Proceedings of the 2nd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/75/20240536

Singing voice separation based on U-NET neural network and its application

Bruce Wu

Shanghai High School International Division, Shanghai, China

brucewu1108@outlook.com

Abstract. Singing Voice separation is a very important method that helps people to obtain a vocal or accompaniment for further usage like music composing. In this paper, I developed a Song accompaniment separation application based on Convolutional neural network—Unet model and Deep learning (DL). Basically, the U-net convolutional network is applied to the two-dimensional spectrum of a music audio in order to achieve a more precise separation effect. During this deep learning process, a significant amount of data was inserted into the model to get the best fit parameters. Then, after all, the model can automatically generate a mask that filters out the accompaniment or vocal part that users want. This completed model is then transmitted to the terminal of a internet server to enable users to get access to the model through internet. After the test of both quantitative evaluation and subjective assessment, the application impressively achieved the Song accompaniment Separation that provides a good quality of both vocal and song.

Keywords: U-net, Deep Learning, Sing-voice separation.

1. Introduction

Nowadays, the field of music composition relied on music composing software or Digital Audio Workstations (DAW). DAW is an application that is used for audio recording, audio editing, MIDI editing, mixing, and mastering. It processed analog audio into a digital audio information [1]. Then, the digital audio information presents as either a piece of sound wave or a stack of music notation on a panel. These visualized elements can be then manipulated by people using the functions inside the DAW. In order to gather digital audio information, many existing songs are converted into samples that can be used in the DAW [2]. There are two types of samples, vocal, and accompaniment. However, in many songs, vocal and accompaniment are mixed together that users cannot use one of them directly from the songs. Due to this reason, Singing voice separation (SVS) becomes an important step in the field of music making, remixing songs, editing audio [3], for examples. To achieve the separation of melody and vocal, the program needs to recognize the differences between them. Similar to human's ear, where there are sensory cells called heir cells that can easily interpret the sound and further analyze it in brain. However, singing voice separation in computer is far more challenging, since programs require well designed solutions and algorithms that the computer can use instead of human's brain and nerve system. In the beginning of the development of SVS, computational auditory scene analysis system is proposed [4]. The system is firstly used in speech separation, it is a system consists of both non-speech

DOI: 10.54254/2755-2721/75/20240536

components and speech components, the system simulates the human's auditory system in order to accomplish the separation of speech and background noises [5].

Besides planning an algorithm to the computer, make the computer to learn to build a system itself to separate the song is an easier solution. Recently, deep learning (DL), an aspect of machine learning (ML) is very popular. It is a way to teach artificial intelligence to construct a method that helps it to succeed in completing the tasks given by human based on massive data including labeled data or unlabeled data. [6]. Deep Learning (DL) has two major advantages in program construction. First, it assures the accuracy of the entire system [7]. The more data it obtains, the more accurate the system can become, plenty of data are references that the computer can use and enable the computer to rectify the parameter inside the system. After multiple rectification, the whole program can eventually reach people's expectation with only a little error. Second, DL is easier to be understood and applied, since a majority of parts of DL is gathering data to train the program system or model. Comparing to the way that people directly give the computer a scenario, DL requires less efforts in proposing an algorithm [8].

Similarly, SVS can be addressed using DL. As mentioned earlier, in order to separate the voice and accompaniment, the program needs to recognize the differences between these two elements. Abundant resources of existing music samples help the program to consult a proper model. In this paper, I first establish a model framework. Then, by sending a large number of music sample to the program, it completes the model framework by providing adjusted parameter to the model. After multiple adjustments, the program would find the most suitable parameter for the model.

2. Method

2.1. Data Preprocessing

For the music samples that I used in the supervised learning, I choose MIR- 1K dataset, IKALA dataset, and MedleyDB dataset. IKALA dataset is a collection of 252 songs that are 30s long [9]. MIR-1k is a dataset that contains more than 1000 music clips, they are all music pieces selected from Chinese pop songs. MedleyDB is a dataset that contains 122 multi-track songs that belongs to many different music genres, for example, jazz, pop, and rap, and there are 108 *.yaml* files that each labeled one of the multi-track song [10]. All of them contained abundant music samples that are very helpful to train the U-net model. In the three datasets, almost all of the music samples are stored in *.wav* file format or *.mp3* file format. However, all of the datasets are mixed audio signal that can't be directly used as a suitable data for the model. I need to first spawn the ground truth accompaniments and vocals of each song in the datasets so the model can use them as references to train itself. Due to this reason, the music samples require a preprocessing step that can convert all the original songs into its accompaniment part with the vocal part separately.

- Process IKALA: For both MIR-1K and IKALA datasets, they are composed of left and right channels. The channels refer to either the recorded vocal or the accompaniment. As shown in the **Figure 1 i**n order to process these two datasets, I used function *librosa.load.* The function helps to load an audio file as a floating point time series with a certain rate. *y, _ = load(audiofile, sr=None, mono=False)* for instance. Then I defined three variables each refers to the channels with vocal, accompaniment, and the mix of them (adding the vocal and the accompaniment together). After that, the three variables and the file names are stored by using function, *util.SaveSpectrogram* that helps to save all the time series into a spectrogram.

```
y, _ = load(audiofile, sr=None, mono=False)
inst = y[0, :]
mix = y[0, :]+y[1, :]
vocal = y[1, :]
util.SaveSpectrogram(mix, vocal, inst, fname)
```

Figure 1. A display of the code used in the separation of IKALA datasets

DOI: 10.54254/2755-2721/75/20240536

-Process MedleyDB: In the MedleyDB datasets, all songs are specialized into several different sound tracks. All the song tracks including their accompaniments and vocals are labeled in *.yaml* file that contains basic description about each music. As shown in **Figure 2** the program basically searches the the MedleyDB dataset based on the features provided in the *.yaml* file to find corresponding part of the songs like accompaniments, and vocals, and rearrange them into the format that can be used in training the model. For instance, the function *find("male"),* is used to find the songs with the feature "male".

```
print(
      "stem: %s %s %s" % (fname, stem["component"],
 stem["instrument"]))
  if (''.join(stem["instrument"]).find("male") >= 0) or \
          (''.join(stem["instrument"]).find("singer") > 0):
      stem_voc.append(fname)
      all_voctracks.append(fname)
      print("Is vocal!")
  else:
      stem_inst.append(fname)
      all_insttracks.append(fname)
```

Figure 2. A display of the code used in the separation of MedleyDB datasets

2.2. Network Model Introduction

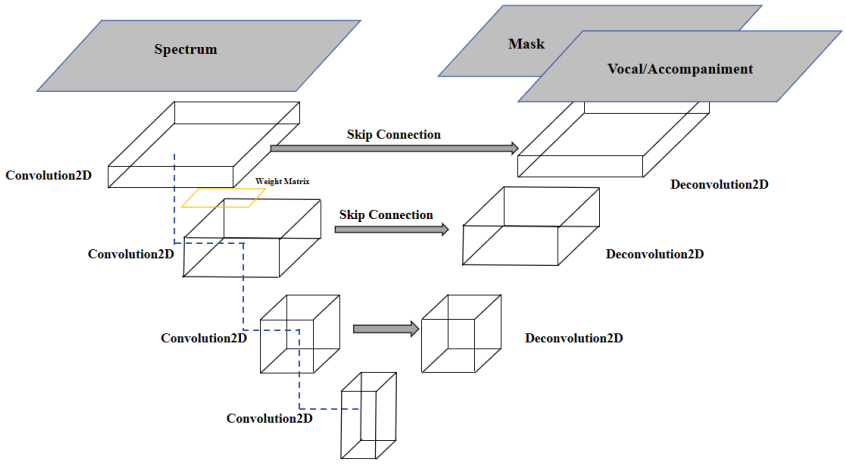

Figure 3. A display of my U-net convolution network.

U-net is a network model that is originally designed to segment a picture based on a given feature like medical image segmentation. It is a chain composed of convolutions. The structure of the U-net network model can be specialized into two parts: contracting path and expansive path [11]. According to Fig.3, in the contracting path of my U-net model, 4 layers of 2D convolutions are being contacted together. All 2D convolutions contained in the contracting path have a certain kernel that multiplies with the input from the previous layers in order to obtain the related features of the given data. On the opposite side of the contracting path is the expansive path. Expansive path also possesses four layers, however, each layer is constructed by deconvolution. The deconvolution expands the picture into normal size that contains the characteristics that I need. Furthermore, each corresponding convolution and deconvolution are connected by a type of connection called skip connection. The skip connection mentioned here helps to recover fine grained details in the prediction.

As mentioned previously, U-net model is originally used for image segmentation, in my project, I use Fourier transform to transform the audio from time domain into frequency domain that changed the one-dimensional audio into a two-dimensional spectrum image of the audio. The spectrum is firstly

DOI: 10.54254/2755-2721/75/20240536

entered as an input to the U-net neural network. Then, the spectrum is transformed into the four convolutions and the four deconvolutions, respectively, finally, it produced a mask and the accompaniment/vocal as output. My program uses *Convolution2D* function from the *chainer.links* documentation. This links function wraps the *chainer.functions.convolution_2D* function and holds the filter weight and bias vector as parameters, which the *chainer.functions.convolution_2D* is the function that fulfill a 2-dimensional convolution. On the other hand, in order to deconvolute, I use the function deconvolution2D also from the *chainer.links* documentation. It also holds the filter weight and bias vector as parameters as well as putting together the *chainer.functions.deconvolution_2D*. Meanwhile, the program uses normalization that regulates the data into a certain range that helps the program to analyze it more efficiently. I use function *Batchnormalization* from the *chainer.links* documentation to link each layer from the neural network.

2.3. Application

In order to put the U-net neural network into actual usage. I wrote an extra program that contains functions which directly convert the .mp3 music file into the accompaniment and vocal that I want. The three functions are *convert_wav_mp3, use_model_get_VA*, and *split_it*. *Convert_wav_mp3* function is to convert the given mp3 file into *.wav* format. *Use_model_get_VA* is to apply the neural network model to deal with the spectrum of a music. Finally, *split_it* is the final function that contains every step used to split a music piece into accompaniment and vocal. Including transformation of a .mp3 file into *.wav* file (refers to the function *convert_wav_mp3*), converting the *.wav* file into spectrum, process the spectrum through the model, getting the final spectrums that each refer to the accompaniment and vocal, and converting the spectrums into normal *.wav* file format.

The next step is to post the program onto a website with a server that enables others to also use this trained U-net model. I employ Flask and Nginx in my web application server setup. Flask, a Python micro web framework, is intentionally designed for simplicity and ease of use. It equips developers with the necessary tools and libraries to construct web applications with minimal effort, catering to both novices and seasoned developers. Flask adheres to the WSGI (Web Server Gateway Interface) standard, enabling seamless integration with various web servers, including Nginx that I used in the web server deployment [12]. Nginx stands as a high-performance, open-source web server, reverse proxy server, and load balancer. It excels in efficiently delivering static content, managing incoming web traffic, and distributing requests to backend servers. Nginx is frequently employed in conjunction with application servers like Flask to enhance performance, security, and scalability [13]. Finally, I connect to the server's terminal using Termius, as shown in **Figure 6** in order to upload my programs and model onto the server.

The webpage that allows users to access my model is displayed below **Figure 4** and can be accessed via the public link: http://gsbzfl.xyz/

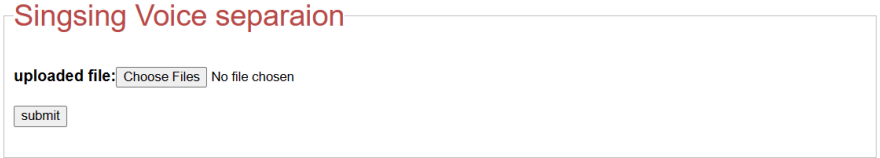

Figure 4. A display of the website before uploading music file

Upon uploading audio and clicking the submit button, the system will send the audio to my server, where it will be processed using my trained U-Net model for separating singing voices. Subsequently, the separated vocal and music audio will be made available for download on a dedicated page, like the one shown below **Figure 5**.

DOI: 10.54254/2755-2721/75/20240536

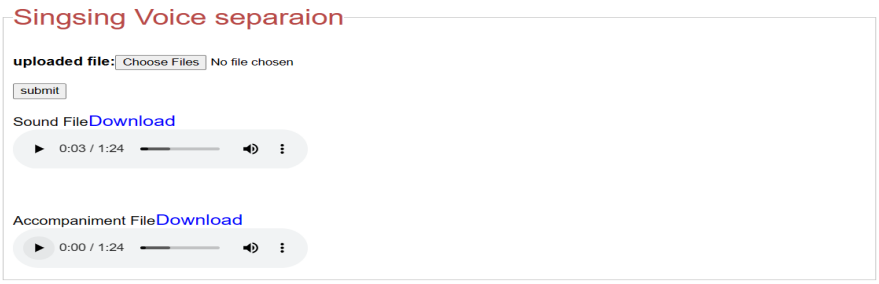

Figure 5. A display of the website after uploading music file

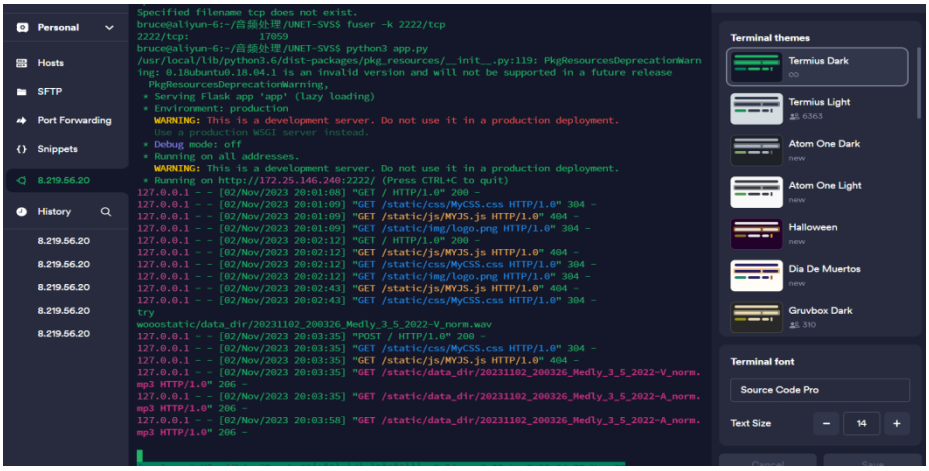

Figure 6. An overview of the server's terminal

3. Experiment

In order to testify the outcome of the U-net Deep learning model in my SVS project. I use totally three references to be the testify data. They are Source-to-distortion Ratio (SDR), Source-to-artifact Ratio (SAR), and Source-to-interference Ratio(SIR). Both of the three evaluation indexes are kinds of numeral data that can be generated from the separation product, and compared to the ground-truth to see the overall quality of the separation of my model. Among the evaluation based on these three values, I use functions stored from the *fast_bss_eval*'s documentation. Additionally, I put the comparison between the spectrum of my separated product and the spectrum of the ground truth source into the objective evaluation. Eventually, I will use my generated data to compare with the data that is generated from other algorithms or models that are used in SVS.

My separated source contains the target source that I want as well as other sound source, noises, artifact [14]

$$s_r = s_{target} + e_{interf} + e_{noise} + e_{artif} \tag{1}$$

- SDR: Source-to-distortion Ratio is a value that reflects the quality of a song [14].

$$SDR = 10log_{10}\left(\frac{\|s_{target}\|^2}{\|e_{interf}+e_{noise}+e_{artif}\|^2}\right) \tag{2}$$

-SAR: Source -to-artifact is the amount of unexpected sources that I don't want them to appear in my final product after the splitting process [14].

$$SDR = 10log_{10}\left(\frac{\|s_{target}+e_{noise}+e_{interf}\|^2}{\|e_{artif}\|^2}\right) \tag{3}$$

DOI: 10.54254/2755-2721/75/20240536

-SIR: In the accompaniment and vocal music files produced by my SVS model, they might contain music sources from each other, which means the sound of vocal in the accompaniment component is not cleared completely. Source-to-interference [15] Ratio is the index that shows the amount of the other source that was contained in either accompaniment part or vocal part [14].

$$SDR = 10log_{10}\left(\frac{\|s_{target}\|^2}{\|e_{artif}\|^2}\right) \quad (4)$$

Table 1. The results of evaluation based on SAR, SIR, and SDR

Songs	U-net neural model's Values	GRU-NRI[16]	SEMI-SUPERVISED MONAURAL[17]	GRU-RIS-S	GRU-RIS-L
bug_1_08.wav	sdr:6.45, sir:152.56, sar:6.45	-	-	-	-
21035_chorus.wav	sdr:36.02, sir:134.02, sar:36.02	-	-	-	-
54189_chorus.wav	sdr:23.81, sir:145.40, sar:23.81	-	-	-	-
Al James - Schoolboy Fascination..mp3	sdr:35.54, sir:110.84, sar:35.54	sdr:4.02, sir:136.51, sar:4.02	sdr:35.94, sir:101.27, sar:35.94	sdr:3.92, sir:126.34, sar:3.92	sdr:2.69, sir:120.45, sar:2.69
James Elder - The English Actor.mp3(0sec - 3sec)	sdr:29.01,sir:130.43, sar: 29.01	sdr: 12.87, sir: 142.92, sar: 12.87	sdr:26.50, sir: 132.78, sar: 26.50	Sdr:10.94, sir: 146.99, sar: 10.94	sdr: 13.38, sir: 148.41, sar: 13.38

As shown in **Table 1** After the evaluation about the sdr, sir, and sar value, the sdr and sar value of the product in produced by my model is much higher than the sdr and sar value produced by GRU-RIS-S, GRU-NRI, and GRU-RIS-L. Additionally, my method produced a higher sar and sdr than semi-supervised Monaural in the music example, "The English Actor". This implies that my U-net model SVS produced relatively high-quality separation products. Furthermore, by comparing sir values with other four methods, my method gets the smallest or the second smallest value of sir which indicates that our model produced separation audio that contains fewer interference.

Furthermore, as shown in **Figure 7, Figure 8,** and **Figure 9**, by comparing the waveform of the ground truth and the waveform of my product, two waveforms are very similar to each other which indicates that the vocals produced by my model are very close to the ground truth. However, in both **Figure 7** and **Figure 9**, my products contained contents in places that are originally empty in the ground truth, like the waveform from 0 to 20000 in **Figure 7.** This is probably because my model mistakenly extracted information from the spectrum that should be in other time slot of the audio. This implies that the model requires more data to adjust its parameter in the U-net model.

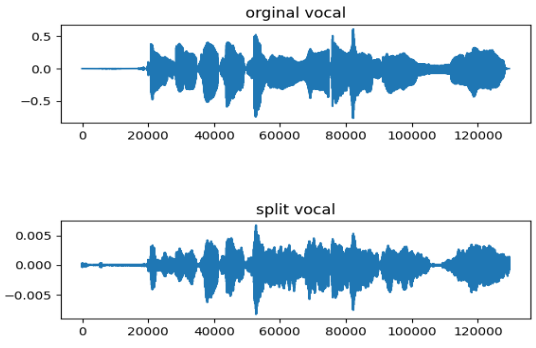

Figure 7. Wave pattern of bug_1_08-Vocal

DOI: 10.54254/2755-2721/75/20240536

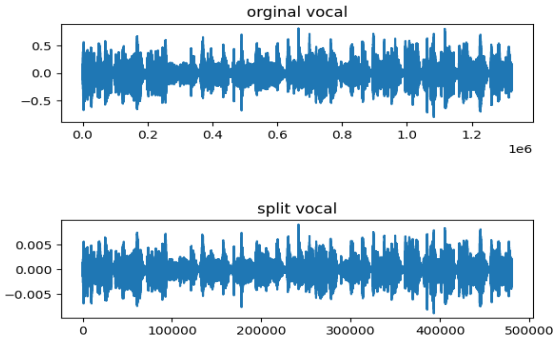

Figure 8. Wave pattern of 21035_chorus-Vocal

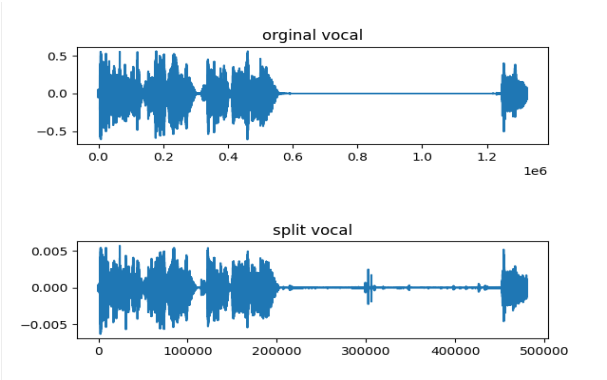

Figure 9. Wave pattern of 54189_chorus-Vocal

4. Conclusion

In this paper, I construct a website based on the U-net Neural Deep Learning Model, Flask framework, and Nginx server to achieve sing-voice separation of music pieces. The website allows users to access and use my trained model easily and efficiently. This helps the people who are interested in collecting and remixing music samples to get a vocal or accompaniment easily. The U-net Neural DL Model is a convolution chain that is trained using several datasets that contain abundant labeled data. The model gets a soft mask based on the input of an audio mixed signal, Moreover, I evaluated the outcome of the model by comparing the sir, sdr, and sar values with several other SVS methods, and the spectrum of the ground truth and our product. For future work, more datasets will be added into the training process to further improve the model, and the application is going to combine with Digital Audio Workstation and involve in music producing field.

References

[1] Kris, Wouk. “What Is a Digital Audio Workstation(DAW)?”gotogeek, 28 Dec 2022, www.howtogeek.com/853650/what-is-a-digital-audio-workstation-daw/

[2] Mark Marrington, ‘Composing with the Digital Audio Workstation’, in J.Williams and K.Williams (eds), *The Singer-Songwriter Handbook*(New York:Bloomsbury Academic, 2017), pp.77-89

[3] L. Prétet, R. Hennequin, J. Royo-Letelier and A. Vaglio, "Singing Voice Separation: A Study on Training Data," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, 2019, pp. 506-510, doi: 10.1109/ICASSP.2019.8683555.

[4] C. -L. Hsu and J. -S. R. Jang, "On the Improvement of Singing Voice Separation for Monaural Recordings Using the MIR-1K Dataset," in *IEEE Transactions on Audio, Speech, and*

DOI: 10.54254/2755-2721/75/20240536

Language Processing, vol. 18, no. 2, pp. 310-319, Feb. 2010, doi: 10.1109/TASL.2009.2026503.

[5] H. G. Okuno and K. Nakadai, "Computational Auditory Scene Analysis and its Application to Robot Audition," *2008 Hands-Free Speech Communication and Microphone Arrays*, Trento, Italy, 2008, pp. 124-127, doi: 10.1109/HSCMA.2008.4538702.

[6] Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999).LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* 521, 436–444 (2015). https://doi.org/10.1038/nature14539

[7] Mathew, A., Amudha, P., Sivakumari, S. (2021). Deep Learning Techniques: An Overview. In: Hassanien, A., Bhatnagar, R., Darwish, A. (eds) Advanced Machine Learning Technologies and Applications. AMLTA 2020. Advances in Intelligent Systems and Computing, vol 1141. Springer, Singapore. https://doi.org/10.1007/978-981-15-3383-9_54

[8] Metana Editorial. “The 10 Biggest Advantages of Deep Learning”, Metana, 24 Aug 2023, metana.io/blog/the-10-biggest-advantages-of-deep-learning/

[9] Chan, Tak-Shing, et al. "Vocal activity informed singing voice separation with the iKala dataset." 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2015.

[10] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam and J. P. Bello, "MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research", in 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan, Oct. 2014.

[11] Y. Weng, T. Zhou, Y. Li and X. Qiu, "NAS-Unet: Neural Architecture Search for Medical Image Segmentation," in IEEE Access, vol. 7, pp. 44247-44257, 2019, doi: 10.1109/ACCESS.2019.2908991.

[12] Grinberg, Miguel. Flask web development: developing web applications with python. " O'Reilly Media, Inc.", 2018.

[13] Reese, Will. "Nginx: the high-performance web server and reverse proxy." Linux Journal 2008.173 (2008): 2.

[14] Ethan Manilow, Prem Seetharaman, Justin Salamon. “Evaluation”, 2020, source-separation.github.io/tutorial/basics/evaluation.html

[15] M. M. Nandakumar and K. E. Bijoy, "Performance evaluation of single channel speech separation using non-negative matrix factorization," 2014 IEEE National Conference on Communication, Signal Processing and Networking (NCCSN), Palakkad, India, 2014, pp. 1-4, doi: 10.1109/NCCSN.2014.7001159.

[16] Michelashvili, Michael, Sagie Benaim, and Lior Wolf. "Semi-supervised monaural singing voice separation with a masking network trained on synthetic mixtures." ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019.

[17] Mimilakis, Stylianos Ioannis, et al. "Monaural singing voice separation with skip-filtering connections and recurrent inference of time-frequency mask." 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018.