Using Recurrent Neural Network to Noise Absorption from Audio Files

Nataliya Boyko[1] and Anastasiia Hrynyshyn[1]

[1]*Lviv Polytechnic National University, Profesorska Street 1, Lviv, 79013, Ukraine*

Abstract

The study reveals the idea of noise absorption, which is reducing any noise from input signal with minimal distortion of speech. During the study and research of this topic, many articles and publications were analyzed, in which new approaches to solving the problem of noise absorption or modification of existing ones were presented. This paper considers noise absorption algorithms. Also, high-performance algorithms for noise and human speech separation in the audio stream are analyzed. The paper uses traditional algorithms for digital signal processing. The practical value of the results will help improve the quality of video and audio calls by eliminating background noise, as well as voice recognition. The paper uses classic solutions for filtering unwanted noise. Experiments were performed to compare three different methods of noise processing in audio files. Statistical methods are used to build a noise model, which is then used to recover the output sound of the input signal with noise. The study uses deep learning for comparison. STOI and PESQ scores are used to evaluate audio recordings obtained after noise removal.

Keywords

Artificial Intelligence, fourier transform, fast fourier transform, discrete fourier transform, Convolutional Neural Network, Recurrent Neural Network, Short-time objective intelligibility, mean square error

1. Introduction

Today there are many means of communication. The companion can be on the other side of the world, yet talking to them is not a problem for us, is it? There are situations when communication is impaired due to ambient background noise, as it is impossible to find a quiet place to talk. In this case, noise absorption algorithms are used.

There is traditional noise suppression – the introduction of two or more microphones [16]. The first microphone is located in the lower front of the phone, closest to the user's mouth, to directly capture their voice during a conversation. The second microphone is as far away from the first one as possible, usually on the top back of the phone.

Both microphones pick up ambient sounds. The microphone closer to the mouth captures more of the speaker's voices,while another – less. The software effectively separates them from each other, giving an almost clear "voice".

[1]CITRisk'2021: 2nd International Workshop on Computational & Information Technologies for Risk-Informed Systems, September 16–17, 2021, Kherson, Ukraine

EMAIL: nataliya.i.boyko@lpnu.ua (N.Boyko); anastasiia.hrynyshyn.knm.2018@lpnu.ua (A.Hrynyshyn)

ORCID: 0000-0002-6962-9363 (N.Boyko); 0000-0003-4289-9475 (A.Hrynyshyn)

This may sound easy, but there are many situations where such technology does not work. For example, when a person does not speak, so the microphones receive only noise, or actively shakes/turns the phone during a conversation, as during a run. Solving these problems is a complex process.

Traditional digital signal processing (DSP) algorithms [17] constantly try to find a noise pattern and adapt it. These algorithms work well in some cases; however, they don't scale to the variety and variability of noise in our everyday environment. That is why deep learning is used to solve this problem.

The relevance of the topic: There are many definitions of noise, but in general, it is background sounds caused by people, music, car buzzing, and more. These are primarily the sounds that should not be present in a conversation, video, or audio file. Noise distracts the audience's attention from the core material and therefore deteriorates the perception of information. But the main risk of noise for audio files is poor speech recognition. Many technologies work with voice commands, but due to excessive noise, the voice may be poorly recognized, due to which the program will not perform the correct task or could not receive the signal at all. Noise suppression is used to eliminate this risk.

The main idea of noise absorption is that the input was a signal with noise and the output without minimal speech distortion. This topic has been considered since the 70s. One example was the absorption of acoustic noise in a speech by spectral subtraction [18]. Although the research of this problem began a long time ago, the topic's relevance remains to this day since there is no perfect solution.

Having received a signal with noise at the input, we strive to filter out unwanted noise without degrading the input signal. There are classic solutions to this problem. First, they use generative modeling, which uses statistical methods like Gaussian filters to build a noise model. Next, we can use it to recover the output sound of the input signal with noise. But recently, developments have shown that deep learning is superior to that decision and provided enough data.

The work's goal is to increase noise absorption efficiency to reduce the risk of incorrect speech recognition and train the recurrent network on different types of noise.

The practical value of the results obtained in this work will help improve the quality of video and audio calls, eliminating background noise. This model will also reduce the risks of incorrect voice recognition caused by background noise.

2. Review of Literature Sources

During the study and research of this topic, I found many different articles and publications. Each of them represents a new approach to solving noise absorption or modification of existing ones. These materials are presented below with a brief analysis of the use of specific techniques.

First covered the idea of using deep neural networks in the article «A regression approach to speech enhancement based on deep neural network»[19], authored by Yong Xu, Jun Doo, Lee-Rong Dai, and Chin-Huel Lee. The basic idea is to use a regression method, which produces a mask of relations for each sound frequency. The purpose of this mask is to remove extraneous noise, leaving the human voice intact. This method was far from perfect but an excellent early solution.

After the publication of the idea using deep neural networks, various theories were proposed, one of which is using a recurrent neural network. This method was demonstrated in the RNNoise project. Combining classical signal processing with deep learning to create a real-time noise absorption algorithm is the main idea. A more detailed description is given in the article «A Hybrid

DSP/Deep Learning Approach to Real-Time Full-Band Speech Enhancement»[20], authored by Jean-Marc Valin.

Another exciting example of the use of neural networks for noise absorption was proposed in «Practical Deep Learning Audio Denoising»[21], authored by Thalles Santos Silva. This article used a convolutional neural network (CNN) to create a statistical model that can extract a pure signal and return it to the user. But in most results, the model manages to smooth out the noise, not get rid of it. Therefore, my choice was for recurrent neural networks.

The following article, authored by Michael Michelashvili, Lior Wolf, proposed a sound absorption method that trains on a noisy sound signal and provides a pure baseline signal [22]. However, the technique is not entirely controlled and is taught only on a specific audio file that is denominated. Disadvantages of this implementation: if the type of noise changes, the neural network, which was trained on other data, will not provide sound absorption.

Another method of teaching recurrent neural networks was proposed in the article: «Listening to Sounds of Silence for Speech Denoising»[23] by Ruilin Xu, Rundi Wu, Yuko Ishiwaka, Carl Vondrick, and Changxi Zheng. The proposed approach is based on the observation of human language, namely the pauses in speech between words and sentences. They are using these intervals to study the model. Since this algorithm studies noise in real-time, it is possible to learn the models of noise dynamics and absorb them. This method, in my opinion, is one of the best because it can adapt to noise changes in contrast to the previous one.

Noise absorption is not only used for audio and video calls. It is also used for hearing aids. An article entitled "Use of a Deep Recurrent Neural Network to Reduce Wind Noise: Effects on Judged Speech Intelligibility and Sound Quality" [24], written by Mahmoud Keshavarzi, Tobias Goehring, Justin Zakis Richard E. Turner та Brian C. J. Moore. It demonstrated the use of RNN to reduce wind noise, which added sound quality. Recurrent neural networks were significantly better than high-frequency filtering. Tested these results were with the help of eighteen participants, nine of whom had mild or moderate hearing impairments. According to them, the sound quality and intelligibility were much better when using RNN.

Analysis of the sources described above gave more information about using deep neural networks and various practical applications. In addition, multiple methods and modifications have also been developed, with the help of which noise absorption had a much better result.

3. Materials and Methods

The separation of noise and human speech in the audio stream is a complex problem for which there are no high-performance algorithms.

Traditional digital signal processing (DSP) algorithms [17] try to constantly find the noise pattern and adapt it by processing the sound frame by frame.

There are two types of basic types of noise: stationary and nonstationary. An example is shown below in Fig. 1.

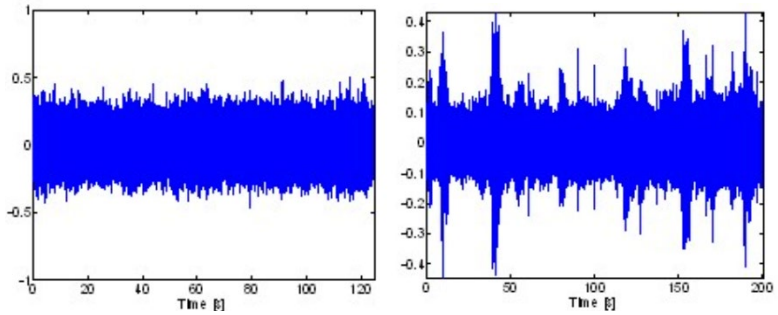


Figure 1: Two types of noise stationary (left) and non-stationary (right)

Stationary means that noise statistics regarding intensity, spectrum shape, or other factors are unchanged over time. Metaphorically speaking, stationary means that none of the statistical parameters of the process changes its position in the parameter space. Traditional DSP algorithms (adaptive filters) can be quite effective in filtering such noise. Let's take a closer look.

Digital Signal Processing (DSP) is a field of computer technology that is dynamically evolving and covers both hardware and software [25]. In particular, related areas for digital signal processing are information theory, optimal signal reception theory, and pattern recognition theory. In the first case, the main task is to select the signal against the background noise and interference of different physical nature; in the second - automatic recognition, i.e., classification and identification of the signal.

Digital processing uses the representation of signals in the form of sequences of numbers or symbols. The purpose of such processing may be to evaluate the signal's characteristic parameters or convert the signal into a format that is in some sense more convenient. For classical numerical analysis, formulas such as interpolation, integration, and differentiation are digital processing algorithms. High-speed digital computers contribute to increasingly complex and efficient signal processing algorithms; recent advances in integrated circuit technology promise high cost-effectiveness in building very complex digital signal processing systems.

Digital signal processing is an alternative to traditional analog. Its most critical qualitative advantages include implementing any arbitrarily complex (optimal) processing algorithms guaranteed and independent of destabilizing factors accuracy; programmability and functional flexibility; the possibility of adaptation to the processed signals; manufacturability.

The development of a new perspective on digital signal processing was accelerated by the discovery in 1965 of efficient algorithms for calculating Fourier transforms. This class of algorithms became known as fast Fourier transform (FFT).

3.1. Fast Fourier Transform

Fast Fourier transform (FFT) is a mathematical algorithm that calculates the discrete Fourier transform (DFT) of a given sequence [20]. The only difference between FT (Fourier transform) and FFT is that FT considers a continuous signal, while FFT receives a discrete signal at the input. DFT converts a sequence into its frequency components in the same way that FT does for a continuous signal. FFT converts the time domain to a frequency domain.

The visualization of the process is demonstrated below (Fig. 2).

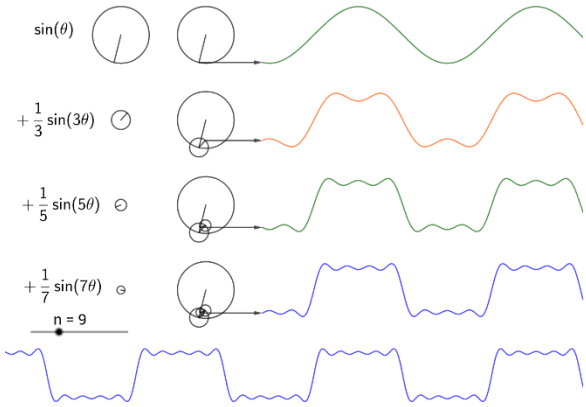


Figure 2: Geometric Fourier transform

FFT works as follows. In the first step, the signal portion is scanned and stored in memory for further processing. Two parameters are appropriate:

1. Sampling frequency (fs) of the measuring system (for example, 48 kHz). This is the average number of samples obtained per second.
2. Selected number of samples; block length (BL).

From the two main parameters fs and BL you can determine further measurement parameters. For example, bandwidth (fn) indicates the theoretical maximum frequency determined using FFT (Formula 1).

$$fn = fs / 2 \tag{1}$$

For example, at a sampling frequency of 48 kHz, it is theoretically possible to determine frequency components up to 24 kHz. However, in an analog system, the practically realizable value is usually slightly lower than this, thanks to analog filters - for example, at a frequency of 20 kHz.

Measurement of duration (D). The measurement duration is determined by the sampling frequency fs and the length of the block BL (Formula 2).

$$D = BL / fs\,, \tag{2}$$

where f_s = 48 kHz and BL = 1024 it gives 1024/48000 Hz = 21.33 ms.

Frequency resolution (df) indicates the frequency interval between two measurement results (Formula 3).

$$df = fs / BL \tag{3}$$

In practice, the sampling rate fs is usually a variable given by the system. However, by selecting the length of the BL block, you can determine the measurement duration and frequency resolution. The following applies:

- The short block length results in rapid repeats of measurements with coarse frequency resolution.
- Long block length results in slower repetitions of measurements with accurate frequency resolution.

3.2. Spectral subtraction

The method of spectral subtraction is widespread.

Additive stationary noise - generated by the environment, sound recording equipment, etc. Stationarity means that the properties of noise (power, spectral composition) do not change over time. Additivity implies that the noise is summed with the "pure" signal y [t] and does not depend on it (Formula 4):

$$x[t] = y[t] + noice[t], \tag{4}$$

where t is the time.

A spectral subtraction algorithm is used to suppress additive stationary noise. It consists of the following stages:

1. Signal decay by short-term (window) Fourier transform (STFT) compactly localizes the signal energy.
2. Assembling the noise footprint subtractor. The noise model is obtained by averaging the amplitudes of the spectrum taken from a pre-prepared area of noise that does not contain a proper signal (Formula 5).

$$footpr\,\mathrm{int}[f] = \sum_{t=1}^{k} noice[f,t], \tag{5}$$

where $noice[f,t]$ is the noise spectrum; f is the Fourier transform index corresponding to the frequency, t is the number of the current STFT window, k is the number of windows in the area with noise.

3. "Subtraction" (in the generalized sense) of the amplitude spectrum of noise from the amplitude spectrum of the signal.
4. Inverse conversion of STFT - synthesis of the resulting signal.

Subtraction of amplitude spectra is carried out by formula 6:

$$Y[f,t] = \max\{X[f,t] - k * W[f,t], 0\}, \tag{6}$$

where $X[f,t]$ and $W[f,t]$ - amplitude spectra of signal and noise, respectively;

$Y[f,t]$ - the amplitude spectrum of the resulting purified signal;

k is the suppression factor.

The phase spectrum of the cleared signal is equal to the phase spectrum of the signal interference. The result of this method is shown in Fig. 3.

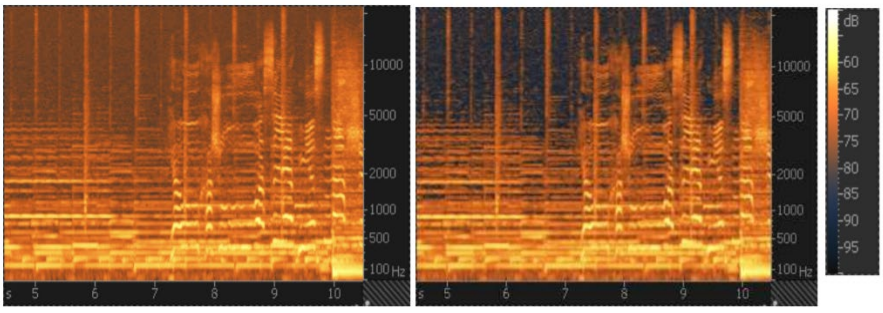


Figure 3: Spectrograms of noisy signal (shower) and cleared (right)

The problem with these methods is that FFT and spectral subtraction are not suitable for nonstationary signal analysis because nonstationary signals consist of frequency components that change over time. As is known, the Fourier transform is suitable for those signals that have frequencies fixed at a specific time (e.g., sine waves, voice signals). Therefore, the Fourier transform cannot give the proper spectrum, and we will not know which frequencies are present at what time. In spectral subtraction, the STFT coefficients of noise signals are statistically random, which leads to uneven noise elimination.

Nonstationary noises have complex patterns that are difficult to distinguish from the human voice. However, the signal can be concise and come and go very quickly (for example, keyboard input or siren). To handle both stationary and nonstationary noise, you need to go beyond traditional DSP.

To better eliminate noise, various methods of neural networks, some of which have been superficially discussed in the analysis of literature sources. Consider some of the methods in more detail.

3.3. Method using convolutional neural networks

This method is based on "A Fully Convolutional Neural Network for Speech Enhancement" [21]. In it, the author offers a cascade backup convolutional network encoder-decoder (CR-CED).

The model is based on symmetric encoder-decoder architectures. Both components contain repetitive convolution blocks, ReLU, and batch normalization. In total, the network includes 16 such blocks - this adds up to 33K parameters.

In addition, there are connection gaps between some encoder and decoder units. Here, the function vectors of both components are combined by addition. Like ResNets, bandwidth accelerates convergence and reduces gradient disappearance.

Another essential feature of the CR-CED network is that the convolution is performed in only one dimension. More specifically, given the input spectrum of the form (129 x 8), the convolution is performed only on the frequency axis (i.e., the first). This ensures that the frequency axis remains unchanged during forwarding propagation.

Combining a small number of learning parameters and model architecture makes this model extremely easy, with fast execution, especially on mobile devices.

Once the network evaluates the output, we optimize (minimize) the root mean square difference (MSE) between the output and target (pure sound) signals (Fig. 4).

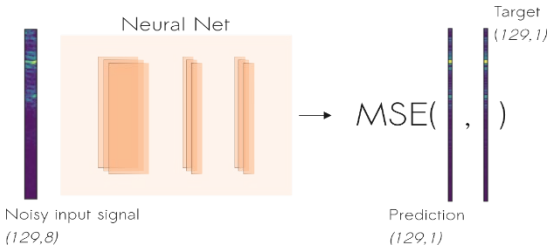


Figure 4: The principle of operation of the backup convolution network

The results of this method are presented in Fig. 5.

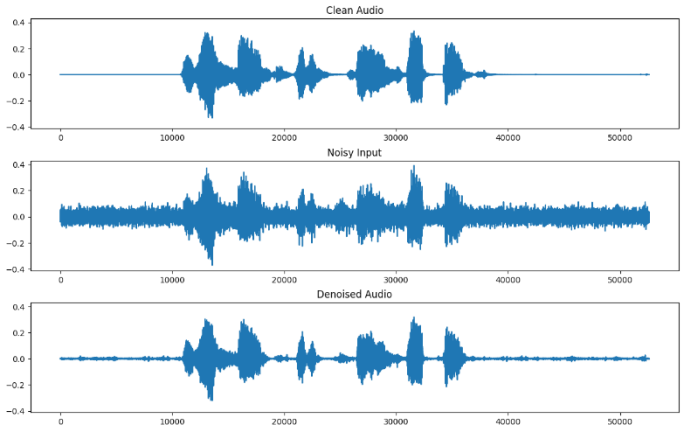


Figure 5: The results of methods using convolutional neural networks

Figure 5 shows the initial audio without noise, the audio to which the noise was added, and the result of processing the method. As you can see, given the complexity of the task, the results are somewhat acceptable but not perfect because the noise remained on this audio file.

3.4. Method using a recurrent neural network (GRU)

This method started with the removal of noise using artificial intelligence [19]. The method consists not only of in-depth training; it uses a hybrid approach. The central processing cycle is based on 20 ms windows with 50% overlap (10 ms offset). Use both analysis and synthesis of a Worbis window that satisfies the PrincenBradley criterion. The window is defined using the following formula 7:

$$w(n) = \sin[\frac{\pi}{2}\sin^2(\frac{\pi}{N}\Gamma)], \tag{7}$$

where N is the length of the window.

In fig. 7 shows a block diagram of this method.

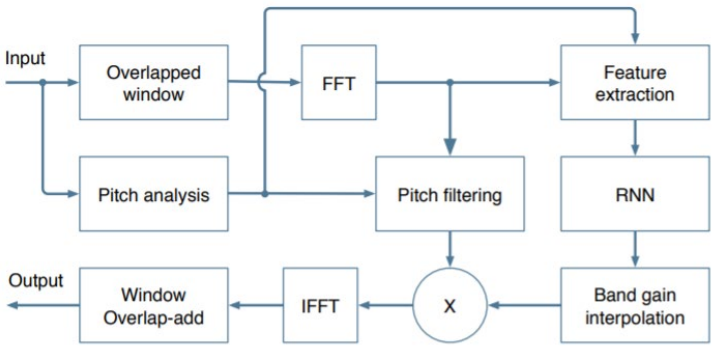


Figure 6: Block diagram of the method using a recurrent neural network

To avoid a huge number of outputs and, consequently, a large number of neurons, the algorithm does not work directly with the samples or with the spectrum. It considers the frequency bands that correspond to the Barca scale, corresponding to how we perceive sound. A total of 22 bands are used instead of 480 spectral values, which reduces the number of calculations. Let w_b *(k)* be the amplitude of the band *b* at the frequency *k* we have $\sum_b w_b(k) = 1$. For the converted signal *X (k)*, the energy value in the band is calculated by formula 8:

$$E(b) = \sum_k w_b(k)\left|X(k)\right|^2 \tag{8}$$

The gain in the band is defined as g_b.

$$y_b = \sqrt{\frac{E_s(b)}{E_x(b)}}, \tag{9}$$

where $E_s(b)$ is the energy of pure speech, $E_x(b)$ is the energy of the input (noisy) speech. Considering the ideal gain $\hat{g}_b$, the following interpolated gain is applied to each basket of frequencies *k* (formula 10):

$$r(k) = \sum_b w_b(k)\hat{g}_b \tag{10}$$

The main drawback of the lower resolution we get from using bands is that we do not have a fine enough resolution to suppress the noise between pitch harmonics.But this task is not essential and can be easily implemented with a comb filter.

Since the result we calculate is based on 22 bands, using a higher input resolution would make sense, so the same 22 bands are used to supply spectral information to the neural network [22].

To improve the preparation of data for training, DCT is used on the logs of the spectrum. At the output, we obtain 22 Cestral Barca frequency coefficients (BFCC). The data obtained is a bar current based on the Barca scale, closely related to the MFC coefficients, often used for speech recognition.

In addition to our cepstral coefficients, the following is also added:

- The first and second derivatives of the first 6 coefficients across frames
- The pitch period (1/frequency of the fundamental)
- The pitch gain (voicing strength) in 6 bands
- A special non-stationarity value that's useful for detecting speech (but beyond the scope of this demo).

This makes a total of 42 neural network input functions. The traditional approach to noise suppression inspires the neural network architecture used in this method. Most of the work is performed by three layers of GRU. Figure 7 shows the layers used to calculate the bands.

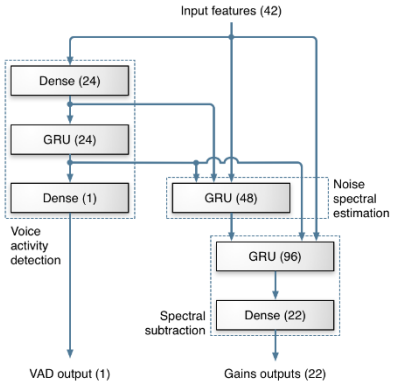


Figure 7: Scheme of neural network architecture

4. Experiments

We will conduct experiments comparing three different methods of noise processing in audio files. Two of them relate to algorithms using artificial intelligence, namely CNN and RNN.

The first algorithm to be used for comparison is spectral subtraction. The main steps of this algorithm:

- Calculate the FFT using an audio clip that contains noise
- Statistically calculate FFT by noise
- Calculate the threshold based on the statistical noise
- FFT is calculated by the signal
- The mask is determined by comparing the FFT signal with the threshold value
- The mask is smoothed by the filter by frequency and time
- The mask is applied to the FFT signal and inverted

To begin, download the data without noise (Fig. 8). Then, divide the data from the file by 32768 because the file we download has a wav extension. The data in it is in the range of [32768, 32767], so dividing by 32768, we get the appropriate degree of additions in two [-1, 1].

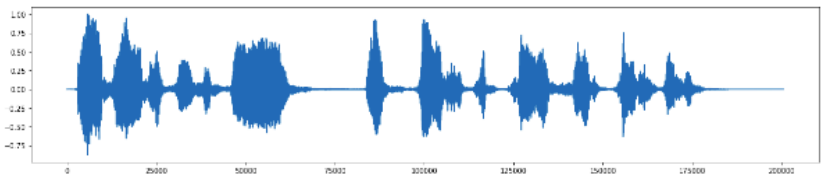


Figure 8: Noise-free signal

The next step is to add noise to the audio file (Fig. 9-10). The noise file also has a wav extension.

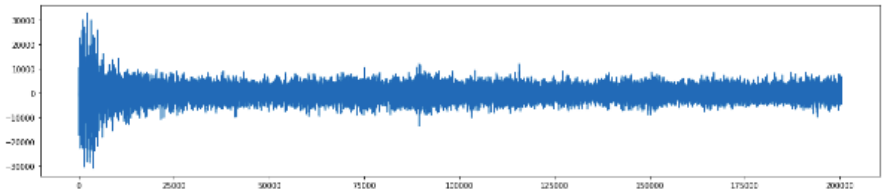


Figure 9: Noise signal

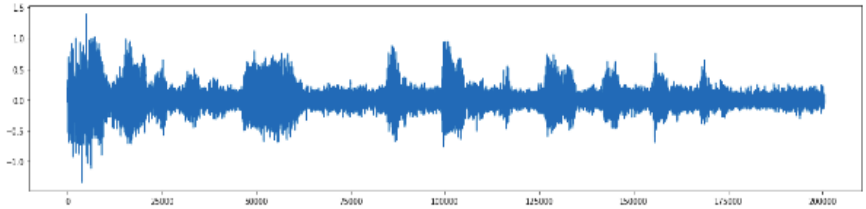


Figure 10: Signal after adding noise

Preparations for working with a noisy audio file are completed. Now let's start with the central part, namely the calculation of FFT. The STFT (Short Fourier Transform) function calculates the FFT of an audio file containing noise. Short-Term Fourier Transform (STFT) - is a Fourier-related transform used to determine the frequency of the sinusoidal and phase contents of local signal cross-sections as it changes over time. In practice, calculating the STFT is to divide the signal of a long time into shorter segments of the same length and then calculate the Fourier transform separately for each shorter part.

The STFT algorithm consists of the following steps:

- Select a data segment from the overall signal
- Multiply this segment by the semi-cosine function
- Zero the end of the segment with zeros
- Normalize the Fourier transform for this segment into positive and negative frequencies
- Combine the energy of positive and negative frequencies and return the one-way spectrum
- Scale the resulting spectrum in dB for easy viewing
- Record the signal to eliminate the noise beyond the noise threshold.

After performing the function, we obtain a complex-valued matrix of short-term Fourier transform coefficients. Reduce it to type dB and proceed to the next step.

We calculate noise statistics. Namely, we find the mean and standard deviation. Next, multiply the standard deviation by the change *n_std_thresh*. It shows how many standard deviations the sound must be considered a signal and not a noise. By default, the change has a value of 1,5. Add this value to the standard deviation.

Calculate the STFT for a non-noisy signal and also reduce it to the type dB. Now we create a mask, for this, we look for the minimum value of the complex matrix obtained in the previous step, and we create a smoothing filter for the mask by time and frequency. Calculate the threshold for each frequency interval.

Convert the mask using the smoothing filter *fftconvolve*. Convolution is a simple mathematical operation that requires the multiplication of vectors, so the complexity of execution is $O(n^2)$. But

to speed up the process, the convolution is performed with a fast Fourier transform. Using FFT, the complexity decreases from $O(n^2)$ to $O(nlog(n))$. The algorithm is presented in Fig. 11.

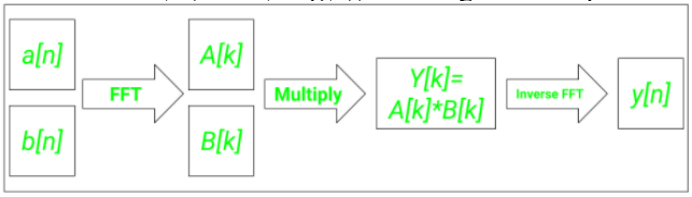


Figure 11: Schematic representation of the FFT workflow

After creating the mask, we proceed to the final stage, removing noise from the audio file. To do this, we use the inverse Fourier transform. The inverse transformation is when each subsequent window is returned to the time domain using IFFT. Then each window is shifted by the size of the step and added to the result of the previous shift. The following diagram represents this process.

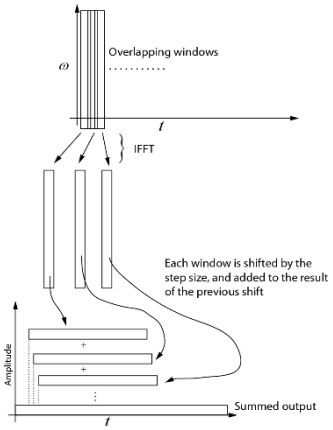


Figure 12: Schematic representation of the ISTFT workflow

And in the end, we return the received audio file in which noise decreased. It is presented in Fig. 13.

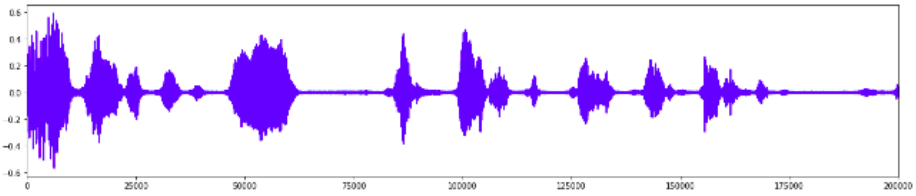


Figure 13: Audio file signal after noise cancellation

The following algorithm uses a convolutional neural network (CNN) to reduce noise in an audio file, the architecture of which consists of an encoder and a decoder with residual connections between pairs of layers.

The first step is to initialize the scales. Initializing the scales is an important step. If the scales are too small, then the dispersion of the input signal begins to decrease as it passes through each

layer in the network. As a result, the input eventually falls to shallow values and can no longer be valid. On the other hand, if the weights are too large, the variance of the input data tends to increase rapidly with each subsequent layer. Thus, initializing a network with suitable scales is very important for a neural network to work correctly. We need to make sure that the scales are within reasonable limits before we start training the net. That's why Xavier's initialization is used.

Xavier initialization is an initialization scheme for neural networks. Changes are initialized to 0, and the weight W_{ij} at each level is initialized as: $w_{ij} \approx \mathrm{U}[-\frac{1}{\sqrt{\mathrm{n}}}, \frac{1}{\sqrt{\mathrm{n}}}]$, where U is a uniform distribution and n is the size of the previous layer (number of columns in W).

In the second step, we initialize the vector z with random values from 0 to 1. The next step in obtaining a mask the size of STFT signals with values in the range [0,1], as the method's input, used the signal Y.

Once we have obtained the vector z the method goes through iterations. The number of iterations is set by changing t and the function is passed along with the audio file. Each iteration has the following steps. In the next step, the f_{i-1} network learns in one iteration, obtaining f_i. The following line calculates $f_i\ (z)$ and its STFT for each Y_i. Next, we find H_i value, the absolute difference between $|Y_i|$ and $|Y_{i-1}|$, and normalize the resulting difference with $|Y_i|$.

The following steps check the obtained value of H_i. To get rid of extreme values, it truncates all values below 10 and above 90. The value of C is the product of the matrices. C will have high values in the coordinates of the frequency-time domains, in which the lowest stability of recovery y over the network f.

After completing all iterations, the value of C is normalized to be in the range from 0 to 1. High accumulation of variability implies noise, and therefore flip the value ($max(C) - C$, not $C - min(C)$) before returning the mask M.

The method using recurrent neural networks uses a recurrent network with GRUs designed to overcome the noise in the audio recording. This neural network architecture is based on the assumption that there are three repeating layers, each responsible for one of the main components. It includes 215 units, 4 hidden layers, and the largest layer hides 96 units. Increasing the number of layers does not significantly improve the quality of noise absorption. However, the loss function and the way the training data is constructed substantially influence the final grade.

One of the essential parts of learning is the dataset. To teach the network, you need to use both noisy and pure speech to test, so the learning data is built artificially, as for previous algorithms.

Noise is mixed at different levels to provide a wide range of signal-to-noise ratios, including clear speech and noise segments only. The algorithm does not use central average normalization, and data augmentation is used to make the network resistant to changes in frequency response. This is achieved by filtering noise and speech signals independently for each training example using second-order filters (formula 11).

$$H(z) = \frac{1 + r_1 z^{-1} + r_2 z^{-2}}{1 + r_3 z^{-1} + r_4 z^{-2}}, \tag{11}$$

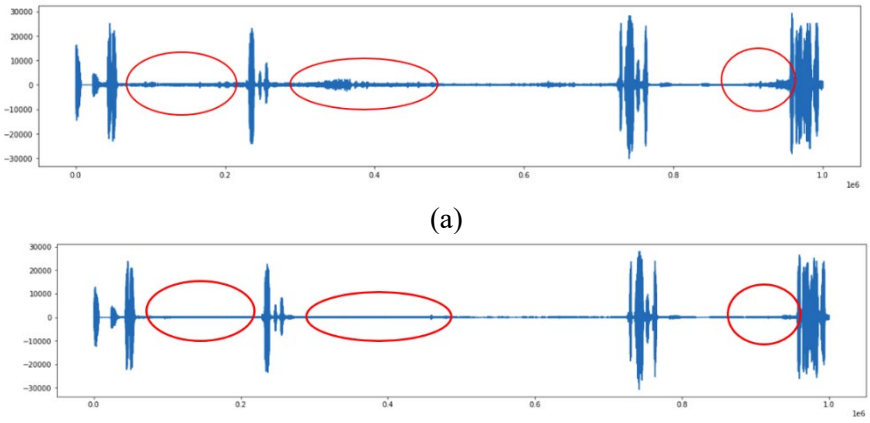
where $r_1, ..., r_4$ are random values, evenly distributed ranges from -3/8 to 3/8. Reliability to the amplitude of the signal is achieved by varying the final level of the mixed signal.

In total, there are 6 hours of speech and 4 hours of noise data, which we use to generate 140 hours of speech noise using various combinations of gains and filters and by oversampling the data to frequencies from 40 kHz to 54 kHz.

The RNNoise class consists of the following methods:

- read_wav (): Takes the name of the .wav audio recording, converts it to a supported format (16 bits, mono), and returns the pydub.AudioSegment object with the audio recording
- write_wav (): Accepts the name of the .wav audio recording, the pydub.AudioSegment object (or a byte string with audio data without wav headers) and saves the audio recording under the transmitted name
- filter (): Accepts the pydub.AudioSegment object (or byte string with audio data without wav headers) leads it to a sampling rate of 48000 Hz, splits the audio into frames (10 milliseconds long), clears them of noise, and returns the object pydub.AudioSegment (or byte string without wav headers) while preserving the original sampling rate
- filter_frame (): Clear only one frame (10 ms, 16 bits, mono, 48000 Hz) of noise (access directly to the binary file of the RNNoise library)

The input is an audio file that has some noise (Fig. 14 (a)), and the output is an audio file with reduced noise (Fig. 14 (b)).



(a)

(b)

Figure 14: Audio file diagram with noise (a) and after processing (b)

5. Results

The above algorithms were tested on 4 different audio, stationary and nonstationary, such as music or background conversation.

STOI and PESQ scores were used to evaluate the audio recordings obtained after noise removal. STOI is a metric for predicting the intelligibility of noisy speech, not the quality of speech (which is usually evaluated in silence). The main subjective tests of this method are tests of intelligibility (request for recognized words /symbols, etc.) [18].

PESQ is a family of standards that includes a testing methodology for automatically assessing the quality of speech experienced by a telephone system user. It was standardized as Recommendation ITU-T P [19].

The results are presented in table 1.

Table 1

Noise removal results

Method	Video	STOI	PESQ
Spectral subtraction	audio_statistical_noise.wav	0,8962362471828313	2.0765810012817383
	audio_offise_noise.wav	0.686388118838794	1.6020467281341553
	audio_street_noise.wav	0.6626346873426312	1.25617253780365
	audio_music_noise.wav	0.6668875301659041	1.273597002029419
CNN	audio_statistical_noise.wav	0.9658304757896437	3.4577016830444336
	audio_offise_noise.wav	0.8712570598849072	2.6612484455108643
	audio_street_noise.wav	0.8299030860960181	2.374866485595703
	audio_music_noise.wav	0.8298315520589445	2.6806342601776123
RNN	audio_statistical_noise.wav	0,9806221906909948	3.5431809425354004
	audio_offise_noise.wav	0.8428522793199281	3.0143574367834783
	audio_street_noise.wav	0.8323328598619955	3.0021986961364746
	audio_music_noise.wav	0.7372574604375085	1.5429587364196777

When working with stationary noises, each network showed high results. However, the worst noise elimination spectral subtraction showed on data with street noise because street noise has sudden declines or rises. Because of this, the results of spectral subtraction for these data were the worst. Recurrent neural network, the worst result shows the removal of music in the background and becomes noise, which is quite challenging to deal with.

We will deduce audio diagrams with noise and without for the addition of musical noise in Fig 15.

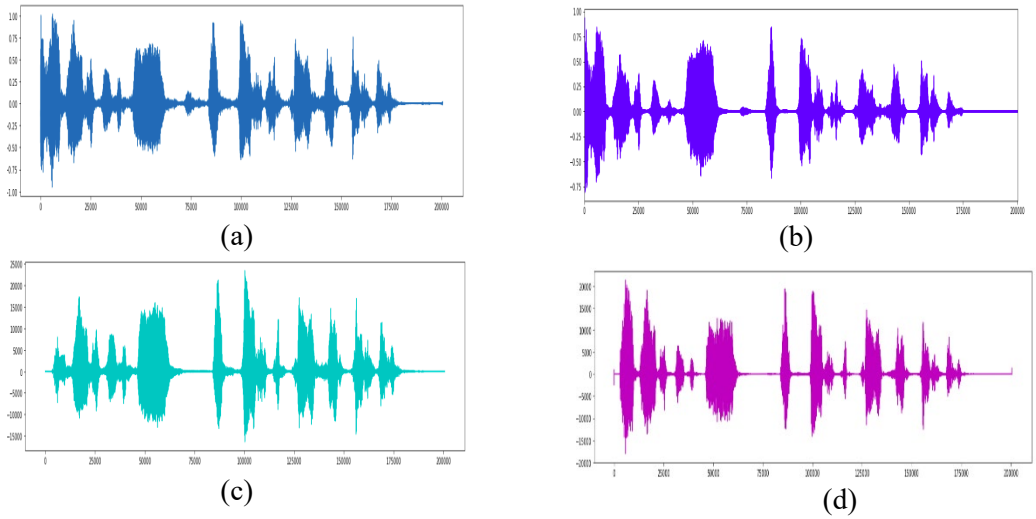


Figure 15: (a) - sound to add music noise, (b) – sound after processing by a spectral subtraction, (c) – sound after processing by a recurrent neural network , (d) – sound after processing by a convolutional neural network

The above is an example of graphs (Fig. 15), which display the sound before processing and after, using different methods. The convolutional network coped best with sound, reduced the amount of noise. It is also worth mentioning that one of the essential tasks of noise absorption is not to

degrade the sound itself. Because of this, spectral subtraction is the worst in use, because in addition to noise, it takes away the sound itself, which sometimes gives difficulties in recognizable languages. The advantages of this algorithm are simplicity and no need for training.

In this case, the CNN algorithm was better than RNN, but as shown in Table 1, RNN was better for noise removal such as street sound and stationary at 0.0024297737659774 0.0147917149013511, respectively. This method also showed promising results of PESQ evaluation. Diagrams of sound from the addition of street noise and sound charts after processing methods are presented in Fig. 16.

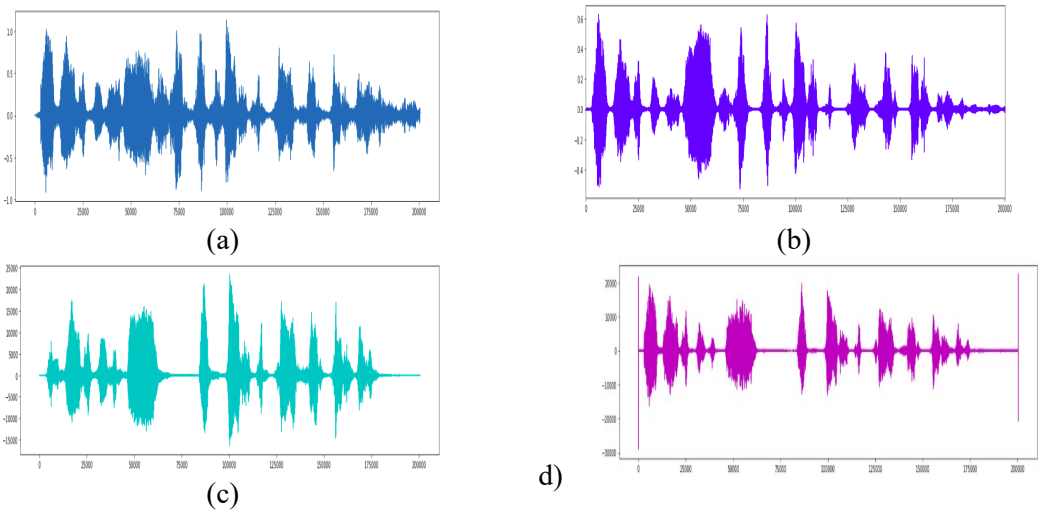


Figure 16: (a) - sound with street noise, (b) – sound after processing by a spectral subtraction, (c) – sound after processing by a recurrent neural network , (d) – sound after processing by a convolutional neural network

6. Discussion

The experiments section demonstrates various methods for removing noise from the audio file, such as spectral subtraction, recurrent neural networks, and convolutional neural networks. These methods were tested on different types of sound: stationary noise, background conversation sounds, street sounds, and background music noises. We will deduce results using the bar chart. To begin with, let's analyze the STOI estimate (Fig. 17).

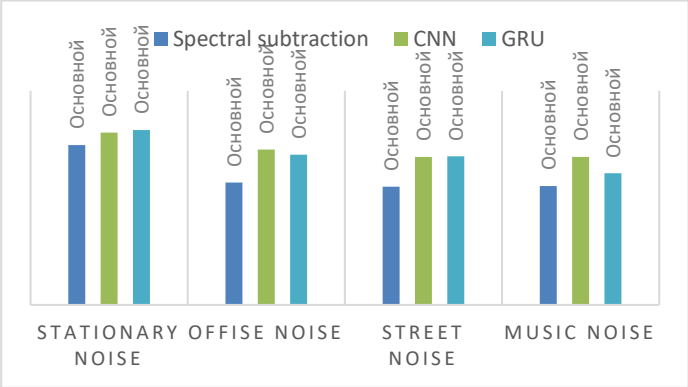


Figure 17: STOI estimates for different audio files using noise absorption using spectral subtraction, convolutional neural networks and recurrent neural networks

From Fig. 17 you can see that the worst result with noise removal was the method of spectral subtraction, which, unlike the other two, does not belong to the algorithms of artificial intelligence. The best results were demonstrated for stationary sound, as the primary purpose of this method was to remove this noise. But the results presented were still worse than CNN at 0.0695942286068124 and from RNN (GRU) at 0.0843859435081635.

So, suppose you compare simple algorithms and methods using artificial intelligence. In that case, the latter is preferred, as they can adapt to different noises. The quality of speech in the audio file suffers much less, which gives a higher STOI score because this assessment is based on language intelligibility. And since spectral subtraction is more damaging to speech, which is the leading indicator of deterioration. Because of this, for data preprocessing, for further speech recognition, this algorithm will work worse and degrade the final data.

The comparison of AI methods showed that each copes with this task, but there is no exact winner. This probably reflects the fact that both signal processing methods reflect a trade-off between different factors. RNN processing reduced stationary and street noise, but CNN processing performed better in removing noise such as background conversations and music noise according to the STOI score. But it should note that the difference between the estimates is not significant, which can be seen in Fig.18.

Let's analyze the indicators of the following assessment, which is based on the quality of speech. To begin with, we will deduce the diagram.

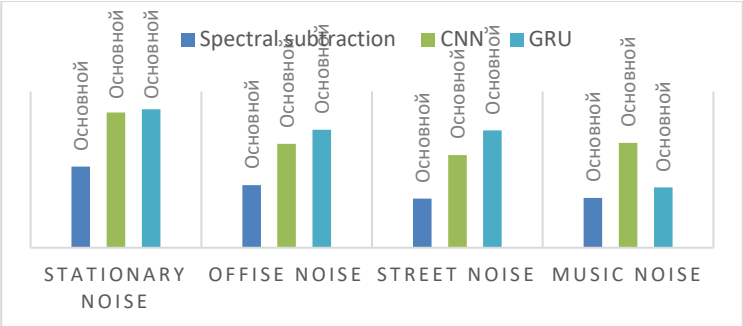


Figure 18: PESQ estimates for different audio files using noise absorption using spectral subtraction, convolutional neural networks and recurrent neural networks

From this diagram, we can see that the best, as for the preliminary assessment, showed when removing stationary noise. Since this assessment is based on speech quality, it is not surprising that spectral analysis showed such low results.

The RNN removal method showed the best results in all cases, except for musical noise, which indicates that this algorithm does not severely damage the audio file when removing noise. The audio file itself remains of good quality.

Therefore, algorithms using artificial intelligence are more advantageous, as they can adapt to sound and less damage to the data itself.

7. Conclusion

Sound noise is a problem that is a classic and began a long time ago but has not yet been fully resolved to this day. It can damage audio files, which can lead to a risk of impaired audio recognition, making it difficult to recognize speech. Most AI technologies are now used to solve this problem, and this paper demonstrates the results that have shown the benefits of these technologies. This technology has outstripped spectral analysis, both in noise removal and in preserved speech intelligibility, which are the main tasks for this topic.

The studies were performed using methods such as CNN and RNN. There was no exact winner in these studies. Although CNN is more commonly used in image processing, but also with problems related to noise in audio files, the algorithm has proven itself on the excellent side. RNN is not far behind in this matter. Each method performed better on different noises. RNN outperformed CNN in removing noise such as stationary and street noise, while CNN voiced background and music. RNN also showed high results in sound quality, in contrast to CNN, which offers some advantages in using this algorithm.

References

[1] L.Junfeng, A.Masato, S.Yoiti, A Two-Microphone Noise Reduction Method in Highly Nonstationary Multiple-Noise-Source Environments, IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences, 2010, E91A. 10.1093/ietfec/e91-a.6.1337.

[2] W.Edmonson J.Tucker, Digital Signal Processing System for Active Noise Reduction, Vol. 1, 2002, p. 49

[3] S.Boll, Suppression of acoustic noise in speech using spectral subtraction, in: IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 27, no. 2, April 1979, pp. 113-120. doi: 10.1109/TASSP.1979.1163209

[4] Y.Xu, J.Du, L.Dai, C.Lee, A Regression Approach to Speech Enhancement Based on Deep Neural Networks, in: IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 23, no. 1, Jan. 2015, pp. 7-19. doi: 10.1109/TASLP.2014.2364452.

[5] J.Valin, A Hybrid DSP/Deep Learning Approach to Real-Time Full-Band Speech Enhancement, in: IEEE 20th International Workshop on Multimedia Signal Processing (MMSP), 2018, pp. 1-5, doi: 10.1109/MMSP.2018.8547084

[6] T.Santos Silva, Practical Deep Learning Audio Denoising, 2019. https://sthalles.github.io/practical-deep-learning-audio-denoising/

[7] M.Michelashvili, L.Wolf, Speech Denoising by Accumulating Per-Frequency Modeling Fluctuations, 2019.

[8] X.Ruilin, W.Rundi, I.Yuko, V.Carl, Zh.Changxi, Listening to Sounds of Silence for Speech Denoising, 2020
[9] M.Keshavarzi, Use of a Deep Recurrent Neural Network to Reduce Wind Noise: Effects on Judged Speech Intelligibility and Sound Quality, Trends in Hearing, 2018, doi:10.1177/2331216518770964
[10] R.N.Kvetny, I.V.Bogach, O.R.Boyko, O.Y.Sofina, O.M.Shushura, Computer Simulation of Systems and Processes, Chapter 7: Digital Signal Processing
[11] D.Reay, Fast Fourier Transform, 2015. 10.1002/9781119078227.ch5
[12] S.R.Park, J.Lee, A Fully Convolutional Neural Network for Speech Enhancement, INTERSPEECH, 2017
[13] A.Omaima, Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks, International Journal of Information Technology and Computer Science, Vol. 7, 2015, pp. 8-18. 10.5815/ijitcs.2015.07.02
[14] J.Ma, Real-Time RNN Speech Noise Suppression on a MCU, STM32, 2020
[15] Real-Time RNN Speech Noise Suppression on a MCU (STM32). https://medium.com/analytics-vidhya/real-time-rnn-speech-noise -suppression-on-amicrocontroller-stm32-e17d8c3eac57
[16] M.Baranov, Recurrent Neural Active Noise Cancellation, 2019
[17] Recurrent Neural Active Noise Cancellation. https://towardsdatascience.com/deep-active-noise-cancellation-e364ce4562d4
[18] G.Vajente, Y.Huang, M.Isi, J.Driggers, J.Kissel, M.Szczepanczyk, Machine-learning nonstationary noise out of gravitational wave detectors, 2019
[19] S.Park, W.Kyung, Comparison of Neural Networks and Least Mean Squared Algorithms for Active Noise Canceling, 2018. All Theses. 2920. https://tigerprints.clemson.edu/all_theses/2920
[20] C.H.Taal, R.C.Hendriks, R.Heusdens, J.Jensen, A short-time objective intelligibility measure for time-frequency weighted noisy speech, in: IEEE International Conference on Acoustics, Speech and Signal Processing, 2010, pp. 4214-4217. doi: 10.1109/ICASSP.2010.5495701
[21] Perceptual Evaluation of Speech Quality https://en.wikipedia.org/wiki/Perceptual_Evaluation_of_Speech_Quality
[22] T.Sainburg, Noise reduction using spectral gating in python. https://timsainburg.com/noise-reduction-python.html
[23] C.Sun, M.Zhang, R.Wu, A convolutional recurrent neural network with attention framework for speech separation in monaural recordings, Vol. 11, 2021, p. 1434. https://doi.org/10.1038/s41598-020-80713-3
[24] N. Boiko, The issue of access sharing to data when building enterprise information model, in: IX International Scientific and Technical conference, Computer science and information technologies (CSIT 2014), Lviv, Ukraine, 2014, pp. 23-24.
[25] N.Boyko, R.Hlynka, Application of Machine Algorithms for Classification and Formation of the Optimal Plan, in: Proceedings of the 5th International Conference on Computational Linguistics and Intelligent Systems (COLINS 2021), Vol. 1: Main Conference Lviv, Ukraine, April 22-23, 2021, pp. 1853-1865