DOI: 10.2478/ijasitels-2023-0006

An Overview on Sound Features in Time and Frequency Domain

Constantin CONSTANTINESCU[1], *Remus BRAD*[1]

[1]*Computer Science and Electrical and Electronics Engineering Department, Faculty of Engineering, "Lucian Blaga" University of Sibiu, Romania*

{ctin.constantinescu, remus.brad}@ulbsibiu.ro

Abstract

Sound is the result of mechanical vibrations that set air molecules in motion, causing variations in air pressure that propagate as pressure waves. Represented as waveforms, these visual snapshots of sound reveal some of its characteristics. While waveform analysis offers limited insights, audio features provide a quantitative and structured way to describe sound, enabling data-driven analysis and interpretation. Different audio features capture various aspects of sound, facilitating a comprehensive understanding of the audio data. By leveraging audio features, machine learning models can be trained to recognize patterns, classify sounds, or make predictions, enabling the development of intelligent audio systems. Time-domain features, e.g., amplitude envelope, capture events from raw audio waveforms. Frequency domain features, like band energy ratio and spectral centroid, focus on frequency components, providing distinct information. In this paper, we will describe three time-domain and three frequency-domain features that we consider crucial and widely used. We will illustrate the suitability of each feature for specific tasks and draw general conclusions regarding the significance of sound features in the context of machine learning.

Keywords: sound, feature extraction, machine learning

1 Introduction

A sound is the result of mechanical vibrations originating from an object. When an object vibrates, it imparts kinetic energy to the surrounding air molecules. These molecules, influenced by the vibrating object, undergo oscillations, moving back and forth. As a result, localized variations in air pressure occur, manifesting as compressions and rarefactions, which propagate through the air as a pressure wave. A sound wave is a type of mechanical wave, representing the movement or vibration that travels through the air due to objects vibrating. This vibration sets the air molecules in motion, causing changes in air pressure that ripple through the air, carrying energy from one point to another. In simple terms, a sound wave is the means by which energy from vibrating objects is transported through the air, perceived by us as sound.

The most common representation of sound is the waveform. A waveform is a visual representation of sound, showing how air pressure changes over time. Think of it as a line graph where time is on the horizontal axis, and the vertical axis represents air pressure. When a sound occurs, this line wiggles up and down, corresponding to variations in air pressure caused by the sound. The shape of the line, or waveform,

provides information about the sound's characteristics. For example, if the line on the graph repeats in a regular and smooth pattern, it indicates a periodic sound, like a musical note. On the other hand, if the line is more jagged and irregular, it represents an aperiodic sound, such as noise. The waveform serves as visual snapshot of the sound, allowing us to study and understand its properties, like frequency, amplitude, and duration. It is essentially a fingerprint for sounds, aiding in identification and analysis.

Amplitude quantifies how strong or loud the sound or signal is. In a waveform representation, it's often observed in the height of the peaks or the depth of the troughs from the central axis. Larger amplitude signifies a stronger, louder signal, while a smaller amplitude represents a weaker, quieter signal.
An audio signal is a representation of sound that encodes all the information necessary to faithfully reproduce the sound it represents. Analog signals have continuous values for both time (on the x-axis) and amplitude (on the y-axis), while digital signals represent data in the form of a sequence of discrete values. These discrete values are finite in number and are used to encode information. In the context of digital signals, both the timing (time) and the amplitude of the signal are quantized into discrete, specific values.

By looking at the sound waveform we can only have a basic idea about the sound with very limited information. Audio features provide a quantitative and structured way to describe sound, enabling data-driven analysis and interpretation. Different audio features capture various aspects of sound, such as pitch, timbre, rhythm, and intensity, facilitating a comprehensive understanding of the audio data. By leveraging audio features, machine learning models can be trained to recognize patterns, classify sounds, or make predictions, enabling the development of intelligent audio systems for applications like speech recognition, music genre classification, and sound analysis.

Audio features are categorized by the domain they operate in, such as time-domain, frequency domain, or specialized domains like the cepstral domain. This is the most important strategy that we have for categorizing different audio features.

There are certain audio features that are in the time-domain. Some of these are amplitude envelope, root-mean square energy and zero crossing rate. They are extracted from a waveform, from the basic raw audio, capturing events over time.

The sound is also characterized by frequency. The frequency is an extremely important descriptor of sound. So, there are other features that go under the name of frequency domain features, which focus on the frequency components of sound. Some of these are band energy ratio, spectral centroid and spectral flux, frequency domain features providing information not readily available in the time-domain.

2 Time domain features

2.1 Understanding time domain features

The core concept in the time domain revolves around waveform analysis, where the waveform visually illustrates the changing amplitude of a signal as it progresses through

time. In a waveform, the x-axis represents time, and the y-axis represents amplitude. Each point on the waveform corresponds to a specific moment in time, providing a direct visualization of the signal's behaviour. This allows us to observe all the events that occurred in a sound over time. Features that extract information from this representation are called time-domain audio features.
For the extraction pipeline, we will consider an analog sound, such as the sound of a violin or ambient noise. First, we want to convert that sound using the ADC (Analog digital conversion) process. This involves sampling and quantizing the analog sound to obtain a digital signal. Once we have the digitalized version of that sound, the next step will be the framing. Framing means that we want to bundle together a bunch of samples. Frames in audio processing serve as fundamental units of analysis, breaking down continuous audio signals into manageable segments. These segments are essential for various tasks, including audio analysis, feature extraction, and signal processing. In other words, frames represent chunks of audio that are perceptible to the human ear. These chunks are chosen to be of a duration that makes sense in the context of the analysis, such as a fraction of a second.
If we take a look at one sample at the sampling rate of 44.1 KHz (sample rate for the CD ROM), we find out that that sample has a duration (inverse of the sample rate) of 0.0227ms. The duration of this single sample is below the threshold of the time resolution of the human hearing (around 10ms). All the things that are below 10ms, the human ear cannot appreciate them as acoustic events. So with frames, we will have enough duration of an audio signal so that we can appreciate that.
To optimize computational efficiency, frame sizes often adhere to a power of 2 for the number of audio samples within each frame. The size of frames can vary depending on the specific application and analysis requirements. Common frame sizes typically range from 256 to 8192 samples. Smaller frame sizes, such as 256 samples, are suitable for capturing rapid changes in audio, while larger frame sizes, such as 8192 samples, are more appropriate for analyzing longer-term audio characteristics.

2.2 Amplitude envelope

The amplitude envelope of a sound frame is defined as the maximum amplitude value among all the individual samples within that frame. Figure 1 [1] illustrates a segment of a sound represented as a waveform. The waveform is divided in 6 frames while the amplitude envelope is marked with the green small rectangles. Only those values are used in this case to represent this sound. It is worth mentioning that, for clarity, in this case the waveform is divided into this six frames. In reality, frames are much smaller so we would have a lot more values for the amplitude envelope.

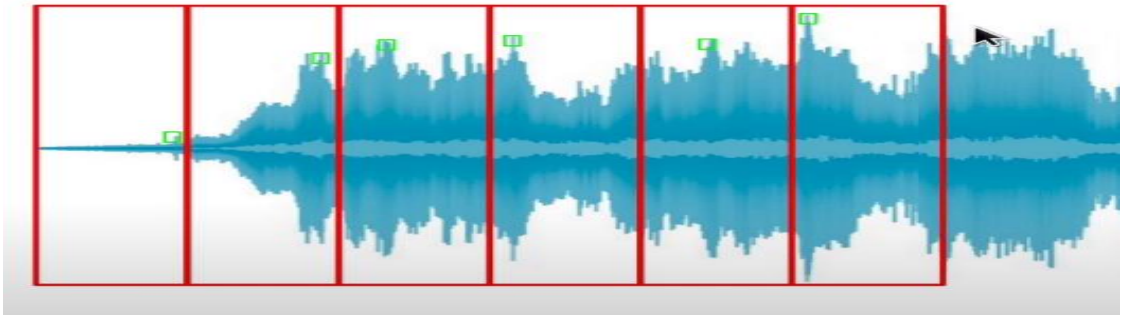

Figure 1. Visual representation of the amplitude envelope of a sound [1]

The formula for calculating the amplitude envelope for one single frame looks like this:

$$\mathrm{AE_t} = max_{k=t\cdot K}^{(t+1)\cdot K-1} s(k) \quad (1)$$

Where:

- t is the frame
- AE_t represents the amplitude envelope at frame t
- K is the frame size (number of samples we have in a frame)
- $k=t*K$ is the first sample of frame t
- $(t+1)*K-1$ is the last sample of the frame t
- $s(k)$ represents the amplitude of the k-th sample

As expected, the formula (1) is a maximum function to determine the highest value within the range of samples that belong to that frame.
The biggest problem of this feature is that it tends to be influenced by extreme values or outliers, because we only consider one value (the maximum), one sample from every frame that might not represent the sound correctly. Imagine a frame where all the values are close to zero, with only one spike at a considerably higher value. The amplitude envelope would only consider that spike, while all other small values from that frame would not have any influence on it. In this case, the value of that spike is not representative for the entire frame.

Applications of AE

The amplitude envelope provides a general sense of loudness, since the amplitude is directly related to intensity of a sound. Its primary application lies in onset detection, where it is utilized to identify the moment a particular note or acoustic event begins, by capturing sudden changes in the signal [2]. Figure 2 demonstrates the efficacy of the amplitude envelope in event detection, showcasing the waveform of a recording featuring a functioning valve in an industrial setting. In this instance, the amplitude envelope (red) serves to identify the moment when the valve is opened or closed, clearly discernible through the spikes in the envelope.

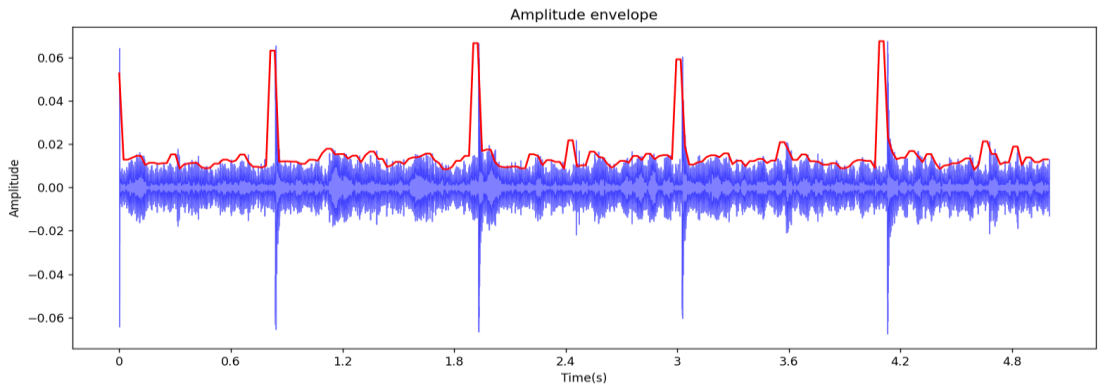

Figure 2. Visual representation of the sound of a working valve and its AE

A big part of the interpretation of sound events rests on duration, a property that provides important information about the materials and actions involved in it. Using amplitude envelope we can estimate the duration of the event and also determine if it had a natural ending or an abrupt one [3]. The difference between percussive and flat envelopes has been extensively researched as they occur not only in duration estimation, but also in tasks like audio-visual integration [4] or associative memory [5]. Percussive sounds, usually heard when two objects collide, carry a lot of information about the event. Flat sounds in contrast are characterized by a period of sustained amplitude with abrupt onsets and offsets. This indicates, that flat sounds usually do not come from nature, but are rather synthetically generated (for example alarms) [6].
Apart from onset/offset detection this feature can be used in a lot of other tasks as well, like for example music genre classification.

2.3 Root-mean square energy

The Root-mean-square energy is the root mean square (RMS) of all samples in a frame. In other words, root-mean-square energy is calculated as the square root of the mean of the squared values of all samples within a frame.
The formula (2) for calculating the root-mean-square energy for one single frame:

$$\mathrm{RMS_t} = \sqrt{\frac{1}{K} \cdot \sum_{K=t\cdot K}^{(t+1)\cdot K-1} s(k)^2} \tag{2}$$

Where:

- *s(k)* is the amplitude at sample *k* (the square of the amplitude is the energy)
- We sum the energy for all the samples in frame *t*
- We take the mean of the sum of the energy

So, the root-mean-square energy is the square root of the mean of the sum of energy.
Root-mean-square (RMS) energy is a measure commonly used as an indicator of loudness in audio processing. Unlike the Amplitude Envelope (AE), RMS energy is less sensitive to extreme values or outliers in the signal. It calculates the square root of the mean of the squared values of all samples within a frame, as opposed to the AE where only one value was taken into consideration. This is why RMS is providing a more stable representation of the signal's energy.

Application of the RMSE

RMS energy is particularly useful in tasks such as audio segmentation, where identifying different sections or events in an audio signal is crucial. For example, when we want to decide whether someone is talking or not, and we have that change in the sound signal and want to segment who is talking. In figure 3, we can observe how the RMS tends to zero when there is a pause in speech.

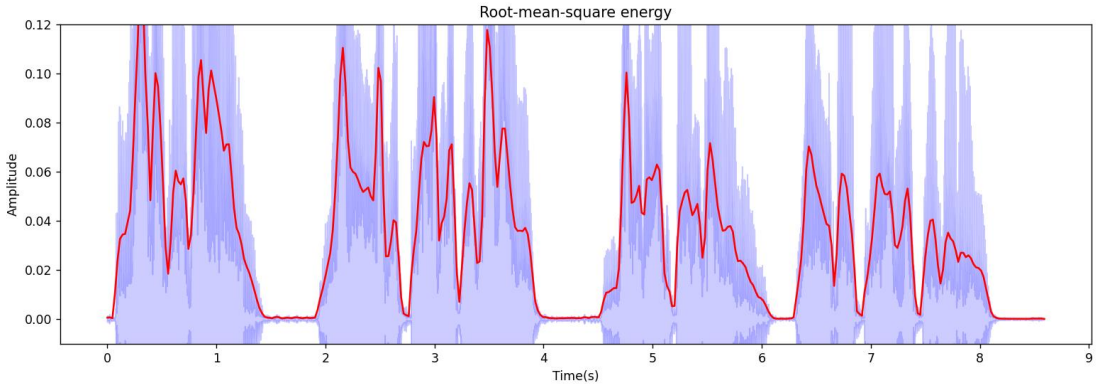

Figure 3. Visual representation of the RMSE (red) of a speech recording

Specifically, RMS of sound has been used in the underwater detection of spiny lobster [7] to establish sound pressure levels and generally in research about underwater animals in relation to sound. In medicine, RMS was also used in combination with other features in automatic diagnosis of COVID-19 from respiratory sound data [8].
Additionally, it plays a role in music genre classification, helping to characterize and distinguish the overall loudness characteristics of different music genres.

2.4 Zero crossing rate

The zero crossing rate provides information about the number of times a signal crosses the horizontal axis.
In Figure 5 [1] we have the visualization of a signal wave of a frame. The zero crossing rate is equal to the count of the green dots. For each of these green dots, we have a crossing of the horizontal axis. For this signal, the zero crossing rate is equal to 6.

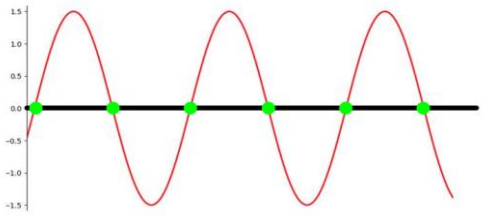

Figure 4. Visual representation of the zero crossing [1]

How do we calculate the zero crossing rate mathematically? The formula (3) for calculating the zero crossing rate:

$$\mathrm{ZCR_t}=\frac{1}{2}\sum_{k=t\cdot K}^{(t+1)\cdot K-1}\left|sgn\big(s(k)\big)-sgn\big(s(k+1)\big)\right| \tag{3}$$

The basic idea here is that we compare the amplitude value for consecutive pairs of samples and we look whether there are differences in the signs of those amplitudes for those consecutive samples.
For this we are going to use the sign function. The sign function *sgn(s(k))* gives back the sign of a given value. In this case, if the amplitude *s(k)* is greater than zero, the sign function gives us back the value "+1" because the amplitude is positive. If the amplitude is negative, we get back the value "-1" and if the amplitude is equal to zero, then we get back the value "0".

But how do we calculate the zero crossing rate for each pair of samples? We take the sign for the amplitude at sample k and then we subtract the sign for the amplitude at sample *k+1*. Basically we are comparing the two consecutive samples.
If both amplitude values have the same sign, we get 0 on that side, so we don't get any value that is going to increase the zero crossing rate.
But if we have alternate opposite signs, for example at sample k we have a negative amplitude and at sample *k+1* we have a positive amplitude, we will get a value of 2.
In this formula (3), we sum over all the samples that we have in a frame so that we are going to get the values for all the zero crossings that we have in a frame.

Applications of the ZCR

Zero Crossing Rate (ZCR) holds significance in diverse applications. It aids in distinguishing between percussive and pitched sounds, contributing to the recognition of distinct auditory characteristics. This is because percussive sounds usually tend to have a quite random zero crossing rate, so they tend to change the zero crossing rate a lot, whereas pitch sounds tend to be way more stable in zero crossing rates.
In the field of audio signal processing, ZCR is employed for monophonic pitch estimation, offering insights into the fundamental frequency of a single musical note or sound. There is a relationship between the number of zero crossings and the pitch. And if we have a monophonic pitch, we can observe that the higher the number of zero crossings that we have, the higher the pitch is going to be.
In the context of speech recognition, the zero crossing rate can be utilized to distinguish between signals containing voice and those that are unvoiced [9]. Typically, voice signals exhibit a lower zero crossing rate compared to unvoiced signals. This distinction may be attributed to the fact that unvoiced segments tend to be noisier, resulting in a higher zero crossing rate [10].

3 Frequency domain features

3.1 Understanding frequency domain features

The initial steps for obtaining a frequency domain representation of the sound mirror those found in the time-domain feature pipeline. We begin with an analogue sound, apply sampling, followed by quantization to obtain the digitized version of the sound (audio signal). Next, we frame the signal, resulting in a series of frames. At this stage, we have a waveform divided into multiple frames. However, how do we transition from this time-domain representation to the frequency domain? The solution lies in applying the Fourier transform, which translates the signal from the time-domain to the frequency domain. Unfortunately, there is a problem we encounter when we do that, and this is the spectral leakage.
In signal processing, spectral leakage arises when processing a signal that lacks an integer number of periods. In real life, most sounds are not periodic so the endpoints of the frames are discontinuous. This leads to additional high-frequency components in the signal's spectrum, introduced by the abrupt changes or discontinuities. Windowing, a technique in signal processing, addresses this issue by applying a specific windowing function to each frame of an audio signal. The Hann window, a widely used windowing

function, is a bell-shaped curve that tapers the signal, reducing spectral leakage by smoothing transitions at the frame edges.
The application of the Hann window involves multiplying it by the original signal at each corresponding sample, effectively eliminating discontinuities. However, we encounter some signal loss when gluing multiple windowed signals together.
To address this, overlapping frames are introduced. In non-overlapping frames, a frame size is applied to the signal without any overlap, so the current frame just follows the previous one as shown in figure 1 [1]. Overlapping frames, on the other hand, involve applying a frame size to the signal with an overlap. The hop length, representing the number of samples shifted to the right for each new frame, allows for minimizing spectral leakage in the Fourier Transform of the windowed signal.
Following this transformation, we acquire a spectrum with minimized spectral leakage, where the x-axis represents frequency and the y-axis represents magnitude.
However, this representation now lacks information about time. To capture both time and frequency information, we need to apply the Short-Time Fourier Transform (STFT) on the waveform, which produces a spectrogram—a representation of sound that incorporates details about both time and frequency.

3.2 Band energy ratio (BER)

This feature provides us information about the relation between the energy in the lower and higher frequency bands. We can consider it as a measure of how dominant the lower frequencies are. To understand how it works, we need to understand the math behind it. The formula (4) for the band energy ratio is as follows:

$$\mathrm{BER_t} = \frac{\sum_{n=1}^{F-1} m_t(n)^2}{\sum_{n=F}^{N} m_t(n)^2} \tag{4}$$

We can observe that the formula is a fraction, which is expected since we are dealing with a ratio of two elements. In both the numerator and denominator, we have the power of the signal at time *t* and frequency bin *n*. The power is essentially the squared magnitude of the signal. The capital *F* in the sum function represents the split frequency, serving as a threshold that distinguishes higher frequencies from lower frequencies. The choice of the split frequency is arbitrary and depends on the application, with a common value being 2000 Hz.
So, in the upper part of this fraction we have the power in the lower frequency bands at a specific point in time, whereas at the denominator we have the opposite of that, meaning the power in the higher frequency bands. It is worth mentioning that the result will be the band energy ratio at a specific frame, so the formula has to be applied to each frame.
The band energy ratio can be used in all sorts of things in music and speech processing, but specifically it has been extensively used to discriminate music from speech, for certain music classification problems, like music genre classification or mood classification.

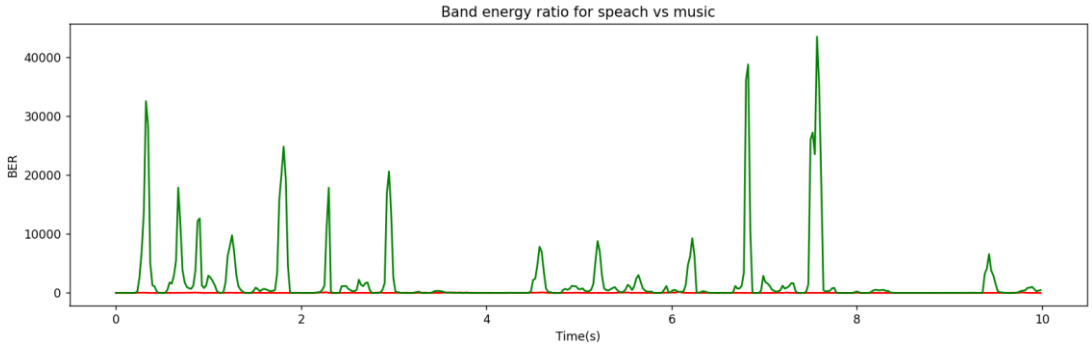

Figure 5. Visual representation of BER of a speech (green) versus music (red)

In Figure 5, we can observe that the band energy ratio (BER) for speech, indicated by the red line, exhibits significantly lower values compared to the BER for a musical piece, represented by the green line. This illustrates one of the most commonly applied uses of the band energy ratio in sound processing.
But music and voices are only 2 classes of sounds and the band energy ratio showed very good results in environmental sound recognition and classification experiments with more than ten classes [11] that include for example the inside of a running car or the sound of a forest. [12]

3.3 Spectral centroid (SC)

A very common and famous feature is the spectral centroid. It is going to provide us the center of gravity of the magnitude spectrum [13]. In other words, it will give us the frequency band where most of the energy is concentrated. It maps onto a very prominent timbral characteristic of the sound, being a measure of "brightness", meaning it will tell us how open or dull a certain sound is. The spectral centroid is essentially the weighted mean of the frequencies.

$$SC_t = \frac{\sum_{n=1}^{N} m_t(n) \cdot n}{\sum_{n=1}^{N} m_t(n)} \tag{5}$$

The weights are represented by the magnitude at a specific frequency bin, whereas the rest of the formula(5) is a basic mean. Like the band energy ratio, we calculate the spectral centroid for each frame.
Figure 6 highlights the limitations of information obtained solely from a waveform, underscoring the increased utility of frequency domain features, such as the spectral centroid. Specifically, we examine a recorded lung sound captured using an electronic stethoscope. The spectral centroid provides a clearer view of the patient's respiratory cycles, a distinction that is challenging or nearly impossible to discern in the waveform alone. This insight proves valuable, particularly for tasks like segmentation. Instead of blindly partitioning the sound into arbitrary sections, leveraging the spectral centroid allows for precise division into complete respiratory cycles. This approach ensures that essential information about the patient's respiration is preserved, enhancing the accuracy and relevance of the analysis.

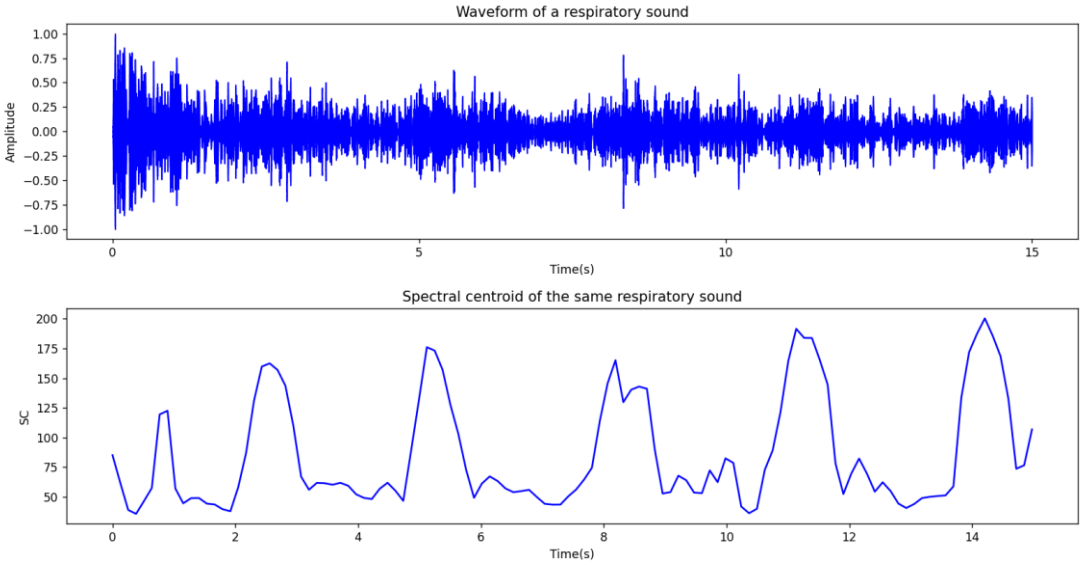

Figure 6. Waveform representation and spectral centroid of a lung sound recording

During the pandemic, it was used in the exploring of automatic diagnosis of COVID-19 from sound data [8] but also generally in medicine to process sounds like cough [14], voice and breathing [15]. The spectral centroid is also very much used in audio classification or music classification problems, usually with classic machine learning techniques. It is one of the key frequency domain audio features.

3.4 Bandwidth (BW)

The bandwidth is somewhat related to the spectral centroid meaning that we can think of the bandwidth as that spectral range, which is of interest and is around the centroid. In simpler terms, the bandwidth represents the variance from the spectral centroid and has a correlation with the perceived timbre. Similar to the spectral centroid, it is a weighted mean. However, in this case, it is not a weighted mean of the frequencies but rather a weighted mean of the distances of frequency bands from the spectral centroid.

$$BW_t = \frac{\sum_{n=1}^{N} |n - SC_t| \cdot m_t(n)}{\sum_{n=1}^{N} m_t(n)} \quad (6)$$

The weights are once again the magnitude for the signal at the specific time frame *t* and the specific frequency band *n*, while the distance between the frequency band and the spectral centroid is represented by the absolute value in the formula(6). Consequently, the bandwidth varies based on how energy is distributed across the different frequency bands. If the energy is dispersed across the frequency bands, the bandwidth value decreases. On the opposite, if the energy is concentrated in only a few frequency bands, the bandwidth value decreases. It is evident that it measures how much the energy is spread, and for this reason, the bandwidth is also referred to as spectral spread.
Bandwidth has been extensively employed in music processing, including applications like music genre classification or music mood classification.

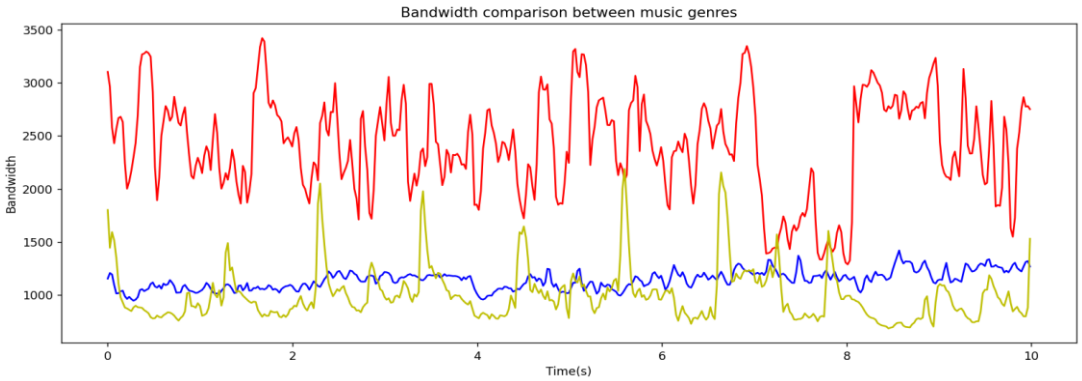

Figure 7. Visual representation of the BW for rock music(red), classical music(blue), jazz music(yellow)

Figure 7 illustrates the bandwidth for 3 music genres: rock, classical and jazz. We can observe that the bandwidth is much higher for rock music than the other genres, proving its utility in music genre classification.
It has also applications in medicine, for example in pre-processing voice sounds, to prepare them for anomaly detection like intoxication [16] or various medical conditions.

4 Experimental results

In order to demonstrate some of the theoretical notions explained and some of the conclusions we drew in this paper, we conducted an experiment. Given that most features find applications in sound classification, we utilized each of these features to classify industrial machine sounds. The sounds were sourced from the MIMII Dataset [17], which contains labeled sounds from four types of machines: fans, pumps, valves and sliders. While the dataset was primarily constructed for anomaly detection, our focus involved approximately 1000 normal sounds (without anomalies) from each machine type, at a signal-to-noise ratio of 6 dB. Each sound was saved as a WAV file and lasted for 10 seconds. In terms of pre-processing, we resampled each sound to 16 kHz and divided it into frames of 1024 samples with an overlap of 512 samples.
With the data prepared, we extracted every feature presented in this paper. As all features were calculated for each frame, the result was six arrays (one for each feature), each containing another 4000 arrays, one for each particular sound. Since all the sound files were 10 seconds long, each array had the same length.
For the classification, we chose a supervised approach, using the classic machine learning algorithm KNN [18] with a *k*-value of 3 neighbours [19]. The data was split into training and testing samples, with the test samples constituting 30% of the entire dataset. We evaluated the algorithm using three metrics as shown in Table 1 with their corresponding results.
In this particular case, the algorithm was able to better distinguish between fan, pump, slider and valve when using the time domain features, with the RMS standing out. This can be explained by the fact that these industrial machines all operated in low frequency but varied in amplitude. We ran the algorithm multiple times with various *k* values, resulting in slightly different outcomes, but the proportion between features remained consistent.

	Accuracy (%)	Precision (%)	F1-Score (%)
AE	79,07	87,30	74,55
RMSE	96,40	96,46	96,46
ZCR	70,89	75,22	63,29
BER	50,61	56,63	41,96
SC	70,32	79,19	70,02
BW	59,04	52,90	52,32

Table 1. KNN Algorithm evaluation results

5 Conclusions

Classic machine learning algorithms, such as decision trees, support vector machines, and k-nearest neighbors, heavily rely on sound features for tasks like audio classification and genre recognition. In traditional machine learning, feature extraction is a critical process, with audio features encompassing Amplitude Envelope, Root-Mean-Square Energy, Zero Crossing Rate, Band Energy Ratio, Spectral Centroid, Spectral Flux, Spectral Spread, Spectral Roll-Off, and more. The selection of features depends on their suitability for the specific problem at hand.

For sound classification, one might choose features like amplitude envelope, zero-crossing rate, and spectral flux. These selected features are isolated, extracted from audio files, and fed into traditional machine learning algorithms like support vector machines for training.

Deep learning techniques, including neural networks like CNNs and RNNs, are increasingly employed for complex tasks such as speech recognition, music generation, and audio synthesis. In deep learning, unstructured audio representations, such as raw audio or spectrogram-like features, can be passed to systems that leverage neural networks to automatically extract relevant features from the audio data. While deep learning eliminates the mandatory feature extraction step, audio features can still be valuable in the preprocessing phase, so the utility of these features should not be overlooked.

In the context of sound anomaly detection, we do not always have the necessary amount of data for the deep learning approach, especially since anomalous sounds occur rarely and unexpectedly, which makes them hard to record. For this reason, we may have to employ classic ML algorithms to perform the necessary tasks. Depending on the task, extracting the appropriate feature or combining some of the features presented in this paper will be necessary and will improve the results. Furthermore, regardless of the approach, the raw sound has to go through some preprocessing steps, like segmentation, smoothing or filtering, steps where these features prove their utility.

Sound is part of every aspect and moments of our lives so the need to process it automatically is undeniable. Regardless if we need to detect anomalies in sound of an industrial machine or in the respiratory auscultation of a patient, these audio features are valuable tools to achieve our goals.

References

[1] V. Velardo, "https://github.com/musikalkemist/AudioSignalProcessingForML," 10 10 2020. [Online]. Available: https://github.com/musikalkemist/AudioSignalProcessingForML. [Accessed 27 11 2023].

[2] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies and M. B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing,* pp. 1035-1047, 2005.

[3] G. T. Vallet, D. I. Shore and M. Schutz, "Exploring the role of the amplitude envelope in duration estimation," *Perception,* vol. 43, no. 7, pp. 616-630, 2014.

[4] . L. Chuen and M. Schutz, "The unity assumption facilitates cross-modal binding of musical, non-speech stimuli: The role of spectral and amplitude envelope cues," *Attention, Perception, and Psychophysics,* pp. 1512-1528, 2016.

[5] M. Schutz , J. Stefanucci, S. Baum and A. Roth, "Name that percussive tune: Associative memory and amplitude envelope," *Quarterly Journal of Experimental Psychology,* pp. 1323-1343, 2017.

[6] S. Sreetharan, J. Schlesinger and M. Schutz, "Decaying amplitude envelopes reduce alarm annoyance: Exploring new approaches to improving auditory interfaces," *Applied Ergonomics,* 2021.

[7] Y. Jézéquel, L. Chauvaud and J. Bonnel, "Spiny lobster sounds can be detectable over kilometres underwater," *Sci Rep 10,* 2020.

[8] C. Brown, J. Chauhan, A. Grammenos, J. Han, A. Hasthanasombat, D. Spathis, T. Xia, P. Cicuta and C. Mascolo, "Exploring Automatic Diagnosis of COVID-19 from Crowdsourced Respiratory Sound Data," *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20),* pp. 3474-3484, 2020.

[9] G. Sharma, K. Umapathy and S. Krishnan, "Trends in audio signal feature extraction methods," *Applied Acoustics,* vol. 158, 2020.

[10] Y. A. Ibrahim, J. C. Odiketa and T. S. Ibiyemi, "Preprocessing technique in automatic speech recognition for human computer interaction: an overview.," *Ann Comput Sci Ser,* vol. 15, no. 1, pp. 186-191, 2017.

[11] S. Chu, S. Narayanan and C.-C. J. Kuo, "Environmental Sound Recognition With Time–Frequency Audio Features," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 17, no. 6, pp. 1142-1158, 2009.

[12] S. Sivasankaran and K. Prabhu, "Robust features for environmental sound classification," *IEEE International Conference on Electronics, Computing and Communication Technologies,* pp. 1-6, 2013.

[13] F. Alías, . J. C. Socoró and X. Sevillano, "A Review of Physical and Perceptual Feature Extraction Techniques for Speech, Music and Environmental Sounds," *Applied Sciences,* vol. 6, no. 5, 2016.

[14] R. Islam, E. Abdel-Raheem and M. Tarique, "A study of using cough sounds and deep neural networks for the early detection of Covid-19," *Biomedical Engineering Advances,* vol. 3, 2022.

[15] A. Hassan, I. Shahin and M. B. Alsabek, "COVID-19 Detection System using Recurrent Neural Networks," *International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI),* pp. 1-5, 2020.

[16] A. B S, S. R. Shetty, S. Srinivas, V. Mantri and V. R. B. Prasad, "Intoxication Detection using Audio," *2023 IEEE 8th International Conference for Convergence in Technology (I2CT),* pp. 1-6, 2023.

[17] H. Purohit, R. Tanabe, K. Ichige, T. Endo, Y. Nikaido, K. Suefusa and Y. Kawaguchi, "MIMII Dataset: Sound Dataset for Malfunctioning Industrial Machine Investigation and Inspection," in *Proc. 4th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2009.

[18] R. Cretulescu and D. Morariu , Tehnici de clasificare si clustering al documentelor, Cluj Napoca: Editura Albastra, 2012.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and É. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.