.2 Neural Network

We opted for a Convolutionnal Neural Network, which the state of the art in pixel-wise segmentation or classification, as it takes into account the neighborhood of each pixel. It provides also a small and deep representation of the features.

The architecture of the CNN is pictured in Figure 3. The encoder part consists of 5 convolutional 2D layers with strides 2, each one of them use a kernel 5x5 and a leaky ReLu of 0.2 for the activation function. Each time the number of channel is multiplied by 2, the size of the data is divided by 2.

The decoder part consists of 5 Deconvolutional 2D layers (transposed convolution) with strides 2 and kernel 5x5. A dropout of 0.5 was used for the three first deconvolution layers. The decoder layers use a plain ReLu and for the last deconvolution layer a sigmoid activation function.

The original audio signal is divided into samples of 64 time steps and 1024 frequency steps. STFT is applied to each sample. The input shape of the CNN is thus a tensor with shape $(samples, 1024, 64, 1)$. The output of the CNN has the same shape as the input. After applying the ISTFT to each sample, all the samples are concatenated to form the final audio signal.



Audio signal
STFT
Spectrogram of mix
1024 x 64
5
5
1024
64
1
32
2
256
1024
64
1
5 Convolution layers with strides 2
5 Deconvolution layers with strides 2
Mask

FIGURE 3 – Architecture of the Neural Network

The loss function that was chosen is the Mean Squared Error :

$$L(\Theta) = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} ||y(i,j) - \hat{y}(i,j,\Theta)||^2}{N.M} \tag{8}$$

where $y(i,i)$ is the binary value of the ideal mask at time i and frequency j and $\hat{y}(i,j)$ is the value of the mask estimated with the CNN with parameter set Θ.

Several other loss functions were also considered, such as the Binary cross entropy or the mean absolute error but the MSE gave us the best results in term of accuracy.

4 Experimental results

4.1 Dataset

The dataset used in this project is the one that was suggested, namely the *MedleyDB* [1] dataset consisting of 122 songs. The original distribution of the genres is pictured in Figure 4 and the proportion of vocal/non-vocal songs in Figure 5.

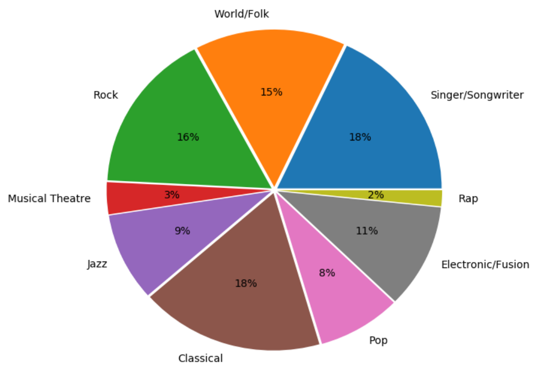


FIGURE 4 – Distribution of music genres of [1]

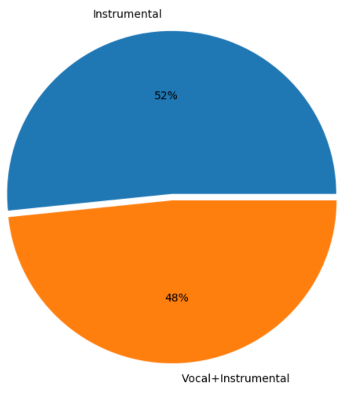


FIGURE 5 – Distribution of vocal songs of [1]

All the instrumental songs (not containing any vocal) were removed.

55 songs were selected for the training set, providing a high diversity of music genres. 9 songs (one of each genre) were selected for the testing set to measure the accuracy of the algorithm.

4.2 Results

The model was trained using the Adam optimizer with the mean squared error loss function for 20 epochs, with a batch size of 16.

As the output of the last layer of the CNN is a continuous mask, a threshold was applied in order to have a final binary mask :

$$y_{bin}(i,j) = \begin{cases} 1 & \text{if } \hat{y}(i,j) \geq \tau \\ 0 & \text{if } \hat{y}(i,j) < \tau \end{cases} \tag{9}$$

To verify the correctness of the implementation, the accuracy between the ground truth mask and the predicted mask with CNN was computed :

$$Accuracy = \frac{1}{N.M} \sum_{i=1}^{N} \sum_{j=1}^{M} (y_{bin}(i,j) \;\; XNOR \;\; \mathcal{M}_{bin}(i,j)) \tag{10}$$

4.2.1 5 Layers CNN

Using this accuracy measure, the final result of the algorithm is :

Accuracy on testing set	**68 %**
Accuracy on training set	81 %

FIGURE 6 – Final accuracy results

The accuracy on the training set is of 81% and lowers to 68% on the testing set. The continuous and binary mask computed on the testing data set can be observed in Figure 7.

A lot of similitude can be found between the true and predicted mask, but, they are still quite different and this difference can be easily heard in the final results. The threshold can also be changed to get a better accuracy as it can be seen in Figure 7. With a too low threshold, the instruments can still be heard and with a too high threshold, instruments are eliminated but so are some parts of the vocal. Finding the right threshold for the algorithm is crucial on the quality of the reconstructed audio. It can be observed that the best accuracy is obtained using a threshold of 0.7.

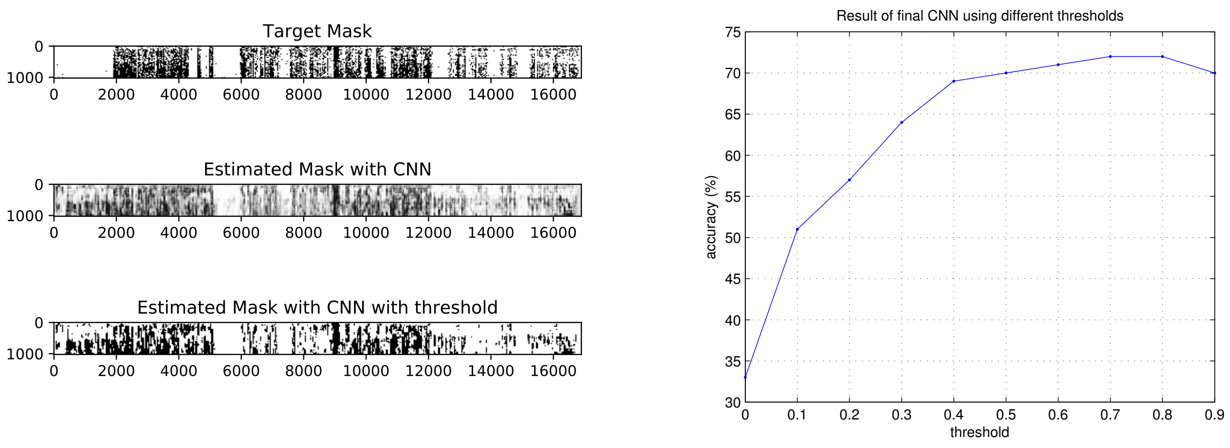


FIGURE 7 – Estimated masks with CNN after 20 epochs (a) and Measure accuracy on testing dataset in function of the threshold (b)

The minimization of the MSE cost function can be observed in Figure 8 as the number of epochs is growing, showing that the CNN is getting better at finding the right output. Arriving at 30 epoch, the loss function will not diminish any more, 20 epoch for the CNN seemed like a good middle ground.

The accuracy on the training data set will increase as the number of epoch grows, arriving at a maximum of 81%.

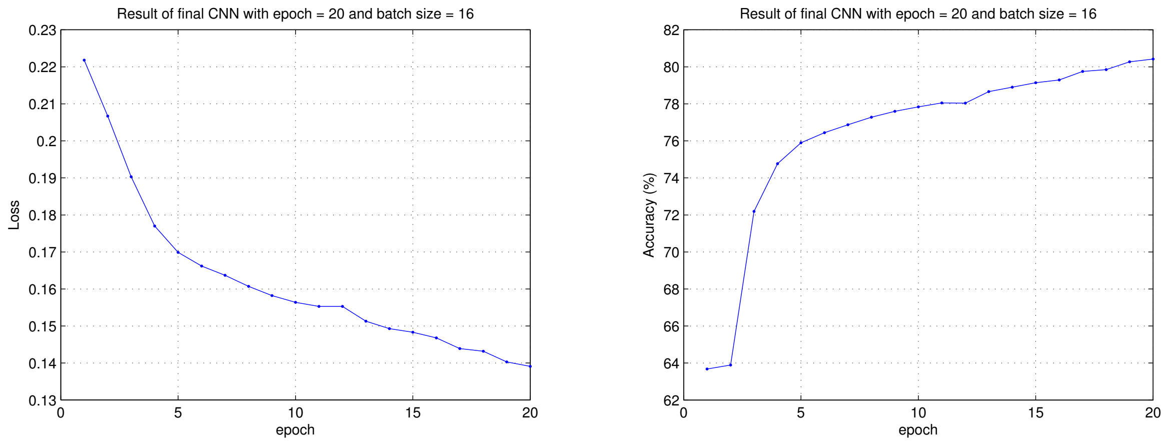


FIGURE 8 – Loss (a) and Accuracy (b) with final CNN in function of the training epochs

Modifications on the different parameters were used and each time the accuracy on the testing set was computed. To test out the importance of every parameter and to diminish the computing time, a smaller training set (1000 samples) and smaller testing set (500 samples) were used.

Firstly, the number of layers of the encoder and the decoder of the CNN was changed. It is quite observable on Figure 9 that the best number of layers is 2 and 3 layers, but by looking at the predicted mask for each cases, the one that looked the most similar was the ones using 3 and 5 layers. It was decided to keep using 5 layers in the CNN.

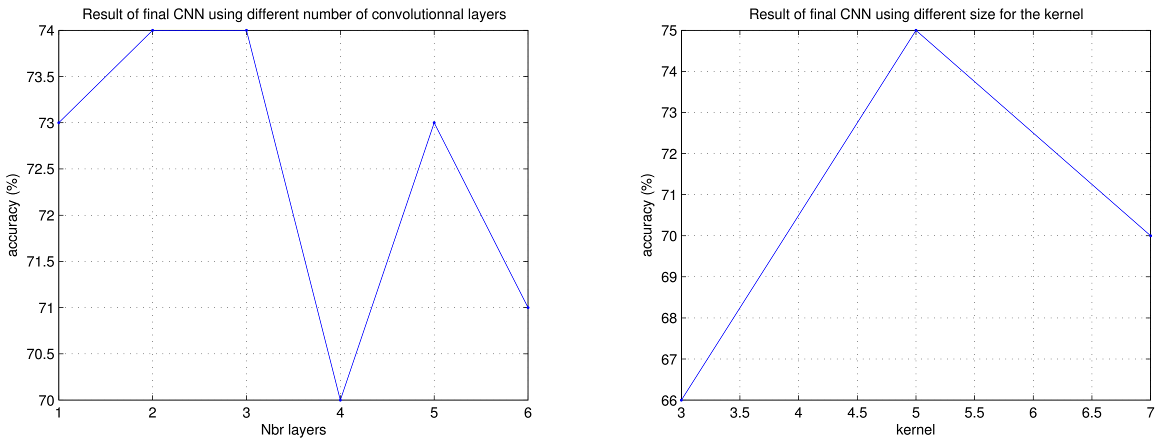


FIGURE 9 – Measure accuracy on testing dataset in function of the number of layers (a) and by changing the kernel size : 3x3, 5x5 and 7x7 (b)

Once the number of layers was decided to 5, the next parameter to observe was the kernel size of each convolutionnal layer. It can be observed in Figure 9 that the best size for the kernel is 5x5.

Next, the value of the dropout was addressed. The dropout is used to prevent overfitting of the model by randomly dropping units in the neural network. With a dropout rate of 0.5, there is a 50% chance that a unit in a visible or hidden layer will be dropped.

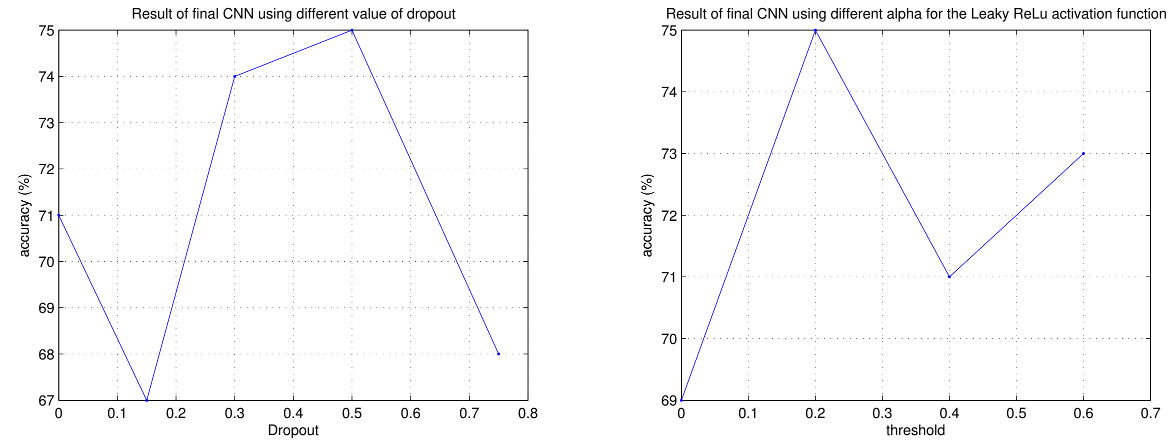


FIGURE 10 – Measure accuracy on testing dataset in function of the dropout rate (a) and in function of α of the Leaky ReLu activation function (b)

Figure 10 tells us that the best accuracy can be found at a dropout of 0.5 which is the value kept for the final neural network. Then, the focus was on the activation function of each layer. For the convolution, a leaky ReLu of 0.2 is used, which allow a small positive gradient when the unit is not active. The best accuracy is obtained when a leaky ReLu of 0.2 is used, at it can be seen on Figure 10. Finally, the batch size of the training was investigated. A batch size of 16 seems like the best fit for our CNN.

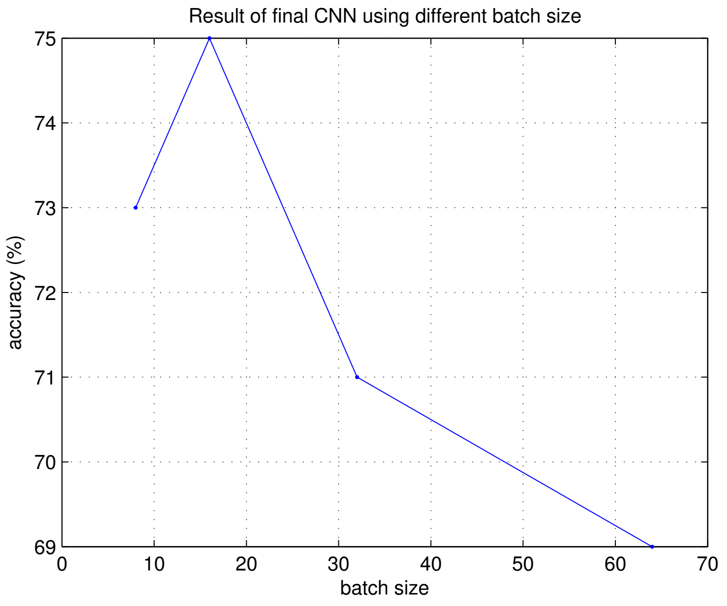


FIGURE 11 – Measure accuracy on testing dataset in function of the batch size : 8, 16, 32 and 64

4.2.2 3 Layers CNN

As the 5 layer CNN wasn't giving satisfying results both in term of accuracy and audio quality, it was decided to train a new neural network with a smaller dataset, namely one genre of music ("Songwriter" songs). The CNN was modified to a 3-layer CNN with a 0.5 dropout rate on the first layer of the decoder.

The training set was composed of 17 songs of one genre. 1 test song of the same genre was used to measure the accuracy.

Accuracy on testing set	**72 %**
Accuracy on training set	80 %

FIGURE 12 – Accuracy results using one genre

This time the accuracy on both training and testing set has improved. 75 epoch was used to train the CNN in order to obtain the best results possible :

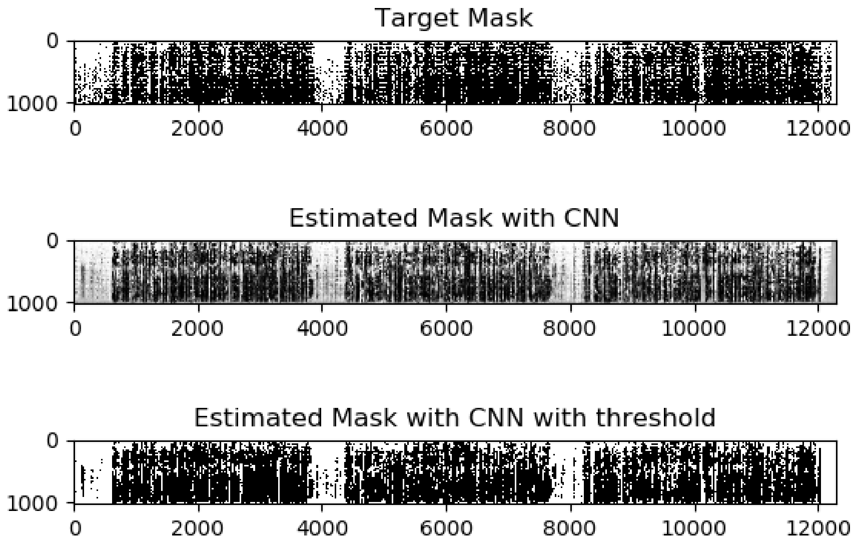


FIGURE 13 – Estimated masks with CNN trained on one genre of songs using epochs = 75, batch size = 16 and threshold = 0.5

The estimated masks and the reconstructed audio signals give better results than using a 5 layers CNN. The improvements can be seen on the mask (Figure 13) and also heard in the reconstructed song. The instruments in the background are almost inaudible.

5 Conclusion

We demonstrated that a deep learning model is effectively capable of separating a singer voice from a mixture song. We investigated the effects of the CNN parameters on the accuracy.

Some possible improvements for this project are building a more complex neural network architecture (RNN..). Also, adding some noise to the audio data and measuring the robustness of the algorithm could be a challenging task.

Unlike natural images were small perturbations can be unseen, small errors in the spectrogram prediction can lead to unpleasant results at hearing. So even with high accuracy values, the results at hearing are still perfectible.

Références

[1] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. Medleydb : A multitrack dataset for annotation-intensive mir research. In *in Proc. the 15th International Society for Music Information Retrieval Conference (ISMIR*, 2014).

[2] Emad M. Grais and Mark D. Plumbley. Single channel audio source separation using convolutional denoising autoencoders.

[3] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. Singing-voice separation from monaural recordings using deep reccurent neural networks.

[4] Gerard Roma, Emad M. Grais, Andrew J.R. Simpson, and Mark D. Plumbley. Singing voice separation using deep neural networks and f0 estimation. *Centre for Vision, Speech and Signal Processing.*