Development of a sound source separation system based on machine learning methods

Tkachuk Orest

Lviv Polytechnic National University

Main topics in presentation

- General overview of the problem
- Methodologies and data description for sound separation
- Development of sound source separator

General overview of the problem

- **Relevance of the topic**

Sound source separation is a critical challenge in various applications such as music production and speech recognition. Developing effective machine learning-based sound source separation systems using audio processing programs is essential to address this challenge.

- **Main idea**

Having recorded a polyphonic composition, we want to separate individual sound sources that present in it.

- **Object of study**

The audio signal processing using neural networks.

- **Goal**

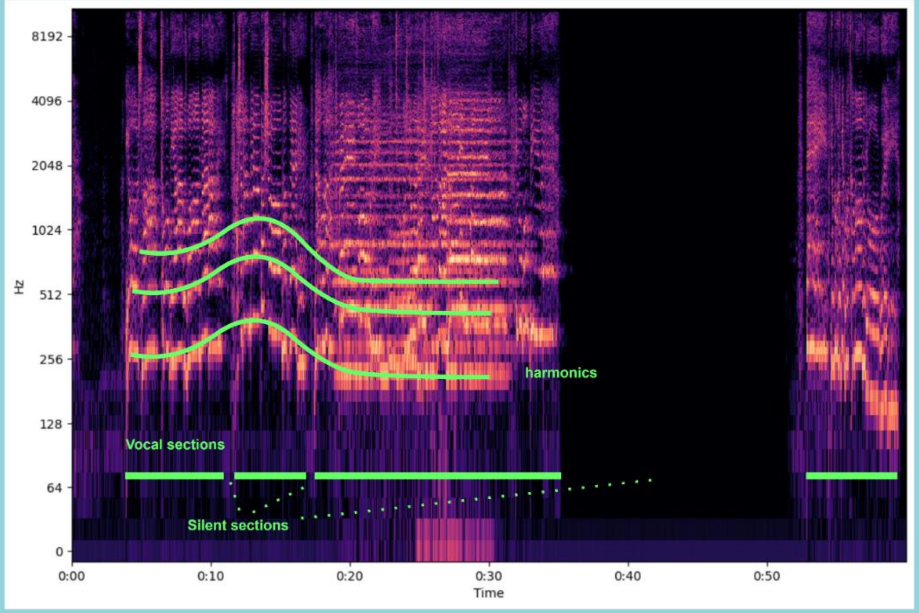
Study of the application of convolutional neural networks to improve signal processing and improve the process of extracting individual sound sources from polyphonic compositions.

- **Practical value**

Solving the task is vital for media production. This technology simplifies audio and video editing, allowing independent sound component editing. It also could be used for enhancing speech clarity in telecommunications.

Methodologies and data description for sound separation

Short-time Fourier transform



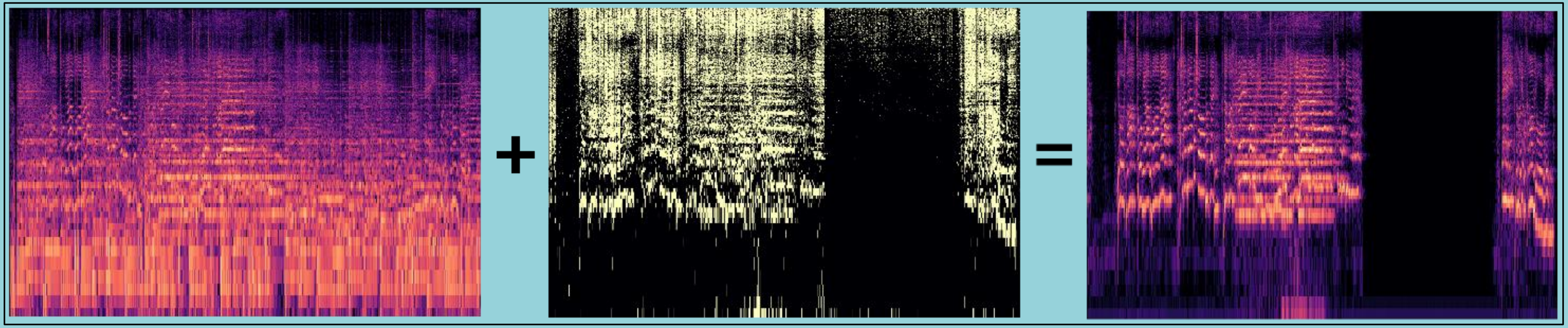
- The STFT splits the audio signal into small time windows and calculates a spectrogram for each, showing the energy distribution across frequencies at different moments in time. This provides information about the frequencies present in the audio signal and how they change over time.
- This information is important because different instruments and voices can have different frequency responses.
- STFT is ideal for CNNs as it captures both time and frequency information simultaneously. This allows the CNN model to detect complex patterns in sound components, enhancing its ability to learn from the data.

Binary masks

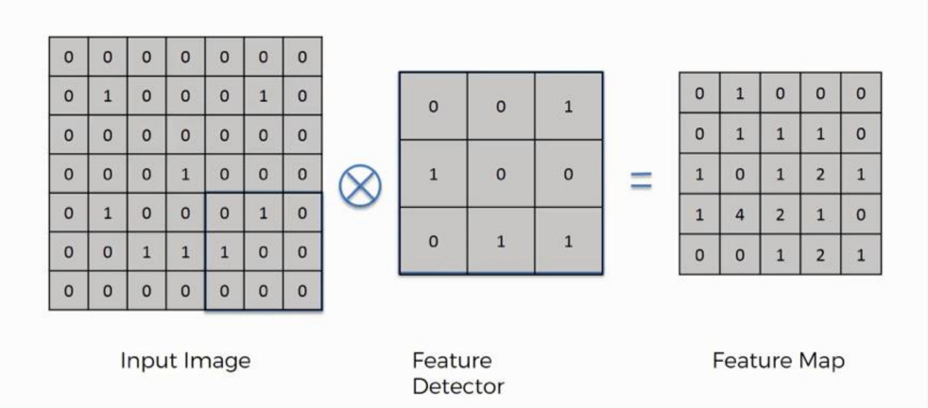
- There exists a simple method for separating audio sources that uses binary masks. The basis of this method is the preliminary creation of masks for various sound sources.

- This method is simple, and has a number of drawbacks. It involves the use of pre-existing masks, which may not be very reliable, especially if the audio recording conditions change.

$$B = (S > T * M)$$



Convolutional Neural Networks



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input Image

⊗

0	0	1
1	0	0
0	1	1

Feature Detector

=

0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

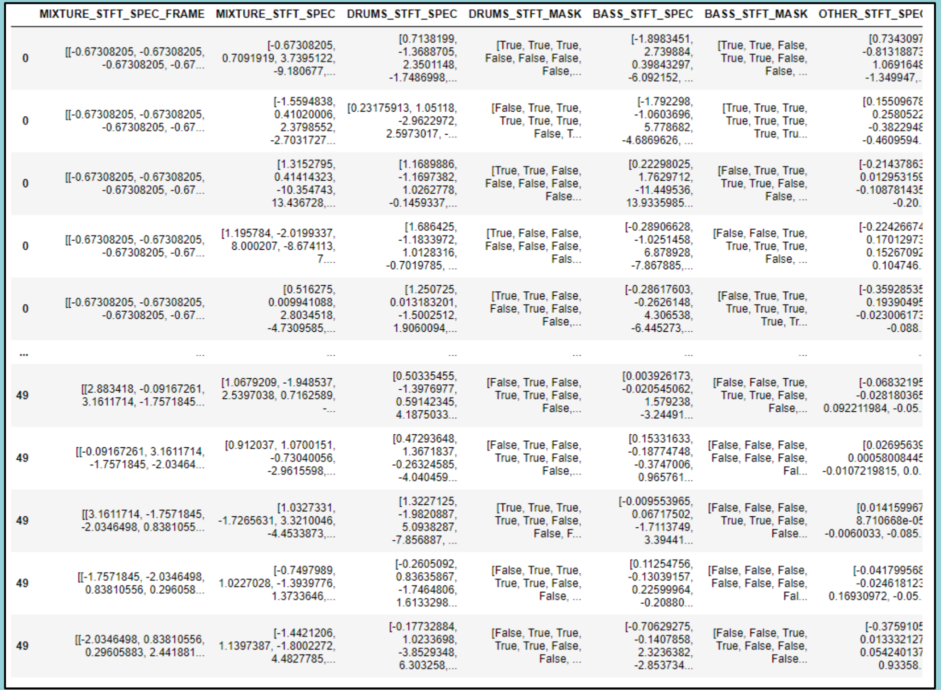
Feature Map

- Convolutional neural networks are effective in identifying and recognizing spatial patterns in input data. Their main idea is to use convolutions to extract important features from the input signal. This makes them particularly useful for image and audio processing where spatial structure is important.

- In cases of audio signal processing, CNNs can effectively recognize acoustic patterns and structures, providing the ability to automatically extract audio features.

- One of the strengths of CNNs is their ability to automatically extract hierarchical features from input data. However, they may be less efficient in cases where the temporal aspect of the input data is important.

Data for training

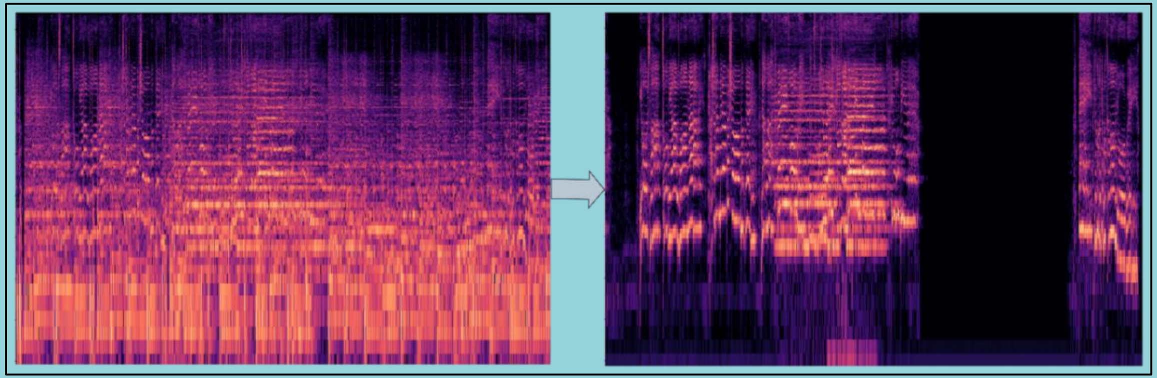


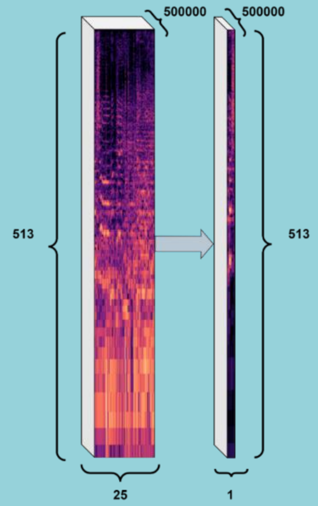
	MIXTURE_STFT_SPEC_FRAME	MIXTURE_STFT_SPEC	DRUMS_STFT_SPEC	DRUMS_STFT_MASK	BASS_STFT_SPEC	BASS_STFT_MASK	OTHER_STFT_SPE
0	[[-0.67308205, -0.67308205, -0.67308205, -0.67...	[-0.67308205, 0.7091919, 3.7395122, -9.180677,...	[0.7138199, -1.3688705, 2.3501148, -1.7486998,...	[True, True, True, False, False, False, False,...	[-1.8983451, 2.739884, 0.39843297, -6.092152, ...	[True, True, False, True, True, False, False, ...	[0.7343097 -0.81318873 1.0691648 -1.349947,.
0	[[-0.67308205, -0.67308205, -0.67308205, -0.67...	[-1.5594838, 0.41020006, 2.3798552, -2.7031727...	[0.23175913, 1.05118, -2.9622972, 2.5973017, -...	[False, True, True, True, True, True, False, T...	[-1.792298, -1.0603696, 5.778682, -4.6869626, ...	[True, True, True, True, True, True, True, Tru...	[0.15509678 0.2580522 -0.3822948 -0.4609594.
0	[[-0.67308205, -0.67308205, -0.67308205, -0.67...	[1.3152795, 0.41414323, -10.354743, 13.436728,...	[1.1689886, -1.1697382, 1.0262778, -0.1459337,...	[True, True, False, False, False, False, False...	[0.22298025, 1.7629712, -11.449536, 13.9335985...	[False, True, True, True, True, False, False, ...	[-0.21437863 0.012953159 -0.108781435 -0.20.
0	[[-0.67308205, -0.67308205, -0.67308205, -0.67...	[1.195784, -2.0199337, 8.000207, -8.674113, 7....	[1.686425, -1.1833972, 1.0128316, -0.7019785, ...	[True, False, False, False, False, False, Fals...	[-0.28906628, -1.0251458, 6.878928, -7.867885,...	[False, False, True, True, True, True, False, ...	[-0.22426674 0.17012973 0.15267092 0.104746.
0	[[-0.67308205, -0.67308205, -0.67308205, -0.67...	[0.516275, 0.009941088, 2.8034518, -4.7309585,...	[1.250725, 0.013183201, -1.5002512, 1.9060094,...	[True, True, False, False, True, False, False,...	[-0.28617603, -0.2626148, 4.306538, -6.445273,...	[False, True, True, True, True, True, True, Tr...	[-0.35928535 0.19390495 -0.023006173 -0.088.
...	...	...	...	...	...	...	.
49	[[2.883418, -0.09167261, 3.1611714, -1.7571845...	[1.0679209, -1.948537, 2.5397038, 0.7162589, -...	[0.50335455, -1.3976977, 0.59142345, 4.1875033...	[False, True, False, True, True, False, False,...	[0.003926173, -0.020545062, 1.579238, -3.24491...	[False, False, True, True, True, False, False,...	[-0.06832195 -0.028180365 0.092211984, -0.05.
49	[[-0.09167261, 3.1611714, -1.7571845, -2.03464...	[0.912037, 1.0700151, -0.73040056, -2.9615598,...	[0.47293648, 1.3671837, -0.26324585, -4.040459...	[False, True, False, True, True, False, False,...	[0.15331633, -0.18774748, -0.3747006, 0.965761...	[False, False, False, False, False, False, Fal...	[0.02695639 0.00058008445 -0.0107219815, 0.0.
49	[[3.1611714, -1.7571845, -2.0346498, 0.8381055...	[1.0327331, -1.7265631, 3.3210046, -4.4533873,...	[1.3227125, -1.9820887, 5.0938287, -7.856887, ...	[True, True, True, True, True, False, False, F...	[-0.009553965, 0.06717502, -1.7113749, 3.39441...	[False, False, False, True, True, False, False...	[0.014159967 8.710668e-05 -0.0060033, -0.085.
49	[[-1.7571845, -2.0346498, 0.83810556, 0.296058...	[-0.7497989, 1.0227028, -1.3939776, 1.3733646,...	[-0.2605092, 0.83635867, -1.7464806, 1.6133298...	[False, True, True, True, True, False, False, ...	[0.11254756, -0.13039157, 0.22599964, -0.20880...	[False, False, False, False, False, False, Fal...	[-0.041799568 -0.024618123 0.16930972, -0.05.
49	[[-2.0346498, 0.83810556, 0.29605883, 2.441881...	[-1.4421206, 1.1397387, -1.8002272, 4.4827785,...	[-0.17732884, 1.0233698, -3.8529348, 6.303258,...	[False, True, True, True, True, False, False, ...	[-0.70629275, -0.1407858, 2.3236382, -2.853734...	[False, False, True, True, False, False, False...	[-0.3759105 0.013332127 0.054240137 0.93358.

- For this task the MUSDB18 dataset was used. This dataset contains training and test data with a distribution of 100 and 50 instances for each, respectively.
- Each composition contains a set of "stems" - which are separate audio tracks that represent 4 different sound sources, such as vocals, drums, bass and other instruments, as well as a stem with the pure overall mix for training.

Development of source separator

General problem



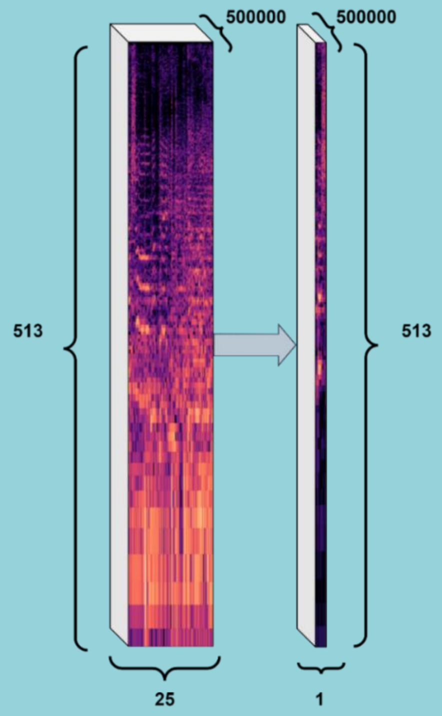


- This task consists in finding the necessary patterns of amplitude-frequency characteristics on the spectrogram of the general mix, highlighting these characteristics, and eliminating all that are not part of the source that needs to be isolated.

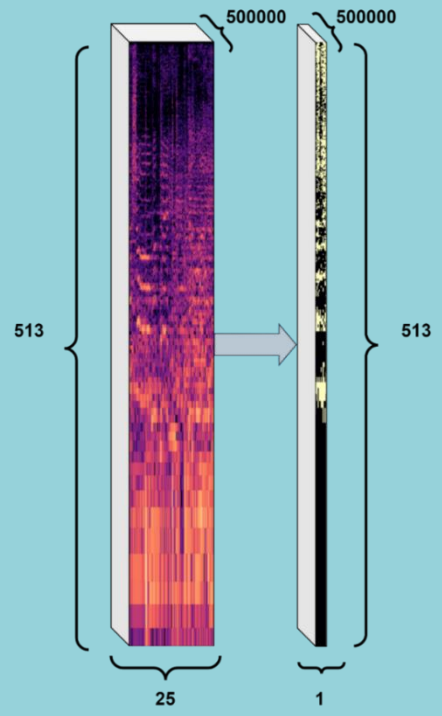
- With the use of CNN, it is planned to take frames of the total spectrogram that has 513 amplitude values with width of 25 and a step of 1, to preserve the temporal context, and predict 513 amplitude values in the middle of this window.

Simplifying the problem

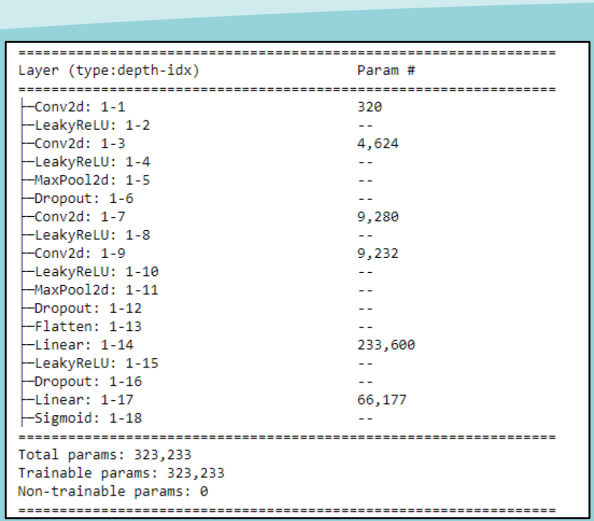
Without binary mask



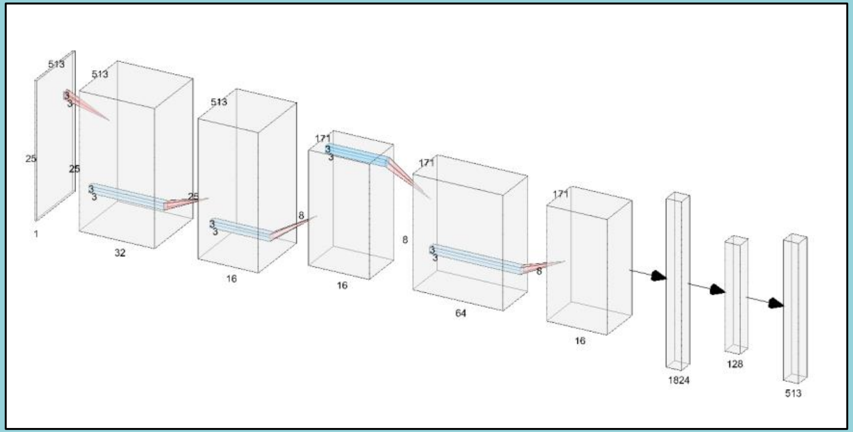
With binary mask



CNN Architecture



```
=================================================================
Layer (type:depth-idx)                   Param #
=================================================================
├─Conv2d: 1-1                            320
├─LeakyReLU: 1-2                         --
├─Conv2d: 1-3                            4,624
├─LeakyReLU: 1-4                         --
├─MaxPool2d: 1-5                         --
├─Dropout: 1-6                           --
├─Conv2d: 1-7                            9,280
├─LeakyReLU: 1-8                         --
├─Conv2d: 1-9                            9,232
├─LeakyReLU: 1-10                        --
├─MaxPool2d: 1-11                        --
├─Dropout: 1-12                          --
├─Flatten: 1-13                          --
├─Linear: 1-14                           233,600
├─LeakyReLU: 1-15                        --
├─Dropout: 1-16                          --
├─Linear: 1-17                           66,177
├─Sigmoid: 1-18                          --
=================================================================
Total params: 323,233
Trainable params: 323,233
Non-trainable params: 0
=================================================================
```



- This sound separation model is quite compact, as it consists of a small number of layers. The total number of parameters in this model is only 323,233.

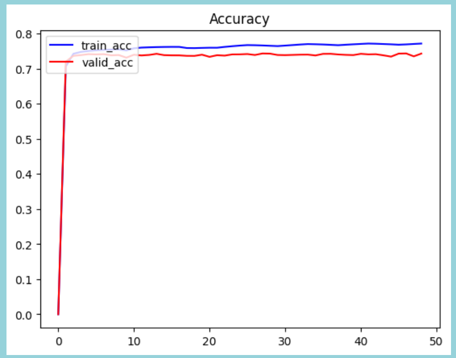
- The model has four convolutional layers. Each layer has 32/16/64/16 3x3 filters respectively. Convolutional layers are designed to detect different features in the input data, which helps to recognize certain features of the audio signal.

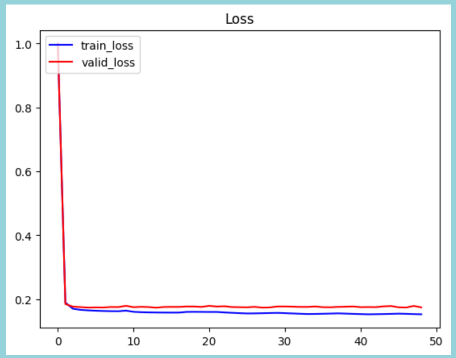
- The network also contains LeakyReLU activation functions to introduce non-linearity.

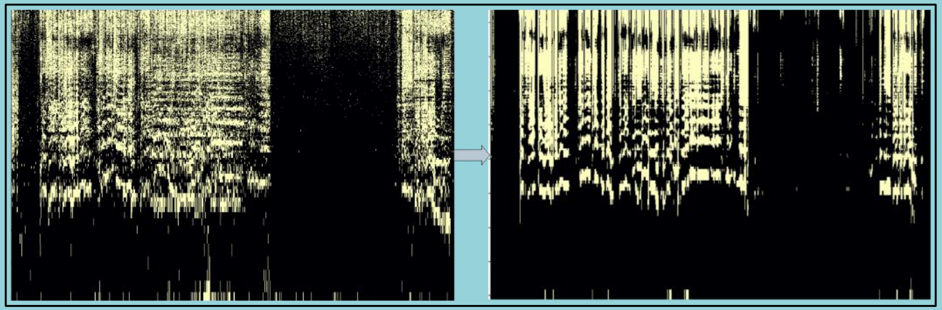
- After every two convolutional layers, a MaxPool2d pooling layer is used to reduce the dimensionality of the data. This layer helps summarize information and reduces the number of parameters in the network.

- The model also has dropout layers that randomly turn off certain neurons during training for model regularization.

Vocal isolation



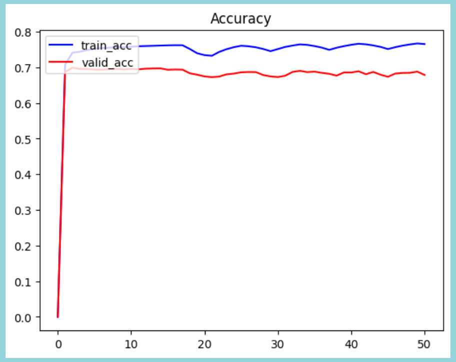


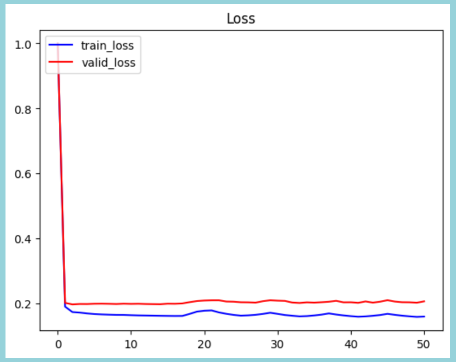


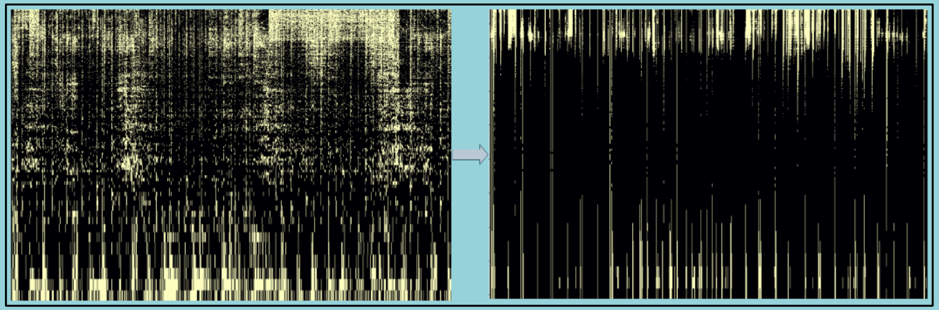
- The results show the prediction performance of the binary mask. The accuracy of 77.2% and 74.3% on the training and validation datasets, respectively, shows a good result, and the model itself can be continued to be trained further, since it still has the opportunity to learn. The results of the loss function are also small (0.153 and 0.174, respectively), which shows the model's confidence in its predicted results.

Vocal isolation	Accuracy	Loss (MSE)
Training	0.772	0.153
Validation	0.743	0.174

Drums isolation



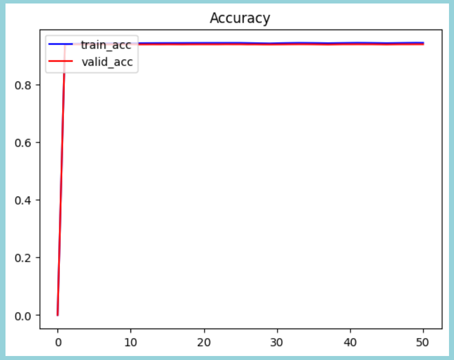


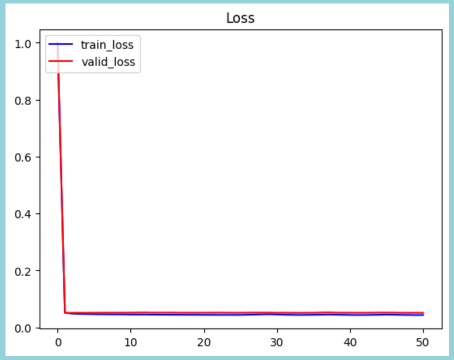


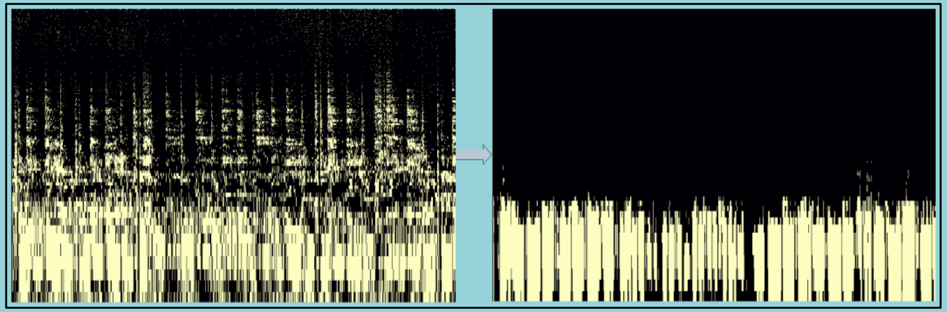
- Neural network training for drums isolation proved to be less efficient compared to vocals, but with still good accuracy results of 76.6% and 68.5% for the training and validation datasets, respectively. However, due to the specificity of the sound source, a limited number of redundant sounds were captured, including only the main ones for the given source.

Drums isolation	Accuracy	Loss (MSE)
Training	0.766	0.159
Validation	0.685	0.203

Bass isolation



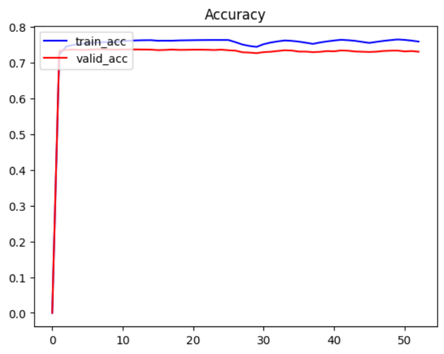


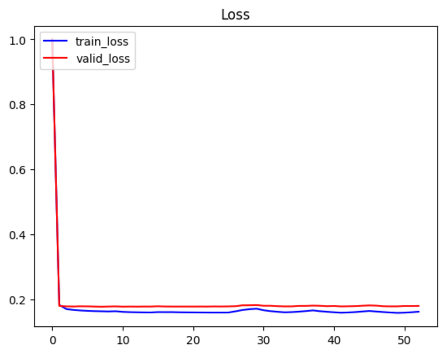


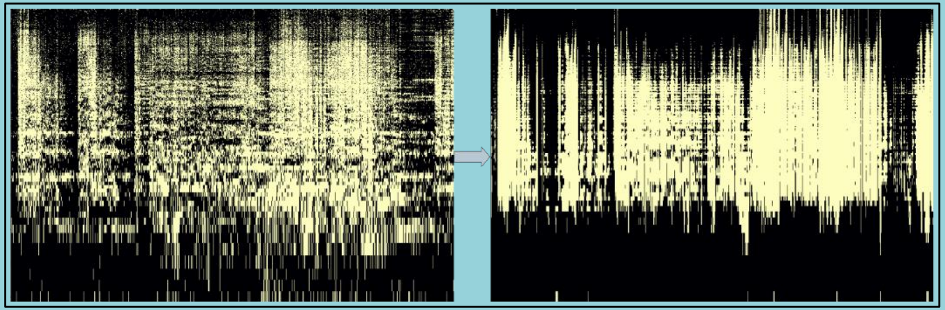
- Bass Isolation showed high accuracy results of 94.4% for the training and 93.9% for the validation data sets, with particular effectiveness in predicting the lower frequency range. However, limited effectiveness in working with the upper range was found. Despite this, its small effect on the original result is due to the fact that that the main acoustic events for this source are mainly located in the lower range.

Bass isolation	Accuracy	Loss (MSE)
Training	0.944	0.044
Validation	0.939	0.051

Other instruments isolation







- The prediction of other instruments showed similar results to the results of vocal isolation – 76.4% and 73.2%. The network was able to effectively predict the presence of other sources at specified time intervals. However, due to training on combined sources, the output is not optimal, although model coped with the task. This application can be effective for removing external sounds from the main mix that did not make it into the other sources.

Other instruments isolation	Accuracy	Loss (MSE)
Training	0.764	0.158
Validation	0.732	0.179

Conclusion

- The research aimed to isolate various audio signal sources, such as vocals, drums, bass, and other components, using convolutional neural networks. The idea was to use a short-time Fourier transform to represent audio signals in the time-frequency domain and process the resulting spectrograms with convolutional neural networks to detected and separated different sound sources. The use of binary masks instead of a direct STFT was chosen to simplify the problem and introduce a hybrid approach combining elements of regression and classification.

- The model successfully isolated various audio signal sources using convolutional neural networks with good accuracy, demonstrating the effectiveness of this method. Through the usage of a STFT and binary masks, different sound sources were effectively detected and separated. The usage of CNNs enabled the identification of acoustic patterns and properties corresponding to different sound sources in the input mix spectrograms.

Thank you for your attention