Monaural speaker separation with supervised deep learning model using deep U-Net architecture and permutation invariant training – Egycsatornás, beszélő szerinti hangszétválasztás felügyelt, permutáció-invariáns tanítású, U-Net felépítésű mély neurális hálóval

Dávid Besenyei
Mechatronical Engineering BSc student
Budapest University of Technology and Economics
Budapest, Hungary
david.besenyei97@gmail.com

Zsolt Fischer
Mechatronical Engineering BSc student
Budapest University of Technology and Economics
Budapest, Hungary
zsolt@fisu.hu

Donát Takács
Mechatronical Engineering BSc student
Budapest University of Technology and Economics
Budapest, Hungary
takacs.donat01@gmail.com

Abstract

This study presents a solution for the well-known speech separation problem. The execution is carried out by means of a deep neural network with U-Net structure. The problem of speech separation has a rich literature. From the possible problems and difficulties, we present the case where two speakers' soundtracks are recorded by a single microphone. The goal of the project is to separate mixed speech and restore the audio channel of each original speaker.

Kivonat

Jelen tanulmány egy U-Net struktúrájú mély neurális hálón alapuló beszédszeparációs megoldást mutat be. A beszédszeparáció problémája széles szakirodalommal rendelkezik. Az előforduló problémák közül azt az esetet mutatjuk be, ahol két beszélő hangsávjait egyetlen mikrofon rögzíti és a cél az összekevert beszéd szétválasztása a két beszélő szerint.

1 Overview

Speech separation is the task of separating target speech from background interference. A special case of this task is multi-talker speaker-separation, where the interference is not noise, but an audio channel from another speaker. If there is only a single signal source available, the task is *monaural speaker separation*. While this task was traditionally treated as a signal processing problem, recent developments in the field [1] allowed for the use of deep neural networks, mostly by formulating the problem as a classification problem, which is well-posed for usage with deep neural networks.

In a general multi-talker case, let us consider S speakers, with *clean audio signals* $\mathbf{x}_1$, $\mathbf{x}_2 ... \mathbf{x}_S$. The *mixed signal* $\mathbf{y}$ is a composition of these signals. Thus the task of speaker separation is reversing this process: generating $\mathbf{x}_i$ from only $\mathbf{y}$. This is a problem that humans are quite good at solving: the so-called 'cocktail-party effect' can be observed by everyday life: in a noisy multi-talker environment, humans can successfully isolate and concentrate on the speech of a single individual. However, replicating this process using computers is still a very much open problem.

One recent, quite successful approach for treating this problem is the above-mentioned classification by deep neural networks. (An extensive, recent overview of state of the art results and practices in this field was published by Wang et al. in 2018 [2] which was used heavily during our work.) By calculating the (complex-valued) Short Time Fourier Transform (STFT) of the signals (denoted by $\mathbf{X}_i$ and $\mathbf{Y}$ respectively), time-frequency domain spectra can be obtained. By applying *masks* $\mathbf{M}_i$ on the mixed $\mathbf{Y}$ spectrum, the original $\mathbf{X}_i$ channels are aimed to be obtained:

$$\mathbf{X}_i = \mathbf{M}_i \circ \mathbf{Y} \tag{1}$$

where $\circ$ denotes element-wise multiplication. However, since speaker-specific data is mostly contained in the magnitude values, a common approach is to use

$$|\mathbf{X}_i| = \mathbf{M}_i \circ |\mathbf{Y}| \tag{2}$$

in separation, and reconstruct $\mathbf{X}_i$ from the above magnitude and the phase of $\mathbf{Y}$:

$$\mathbf{X}_i = |\mathbf{X}_i| \circ \mathrm{e}^{\mathrm{j}\ \arg \mathbf{Y}} \tag{3}$$

where the power is element-wise and j is the imaginary unit.

This way, a deep neural network h with parameters θ can be used to estimate masks for each speaker:

$$\left\{ \hat{\mathbf{M}}_i \right\} = h(|\mathbf{Y}|\,; \theta) \tag{4}$$

which is indeed a classification problem: the network needs to estimate for each time-frequency value how much it belongs to each speaker. Often a softmax restriction is used on the masks (like in [3]): $\sum_i^S \hat{\mathbf{M}}_i = \mathbf{I}$. While this is used for magnitude masks as well as complex masks, it is not an exactly correct approach in the former case, since $|\mathbf{Y}| \neq \sum_i^S |\mathbf{X}_i|$.

It is worth mentioning that this approach of generating masks and then reconstructing signals is considered superior to directly generating separated $\mathbf{X}_i$ spectra [1].

2 Dataset

Because none of the datasets used in the above literature was available to us, we generated our own dataset from a free audiobook downloaded from `https://librivox.org/`. The audio chosen was a reading of Charles Dickens' *Hard Times*. The choice was motivated by the fact that this reading is more than 10 hours long, has ten different readers and is in a lossless WAV format. We tried to mimic the structure of the datasets used in [3]: the audio was split to utterances 5 seconds long, and the Short Time Fourier Transform (STFT) of the individual utterances was calculated, with 512 FFT bins and a hop length of 256 (93 ms frame size at our sampling rate of 22050 Hz). This resulted in $T = 431$ time bins and $F = 257$ frequency bins. From these complex-valued STFT spectra we generated about 7 hours of total input data by randomly mixing spectra from different speakers. The result set was split into training, validation and testing sets with a 90%-5%-5% ratio.

We could have been able to generate more data, but with the size of RAM available to us during data generation this was the upper limit. The resulting dataset was a 17 GB file in `HDF5` format.

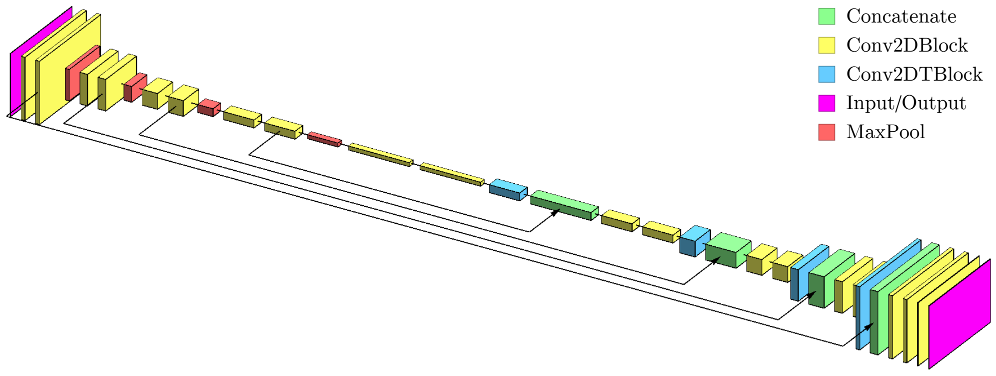

Figure 1: The architecture of our final model. (The dimensions of the blocks correspond to the tensor shapes output by each layer.)

3 Model and training

We based our two-speaker ($S = 2$) model on state the art best practices laid out in [2] and [3] discussed above. The input of the network is the $(257, 431, 1)$ sized $\mathbf{Y}$ STFT magnitude spectra of the mixed audio signal, and the output is a $(431, 257, 2)$ tensor containing the two masks $\mathbf{M}_1$ and $\mathbf{M}_2$. This choice of using entire 5 sec segments is a different approach from that of [3], where the neural network only operated on much shorter meta-frames that had a shifting window. We chose this approach on the grounds that the separation might be easier if the model has entire sentences as its input, as opposed to fragments of words.

Unlike in e.g. [3] or [4], we did not use any densely connected layers in our neural network. Neither did was a recurrent network used, like in [5] or [6]. Rather, we used a static, modified U-Net [7] architecture, which is illustrated on Figure 1. This architecture is universally used in image classification problems, but is not really used in speaker separation problems. In our model, the 2D convolutional blocks come in pairs: the first one doubles or halves the filter size, while the second keeps it the same. Both have kernels size (3,3) and stride (1,1), with padding to preserve the other dimensions of the tensors. The filter sizes increase from 1 to 8 in one step, then to 256 by doubling the filter size on the contracting side, then on the expansive side decrease to 16 by halving the filter size, which is finally reduced to depth 2. The max pooling layers all have pool size (2,2), thus effectively halving the spatial dimensions of their input tensors. Each 2D convolutional block has its input batch normalized, and ReLU activation is used on their outputs. In the case of the transposed convolutional layers on the expansive side, some zero padding was needed for proper concatenation. On the output of the last layer, a softmax activation was used. We implemented the model in Keras, and used a Tensorflow backend.

The training target was (as in [3]) the mean squared error of the separated magnitude spectra reconstructed using the estimated masks:

$$J = \frac{1}{T \times F \times S} \sum_{i=1}^{S} \left\| |\mathbf{X}_i| - \hat{\mathbf{M}}_i \circ |\mathbf{Y}| \right\| \tag{5}$$

where $\|.\|$ denotes the Frobenius-norm of a matrix. It is important to note that this training target is superior to simply training for an ideal mask, since in silent segments, the perfect mask is underdetermined. Because of this, the value of the masks is irrelevant during these time periods, and the actual audible output should be considered, as in the above J loss.

The final loss was calculated in a permutation-invariant way, which was the main idea of the *permutation invariant training* (PIT) presented in [3]. Simply put, a long-standing problem in supervised speech separation training was that the network had to learn which of its outputs are compared to which speaker channel during loss calculation. This makes the model unnecessarily hard to train. When using PIT instead, the loss is calculated for every possible permutation of pairs between output and reference channels. We had to implement the permutation-invariant loss function and the (5) calculation in Keras ourselves.

The training of the final model took about 10 hours on an Nvidia GTX 1070 GPU. The Adam optimization algorithm was used with early stopping based on the validation set results.

4 Hyperparameter-tuning

The final model was the result of a considerable amount of hyperparameter-tuning. First we based our model on those of [3] and [4]: a contractive convolutional part followed by densely connected layers. These models performed poorly in our experience on our 5 sec long audio segments: we suspect that this is at least partly due to the fact that the last layers destroy the spatial structure output by the previous convolutional layers. Also, these models have an unnecessarily large number of parameters. On our second try we tried a fully convolutional network like the final model, but with no skip connections. The results from this network was better (around 7 dB SDR), but adding the skip connections and thus building an U-Net-like architecture proved the best of all approaches. We trained the network every time with and without a final softmax activation layer based on the considerations discussed above. In the final model it did not affect the training loss values significantly, but the SDR values during testing were better by about 1.5 dB when using the softmax layer, and the subjective results were also much better. Because of this, we chose to include the softmax layer in our final network. This result was a bit surprising given that assuming sum of the magnitude-masks to be identity is conceptually flawed, as discussed above. A more detailed hyperparameter-tuning was not possible for us due to the long training time of the model.

5 Test results

The performance of the trained model was evaluated on the separate test set generated previously. An example result is illustrated on Figure 2 with spectrograms. For an objective measure the signal-to-distortion ratio (SDR) [8] was calculated for the test set. For reference, the SDR values of the mixed (non-separated) signals and signals separated using an Ideal Ratio Mask (IRM) [1] were also computed: these can be considered the two possible extremes in performance. The average SDR of our model's output was $9.24\,\mathrm{dB}$ which is quite good considering the reference values $0.12\,\mathrm{dB}$ and $12.02\,\mathrm{dB}$. While [3] used a different dataset, our SDR values compare well to their results. A non-representative, subjective auditory evaluation was also performed on the separated channels output by our model: while the separated signals are not perfect, none of the test subjects had any problem understanding the separated speech signals, which was not true for the mixed (non-separated) signals.

6 Speaker tracing

The previously discussed, improved loss function makes training the model much more successful, but introduces the problem that the network outputs might switch output channels between utterances. However, this can be eliminated using a traditional speaker-tracing algorithm. This was only alluded to in [3], but not realized. We implemented an algorithm for this based on [9]. 20 Mel Frequency Cepstrum Coefficients were used as features, from which a set of feature vectors are calculated for each reconstructed speaker channel at every 5 sec segment:

$$\mathbf{s}_i^j = \mathrm{MFCC}(\hat{\mathbf{x}}_i^j), \quad i = 1, ..., S,\ j = 1, ..., N \tag{6}$$

where N is the number of 5 sec segments that the original audio was separated into. The speaker tracing was done by calculating the negative generalized likelihood ratio for every possible permutation of outputs for consecutive segments. Considering the two-speaker case for simplicity, from four feature sets $\mathbf{s}_1^j$, $\mathbf{s}_2^j$, $\mathbf{s}_1^{j+1}$ and $\mathbf{s}_2^{j+1}$ and their pairwise unions

$$\mathbf{s}_{11}^j = \mathbf{s}_1^j \cup \mathbf{s}_1^{j+1} \quad \mathbf{s}_{12}^j = \mathbf{s}_1^j \cup \mathbf{s}_2^{j+1} \tag{7}$$

$$\mathbf{s}_{21}^j = \mathbf{s}_2^j \cup \mathbf{s}_1^{j+1} \quad \mathbf{s}_{22}^j = \mathbf{s}_2^j \cup \mathbf{s}_2^{j+1} \tag{8}$$

$$j = 1, ..., N-1$$

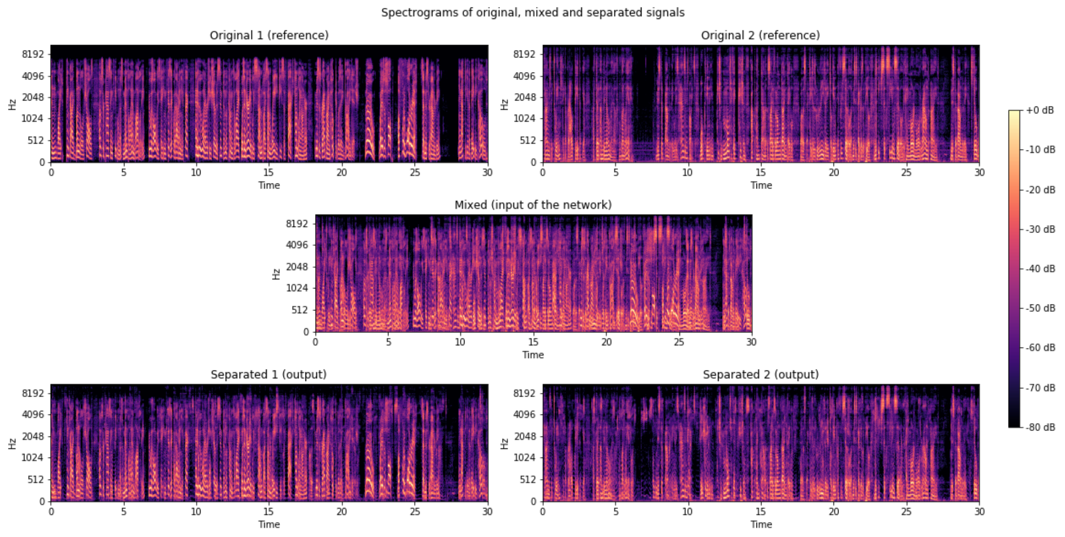

Figure 2: A separation example with spectrograms as an illustration of the process.

multivariate Gaussian distributions $\mathbf{s}_k^j \to (\boldsymbol{\mu}_k^j; \boldsymbol{\Sigma}_k^j)$ with $k \in \{1, 2, 11, 12, 21, 11\}$ are estimated. The generalized likelihood ratio between a certain pairing of two consecutive outputs is:

$$\mathrm{GLR}_{i,k}^j = \frac{L(\mathbf{s}_{ik}^j | (\boldsymbol{\mu}_{ik}^j; \boldsymbol{\Sigma}_{ik}^j))}{L(\mathbf{s}_i^j | (\boldsymbol{\mu}_i^j; \boldsymbol{\Sigma}_i^j)) L(\mathbf{s}_k^{j+1} | (\boldsymbol{\mu}_k^{j+1}; \boldsymbol{\Sigma}_k^{j+1}))} \quad i, k = 1, ..., S, \; j = 1, ..., N-1 \tag{9}$$

where L is the likelihood of the data given the Gaussian model. This value gives the likelihood ratio of the two segments being generated by the same process compared to being generated by two different processes (e.g. being from the same speaker versus from two different speakers). In practice, the log likelihoods $\log(\mathrm{GLR}_{1,1}^j)$ are used, since the likelihoods have very small numerical values for such high dimensional distributions, and the division can result in significant numerical errors. Considering again the two speaker-case, for the two possible pairings of outputs two distance metrics can be constructed based on the above $\mathrm{GLR}_{i,k}^j$ values:

$$d_{same}^j = -\log(\mathrm{GLR}_{1,1}^j) - \log(\mathrm{GLR}_{2,2}^j) \tag{10}$$

$$d_{switch}^j = -\log(\mathrm{GLR}_{2,1}^j) - \log(\mathrm{GLR}_{1,2}^j) \tag{11}$$

If $d_{same}^j > d_{switch}^j$, the outputs of the network were switched between segments j and $j+1$, so these channels have to be switched during the reconstruction of the full-length audio channels.

7 Conclusion and perspectives

We used an U-Net-like deep neural network architecture for monaural two-speaker separation. This approach was not found in relevant literature, but showed promising and usable results, while there is still room for improvement. Training on a larger dataset, preferably the WSJ0 [10] dataset used in [3] would be necessary for objective comparison of our model to the state of the art. Evaluating the model on unknown speakers or languages would be an interesting new angle, as well as varying the relative signal levels of the channels before constructing the mixed signals. A more detailed hyperparameter-optimization would also be in order. Using not only the magnitude but also the phase spectra during separation should also be investigated: it would probably further improve model performance.

Acknowledgments

We would like to thank Dr. Tamás Gábor Csapó from TMIT for the consultations before submitting our work for milestones, as well as Áron Ritter and Bence Sipos for the insightful discussions and kindly provided GPU-time.

References

[1] Y. Wang, A. Narayanan, and D. Wang. On training targets for supervised speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1849–1858, Dec 2014.

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

[3] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. *CoRR*, abs/1607.00325, 2016.

[4] Pritish Chandna, Marius Miron, Jordi Janer, and Emilia Gómez. Monoaural audio source separation using deep convolutional neural networks. volume 10169, pages 258–266, 02 2017.

[5] C. Weng, D. Yu, M. L. Seltzer, and J. Droppo. Deep neural networks for single-channel multi-talker speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(10):1670–1679, Oct 2015.

[6] Yanmin Qian, Xuankai Chang, and Dong Yu. Single-channel multi-talker speech recognition with permutation invariant training. *Speech Communication*, 104, 07 2017.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[8] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469, July 2006.

[9] D. Lilt and F. Kubala. Online speaker clustering. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–333, May 2004.

[10] E. Vincent et al. CHiME2 WSJ0 LDC2017S10. Web Download. Philadelphia: Linguistic Data Consortium, 2017.