Vocals Isolation from Monoaural Music Source with Deep Recurrent Neural Networks

Daniel del Castillo
KTH
danieldc@kth.se

Evangelia Gogoulou
KTH
gogoulou@kth.se

Quentin Lemaire
KTH
qle@kth.se

Abstract

This work aims at obtaining a clean isolated signal containing the vocals information from an arbitrary song. The isolation is performed by a Deep Recurrent Neural Network, which receives the mixed signal coefficients as input and produces the vocal coefficients in the output. Both choices of Mel-Frequency Ceptrum Coefficients (MFCC) and Short Time Fourier Transform (STFT) coefficients are explored for the network input. After training and evaluation of the proposed network on the MIR-1K dataset, vocal separation of moderate performance was achieved.

1 Introduction

Monoaural music source separation has become increasingly relevant in recent years [1], [2]. Moreover, latest Deep Neural Network (DNN) architectures have shown to yield more robust results than conventional methods [3], [4].
The task of music source separation belongs to the field of Music Information Retrieval (MIR). The motivation for this task is to achieve a clean separation of the vocals signal from the rest of signals contained in a monoaural music source. The vocals signal usually contains most of the information that is relevant for MIR-related applications, such as singer identification or automatic lyrics transcription. However, blind separation, i.e. without previous knowledge of the source content, is a challenging task, partly because of the similarity between the spectra from human singing voice and some voiced musical instruments, such as piano.

In one of the recently proposed deep learning architectures, trying to solve the singing voice separation problem, a U-net network is used [3]. The U-net architecture is based on both fully convolutional and deconvolutional networks, with additional skip connections between layers at the same hierarchical level. The network output is a soft mask that is applied to the mix signal in order to separate the voice component from the instrumental one. Simpson et al trained a deep convolutional neural network, consisting of around 1 billion of parameters, in order to produce a probabilistic binary mask that separates the vocal components from the mixed signal [5].

The current project is inspired by the work presented in [6]. In this paper, the option of deep recurrent networks is explored. More specifically, standard RNN and DRNN with temporal or full recurrent connections are studied. Both discriminative functions and generalized Kullback-Leibler divergence criteria are explored as an option for the network optimizer function. The RNN layers are jointly optimized, alongside with the masking function which has been added as the last layer. The network input corresponds to the magnitude spectral features, extracted by Short Time Fourier Transform. The network produces the magnitude spectral coefficients of both the vocal and instrumental components. Subsequently, the corresponding time domain signals are produced, using the inverse Short Time Fourier Transform.

2 Method

In the core of the proposed methodology lies a Deep Recurrent Neural Network (DRNN). The basic Recurrent Neural Network (RNN) consists of an infinate number of layers, which introduce a memory from previous timesteps. In that way, the correlation between audio coefficients from different time steps is captured. The introduction of multiple RNN layers leads to hierarchical processing of the input. That way, more information about the correlation between the audio signal coefficients is captured. The main difference with the main reference work [6] is that the proposed architecture tries to learn only the vocal signal, with the purpose of decreasing the computational time of training. Additionally, the time masking layer is not applied, since both the vocal and instrumental components are needed.

2.1 Preprocessing

In this work we have explored several approaches for the input features to a Deep Recurrent Neural Network (DRNN) model. In an attempt to discriminate between human voice and the rest of instruments in the music source, we experiment by feeding Mel-Filterbank Cepstrum coefficients (MFCC). The main advantage of using these features is the reduced dimensionality, facilitating the training with a DRNN. Following the work in [6], we also feed our model with different number of Short-Time Fourier Transform (STFT) coefficients (129, 257, 513).

To facilitate comparison of the different input feature settings, the same procedure parameters (frame length, overlap, FFT lenght, etc.) are used in the extraction of Mel-Filterbank Cepstrum Coefficient (MFCC) and STFT features.

2.1.1 Mel-Filterbank Cepstrum Coefficient

The Mel-Filterbank Cepstrum Coefficient (MFCC) vector $c_y(n)$ extraction procedure from an audio signal $s(n)$ is described in Figure 1. We assume that the signal $s(n)$ has already been enframed, to ensure that frequencies are stationary over a very short period of time. In our case, since all data audio files were sampled at 16 kHz, we have chosen a frame size of 512 samples, with a 50 % overlap. After enframing, the first step consists in amplifying higher frequencies, which usually have smaller magnitudes, an operation called Pre-emphasis. This will help in posterior steps by balancing the frequency spectrum and improving the Signal-to-Noise Ratio (SNR). Because the Fast Fourier Transform (FFT) operation assumes that the data is infinite, previous to the FFT we apply a Hamming window to the, still, time series signal. The conversion from time series to frequency domain is made by an FFT. The magnitude of FFT output is then extracted and squared to obtain the power spectrum. This step involves information loss, since we discard the phase. Once we have the power spectrum, we apply a set of triangular filters, typically 40, on a Mel-scale to the power spectrum. The Mel-scale is a non-linear frequency scale that resembles human ear perception of sound: it is more discriminative at lower frequencies and less discriminative at higher frequencies. Finally, we can apply Discrete Cosine Transform (DCT, type II) to decorrelate the filter bank features, previously taking the natural logarithm of them.

To minimize the information loss in the inputs and maximize the quality of the reconstructed output audio signals after separation, we select the full set of cepstrum coefficients for each feature vector.

2.1.2 STFT

The STFT features are extracted simply by performing an STFT operation on the time series audio signal. We then obtain the magnitude spectrum by performing a modulo ($|\cdot|$) operation on the complex output of the STFT.

2.2 Network Architecture

In order to process a song in a neural network, we will need a network that could take time into account. Therefore, the input at the frame t will not only be the features extracted for that frame by the preprocessing but we will also include information from the frame t-1, this type of network is called Recurrent Neural Network (RNN). The following figure 2 illustrates the principle of a RNN.

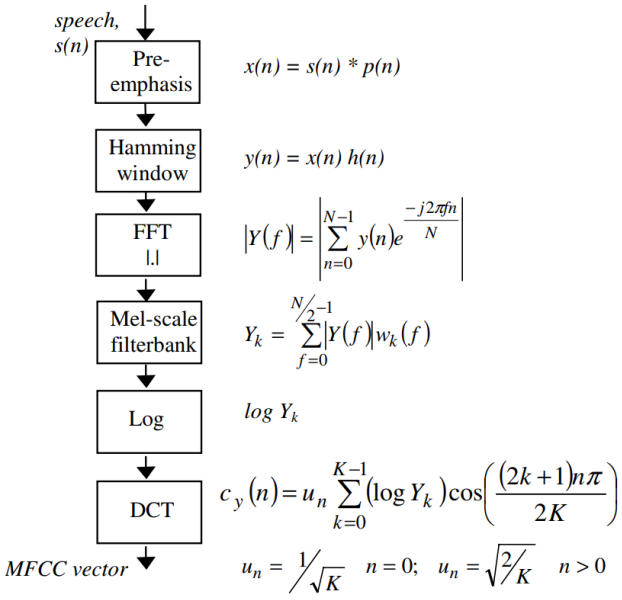


Figure 1: Step-by-step procedure for obtaining MFCCs. Source:[7].

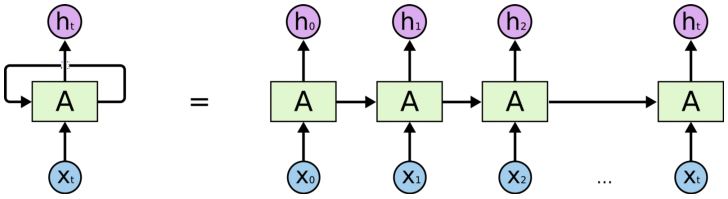


Figure 2: Principle of a Recurrent Neural. Network Source: Source:[8]

There are different ways to use a RNN (Figure 3). In this case, we will use the "many-to-many" way to map input and output. Therefore, by inputting a preprocessed song, we will get another preprocessed output.

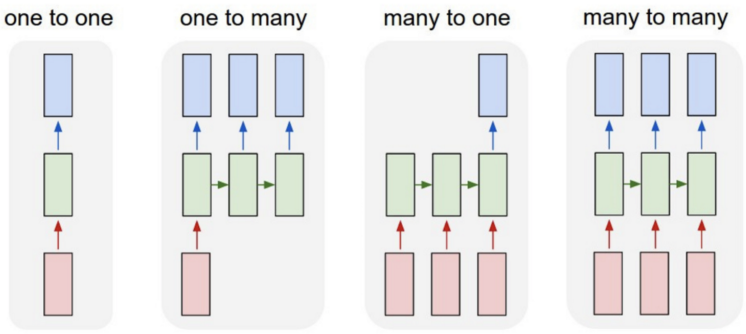


Figure 3: Different usages of a RNN. Source:[9]

This architecture is very simple and presents some limits. The main problem of a RNN network is the vanishing gradient that can restrain the learning process. One of the common replacement is Long Short Term Memory (LSTM) [10]. The first advantage of LSTM is that it solves the vanishing gradient problem. Secondly, an LSTM unit can keep information for a longer period. In the proposed network architecture, a Gated Recurrent Unit (GRU) is used, proposed by Cho et al.[11]. Like LTSM

units, GRUs overcome the vanishing gradient problem and can keep information for a longer period. Additionally, both types of units are equivalent from the perspective of performance. However, the architecture of GRU units is simpler than the architecture of an LSTM unit. For that reason, they have less parameters, a fact that leads to higher computational efficiency [12] and less training time. The fact that GRUs are not characterized by long term memory is irrelevant to the problem of vocal separation, since the number of past coefficients needed at each step is small. Comparing to RNN units, GRUs avoid the problem of vanishing gradients, since they consist of both update and reset step.

In order to overcome some of these limits, we will use what is called Deep Recurrent Neural Network (DRNN) in order to be able to process more complex data like the feature vector we can get after the preprocessing. The concept is very simple, after going through a GRU, we use the output as input of another GRU and so on. In other words, we "stack" the GRU network in order to create a deep GRU network.

After the DRNN, a fully connected layer was added, with as many hidden units as the number of input coefficients. The activation function used was ReLu.

2.2.1 Loss function

Given the output of the network y and the target $\hat{y}$, we have defined the loss of the network as a Mean Square Error (MSE):

$$l = ||y - \hat{y}||_2^2 \tag{1}$$

Unlike the original paper [6], the error component regarding the instrumental component is eliminated, since this is out of the scope of the project. So, the resulting formula of the loss function has a simpler form.

2.2.2 Optimizer

To minimize this loss, we will use the ADAM optimizer [13]. From our experiences in the field, it seems that the ADAM optimizer is a good to train efficiently a network from scratch which is our case.

The ADAM optimizer is less affected by the initialization of the hyper parameters of the network than a simple stochastic gradient descent (SGD) and, therefore, it is easier to train a network with the ADAM method. But the ADAM method seems also more likely to reach a local minimum than the classical SGD.

So in order to have an effective learning in the time allocated for this project, the ADAM solution seemed to be the best compromise between accuracy and easiness of training.

2.3 Post-processing

The DCT Type II is invertible; there is no loss of information when performing this operation. The same applies for the rest of the operations in Figure 1, except the Mel-scale Filterbank and the discard of the phase spectrum. These two steps present a challenge in the task of reconstructing the audio signal from the outputs of the neural network. To overcome this, we have chosen the following strategies.

As the method for reconstructing the power spectrum from the MFCCs, we apply the commonly used $l2 - norm$ criteria, analyzed in detail in [14]. After computing the power spectrum, we simply apply the square root to get the magnitude spectrum of the signal to reconstruct.

Once we have obtained the magnitude-only spectrum of the signal to reconstruct, we need to find a way to estimate the time series samples from it. For the case when the phase spectra is not available, we have selected the Griffin-Lim algorithm [15] for its robustness and applicability. If we have the phase spectra, we can simply compute the Inverse STFT (ISTFT). To do this, and following the strategy presented in [6], we recover the previously saved phase spectrum for each song and combine it with its corresponding reconstructed magnitude spectrum.

The two different approaches followed for reconstructing the audio signals from the estimated magnitude spectra, along with the rest of the data pipeline, are presented in figures 4 and 5.

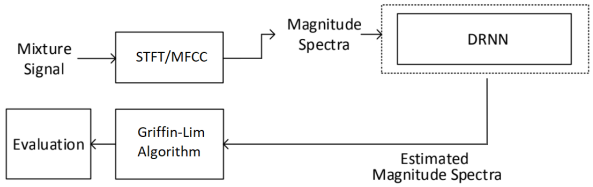


Figure 4: Reconstruction by Griffin-Lim algorithm. Inspired by [6]

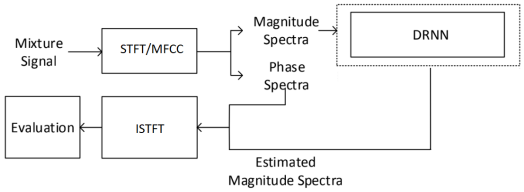


Figure 5: Reconstruction by ISTFT algorithm. Inspired by [6]

Our experiments with both Griffin-Lim and ISTFT for the reconstuction showed that Griffin-Lim proves itself to be a robust algorithm for medium-quality reconstruction from only-magnitude spectra for the vocals isolation task. On the contrary, applying ISTFT by combining the previously saved phase spectra with estimated magnitude spectra gave unexpectedly bad results. As a result, we have chosen to use Griffin-Lim algorithm as the only reconstruction method in our experiments. Additionally, we tried to initialize Griffin-Lim with the target vocals phase, instead of using the random initialization of Griffin-Lim algorithm. However, there was not significant improvement in the reconstruction of the vocals.

2.4 Evaluation method

The evaluation of the proposed vocals isolation system is done after the reconstruction of the predicted vocal signal, which is compared to the original vocal signal. For the qualitative evaluation of the reconstruction, the metrics Source to Inference Ratio (SIR), Source to Artifacts Ratio (SAR), and Source to Distortion Ratio (SDR) are used [16]. The SDR evaluation metric measures the similarity between the predicted vocal signal and the true vocal signal, while SIR measures the similarity between the predicted vocal signal and the true instrumental signal. Finally, SAR measures the dissimilarity of the predicted vocal signal to either original vocal or instrumental components. Ideally, high SDR, high SAR and low SIR values are the ones desirable. Given a test set, the global SDR, SIR and SAR can be obtained by computing the weighted average of the individual ratios, using the length of the signal as weight.

3 Experiments

3.1 Experiment settings

For the experimental evaluation of the proposed system, the MIR-1K dataset [17] is used. This dataset consists of 1000 clips with duration from 4 to 13 seconds and sampling rate 16 KHz. For each clip, the mixed, vocal and instrumental signals are provided. This fact constitutes the main reason why MIR-1K dataset is widely used for the evaluation of blind audio source separation algorithms, since the dataset is labelled.

The first pre-processing step is splitting the dataset into training, validation and testing set, with percentages 70%, 20% and 10% respectively. For the purpose of avoiding overfitting, the music clips, sung by 17 singers, were uniformly distributed into the three sets, in a way that all singers are represented in all sets.

The experiments were executed on a hardware with the following specifications:

- CPU Platform: Intel Sandy Bridge 3.60 GHz
- 4 vCPUs, 26 GB memory
- 1 x NVIDIA Tesla K80

3.2 Training of the network

3.2.1 Implementation

To implement the network, we used the Python library Tensorflow [18].

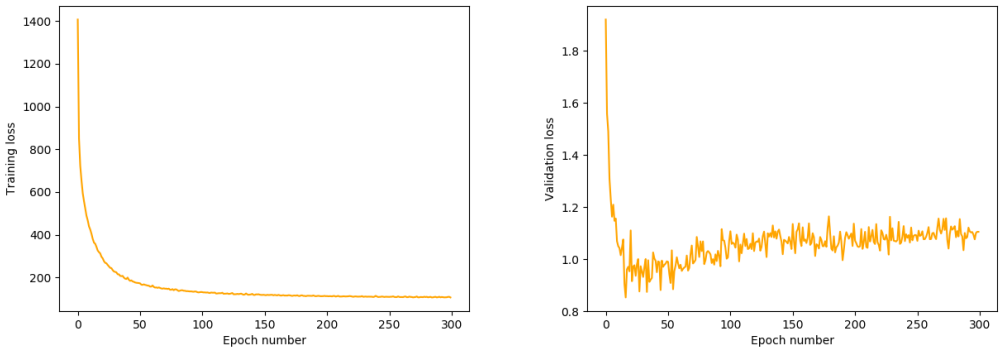
3.2.2 Hyper-parameters

We chose to stack 3 GRU units with 256 hidden nodes each, followed by a fully connected layer with as many units as the dimesions of the network input. As mentioned before, the addition of extra GRU layers might improve the generalization performance of the network. However, the risk of overfitting is higher. So, 3 layers seemed like a good compromise for our experimentation. Regarding the learning rate, both choices of static and exponential decay were explored. After experimenting with both options, we came to the conclusion that the use of static learning rate of 0.001 led to a more stabilized learning curve. Additionally, the option of dropout regularization was explored. After experimenting with the dropout rate, we decided not to use dropout in our experimental evaluation, since no significant improvement of the network performance was observed.

4 Results

4.1 Experiments with STFT

After training a network with 3 RNN layers, the trend training and validation losses obtained are illustrated below:



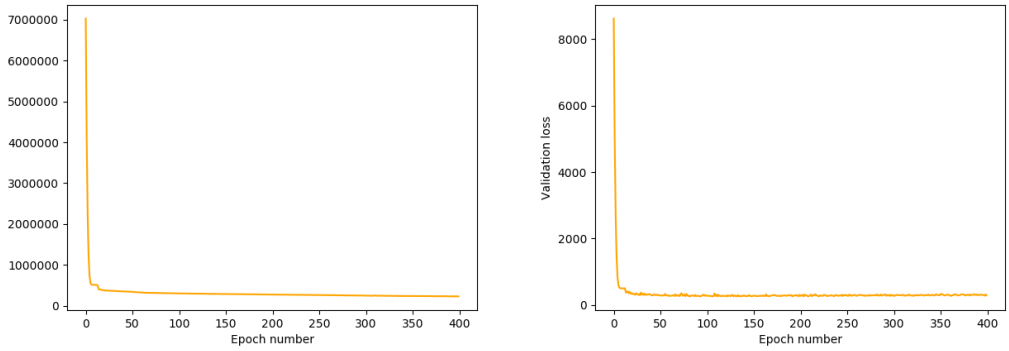
(a) Training loss (b) Validation loss

Figure 6: Training and validation loss for 3 RNN layers with STFT input

As we can see, the training loss converges approximately at epoch 150, which means that the network reaches a stable point quite fast. However, the validation loss keeps oscillating throughout the training phase. This means that the learning procedure is not stabilized. Taking the oscillating validation loss as an indicator of the network performance, it can be concluded that the generalization performance is quite poor.

4.2 Experiments with MFCC

In this case, the MFCC coefficients are used as the network input. After training a network with 3 RNN layers, the following training and validation losses were obtained:



(a) Training loss (b) Validation loss

Figure 7: Training and validation loss for 3 RNN layers with MFCC input

From the figures above, it can be concluded that the quality of learning is even worse, since the network converges after only 50 epochs, while the final value of training loss is quite high. Comparing to the previous case of STFT input, the values of both training and validation loss are quite high. This is unavoidable considering the low dimensionality of MFCC coefficients (49 coefficients), comparing to the dimensionality of STFT coefficients (257 coefficients).

4.3 Qualitative evaluation of reconstructed vocals

The evaluation of the trained network was done on a test set of 200 songs. For each song, the steps of pre-processing with either STFT or MFCC coefficients, network evaluation and post-processing with Griffin-Lim were followed. Subsequently, the evaluation metrics SIR, SAR and SDR were computed for each song. The resulted weighted average evaluation metrics are presented in the following table:

	MFCC			STFT		
#Layers	SDR	SIR	SAR	SDR	SIR	SAR
1	2.56	2.56	206.35	0.68	0.81	122.06
3	3.32	3.32	204.17	1.35	1.47	115.05

Table 1: Evaluation metrics for architectures with 1 or 3 layers and stft or mfcc coefficients as input.

The qualitative evaluation of our trained models leads to the conclusion that vocal separation of moderate quality was achieved. Comparing to the results presented in [6], the global SDR, SIR, SAR, obtained in the present work, are almost 50% worse. In the case of MFCC input, the values of SDR and SIR are quite close, which indicates that the reconstructed vocals don't clearly match either the vocal or the instrumental component of the original song. This is not the case for the STFT input, where there is a clear distinction between the SAR and SDR values. What is quite surprising, is the fact that evaluation metrics obtained with MFCC coefficients are slightly higher that the ones obtained with STFT coefficients. It was expected that MFCC coefficients would not be appropriate for the problem of vocal separation, since the application of the MFCC pipeline removes the correlation between the coefficients.

5 Conclusion

Different conclusions can be drawn from our experiment. First, we have been able to compare two different pre-possessing and post-processing, MFCC and STFT, over a complete pipeline for the problem we have chosen. We expected STFT to perform better than MFCC, since the information encapsulated in the correlation between the coefficients could be a benefit for the network. Surprisingly, the use of MFCC features led to better reconstruction than STFT features.

Secondly, we expected to see the effect of deep architectures for such a complex problem as vocal isolation. We expected that the addition of extra layers will lead to better performance. Our experiments confirm that expectation to a small extent, since the performance of the network with 3 layers is only slightly better that the performance of the network with 1 layer.

Finally, we have seen the difficulties of the reconstruction process. Our experiments illustrate the delicate nature of the reconstruction of a signal from its coefficients. Our attempt to apply ISTFT by combining the phase of the original spectra with vocals magnitude spectra, learned by the network, was not successful. Our initial assumption about this problem was the different scale between the original phase and the magnitude learned by the network. However, our second attempt to normalize the MFCC coefficients didn't lead to better reconstrunction quality.

One thing that can be done with this project the lyrics recognition. In order to simplify the recognition, it could be relevant to, first, extract the voice from the audio and then apply a speech recognition network on the voice. This problem is harder than traditional speech recognition due to the singing voice but our work is one of the ways to simplify it.

Appendices

In the peer-review phase, the current report was reviewed by 2 different individuals. Both reviews are summarized below:

- *Relevance for the learning outcomes:* Both reviewers agreed that the current work is highly relevant to the content of the course.
- *Novelty/Originality:* Both reviewers that the level of novelty of the current work is low. We think that our work is original to some extent, since we modified the architecture proposed in [6]. Our initial idea of using a custom made dataset of folk songs was not applicable, since the variety of the available, labeled datasets, which are suitable for this problem, is small.
- *Literature study:* Both reviewers made positive comments about the quality and extension of the literature study presented.
- *Outcome correctness:* The improvement points proposed by the reviewers were taken into consideration.
- *Clarity of presentation:* The format of the evaluation table was corrected, as proposed by both reviewers. Additionally, more results were added, so that the comparison between the learning curves is meaningful.
- *Overall quality:* After taking into consideration both reviewers' comments, the section of results was enriched with more experiments. Additionally, some minor errors spotted were fixed.

References

[1] I. Y. Jeong and K. Lee, "Vocal separation from monaural music using temporal/spectral continuity and sparsity constraints," *IEEE Signal Processing Letters*, vol. 21, no. 10, pp. 1197–1200, 2014, ISSN: 1070-9908. DOI: 10.1109/LSP.2014.2329946.

[2] H. Tachibana, N. Ono, and S. Sagayama, "Singing voice enhancement in monaural music signals based on two-stage harmonic/percussive sound separation on multiple resolution spectrograms," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 22, no. 1, pp. 228–237, 2014.

[3] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep u-net convolutional networks," 2017.

[4] P. S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, 2015, ISSN: 2329-9290. DOI: 10.1109/TASLP.2015.2468583.

[5] A. J. Simpson, G. Roma, and M. D. Plumbley, "Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network," in *International Conference on Latent Variable Analysis and Signal Separation*, Springer, 2015, pp. 429–436.

[6] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Singing-voice separation from monaural recordings using deep recurrent neural networks.," in *ISMIR*, 2014, pp. 477–482.

[7] B. Milner and X. Shao, "Speech reconstruction from mel-frequency cepstral coefficients using a source-filter model," in *Seventh International Conference on Spoken Language Processing*, 2002.

[8] *All about recurrent neural networks*, https://medium.com/@jianqiangma/all-about-recurrent-neural-networks-9e5ae2936f6e.

[9] *Recurrent neural networks and lstm*, https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, Nov. 1997.

[11] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014. arXiv: 1406.1078. [Online]. Available: http://arxiv.org/abs/1406.1078.

[12] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. arXiv: 1412.3555. [Online]. Available: http://arxiv.org/abs/1412.3555.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. arXiv: 1412.6980. [Online]. Available: http://arxiv.org/abs/1412.6980.

[14] G. Min, X. Zhang, J. Yang, and X. Zou, "Speech reconstruction from mel-frequency cepstral coefficients via ???1-norm minimization," in *2015 IEEE 17th International Workshop on Multimedia Signal Processing (MMSP)*, 2015, pp. 1–5. DOI: 10.1109/MMSP.2015.7340799.

[15] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984, ISSN: 0096-3518. DOI: 10.1109/TASSP.1984.1164317.

[16] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE transactions on audio, speech, and language processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[17] *MIR-1K Dataset*, https://sites.google.com/site/unvoicedsoundseparation/mir-1k.

[18] *Tensorflow*, https://www.tensorflow.org.