A Web Application Facilitating the Analysis of Scaling Techniques on Medical Time Series Clustering

Sindi Veliko[1], Mohan Xu[1] and Lena Wiese[1,2]

[1]Fraunhofer Institute for Toxicology and Experimental Medicine, Nikolai-Fuchs-Strasse 1, 30625 Hannover, Germany
[2]Goethe University Frankfurt, Robert-Mayer-Str. 10, 60325 Frankfurt am Main, Germany

Abstract

Integrating unsupervised machine learning algorithms for large-scale database and E-health time series analysis is still challenging nowadays. By utilizing the International Conference on Biomedical and Health Informatics (ICBHI) Respiratory Sound Database we illustrate the inherent signal processing challenges with a focus on scaling methods. We present a web application that facilitates deeper insights for experts and aids medical professionals in examining respiratory sounds with the aim to develop and validate a machine learning-based digital auscultation system to enhance patient care. The source code is available under https://github.com/sindiveliko/DigitaLung_WebApp.

Keywords

Health Informatics, Respiratory Sound, Time Series Analysis, Unsupervised Machine Learning, K-Means Clustering, E-Health

1. Introduction

Lung sound analysis has significant business impact across various sectors, particularly in healthcare, technology, and pharmaceuticals. Lung sound analysis can detect abnormalities such as wheezes, crackles, and other adventitious sounds, facilitating early diagnosis of conditions like asthma, chronic obstructive pulmonary disease (COPD), pneumonia, and other respiratory illnesses. In this context, automated AI-based analysis reduces human error, potentially leading to more accurate diagnoses and appropriate treatment. In the long run, wearable devices with lung sound analysis capabilities [1] enable remote diagnosis and monitoring, which will be especially beneficial for patients in rural areas. Moreover such devices allow for continuous monitoring of patients, leading to timely interventions and improved patient outcomes. Early detection and improved management of respiratory conditions can lead to reduced hospitalization rates and lower healthcare costs – whereas continuous lung sound monitoring can help in assessing patient adherence to prescribed medications and the effectiveness of treatments, leading to better management of chronic conditions. Developing sophisticated algorithms for lung sound analysis and presenting the outcome in an accessible user interface could hence be a significant business opportunity, with applications in both health apps and professional medical tools.

LWDA'24: Lernen, Wissen, Daten, Analysen. September 23–25, 2024, Würzburg, Germany
sindi.veliko@item.fraunhofer.de (S. Veliko); mohan.xu@item.fraunhofer.de (M. Xu);
lena.wiese@item.fraunhofer.de (L. Wiese)
http://www.dbda.cs.uni-frankfurt.de/ (L. Wiese)
0000-0001-6940-1361 (M. Xu); 0000-0003-3515-9209 (L. Wiese)

While there has been a notable global reduction in age-standardized mortality and morbidity rates, the prevalence of chronic respiratory diseases (CRDs) has witnessed an approximate 40% increase since 1990 [2]. Understanding the characteristics of respiratory sounds is crucial for their effective processing and accurate classification. Respiratory sounds, when digitized as set of numerical values ordered by time, constitute medical time series data. As reported in [3], the frequency range of normal respiratory sounds typically spans from 50 Hz to 4 kHz. Time series analysis in medicine, crucial for accurately predicting disease progression and undertaking large case analysis, involves extracting insights from these data points by understanding how variables change over time [4]. Hence to facilitate the analysis and enable visual exploration of respiratory sounds, we present the development and evaluation of a web-based system to enable the intelligent analysis of time series data.

In particular, in this article we analyze if and how unsupervised learning methods have potential for analyzing respiratory sound data. An analysis pipeline is established for this purpose, specifically for clustering. Scaling methods are evaluated as a preprocessing step. Cluster analysis can be difficult to interpret due to its unsupervised nature and the complexity of the data. This is particularly true for the time series case where the results cannot be plotted intuitively on a flat two-dimensional graph to allow visual inspection of the partitioning. To evaluate such scenarios, we employ a range of evaluation metrics to offer valuable insights into the quality and properties of the clustering results. In terms of a practical application, specialized audio functionalities are incorporated including aspects such as data storage, preprocessing, data visualization, and unsupervised learning. This article begins with related work (Section 2) followed a theoretical exploration of respiratory sounds from a time-series standpoint (Section 3). Section 4 delves into application development and elaborates on various techniques proposed and implemented within the application. Section 5 continues by highlighting key outcomes for different combinations of scaling and clustering settings while Section 6 concludes with a summary and proposals for future work.

2. Related Work

The popularity of mobile and portable medical devices has boosted the creation of large time series databases [5]. Time series can be analyzed with a variety of methods surveyed in [6]. Time series analysis in medicine involves extracting insights from data points by understanding how variables change over time. This analysis involves predicting and accurately estimating disease progression, large case analysis, time-to-event analysis, and more [4]. For our purposes, a time series T with equally spaced time steps is mathematically expressed as:

$$T = (t_1, t_2, \dots, t_n) \text{ where } n \in \mathbb{N} \tag{1}$$

where t_i is the value at the i-th time point, and n the total number of samples. Our performance evaluations are based on the ICBHI 2017 Respiratory Sound Database [7] containing normal and adventitious (crackles / wheezes) lung sounds; for detailed information refer to Table 1.

Integrating advanced lung sound analysis technologies into stethoscopes, wearable devices, and diagnostic equipment can drive innovation and create high-demand products. Several studies consider the application of unsupervised machine learning algorithms to audio datasets

[8, 9]. Park et al. [10] present an ensemble support vector machine to improve the classification of lung sounds of pediatric patients. This model aims to identify and categorize various lung sounds, such as wheezes, crackles, and normal breath sounds, which are critical for diagnosing respiratory conditions in children. The authors collected a substantial dataset of pediatric lung sounds from a diverse patient population. The work by Topaloglu et al. [11] presents a novel deep learning model for asthma detection. This model integrates an attention mechanism with ResNet18 to analyze lung sounds, achieving high accuracy in classification. The research involves preprocessing, training, feature extraction, iterative feature selection, and classification using k-NN or SVM. It uses Grad-CAM for explainable AI, providing visual heat maps to distinguish asthma sounds. The authors report that their model achieves 99.73% accuracy. Additionally, beyond the lung sound use case, [12] applied various machine learning algorithms to cluster and classify datasets of heart sounds, including both normal and abnormal sounds. From a machine learning perspective, other prior work [13] exists – however without a focus on interactive visualization in a web application as in our system.

3. Background

We describe here the theoretical components for our analysis on audio scaling, time series clustering and time series compression.

3.1. Audio Scaling

Real-world data often exhibit variations in amplitudes, a crucial aspect to consider in machine learning applications. Peak normalization enhances the dynamic range, which is particularly useful when relative differences between recordings are more important than their absolute amplitude. Figure 3 illustrates the impact of peak normalization. The time series' maximum peak $\max(|T|)$ for a time series T of Eq. 1 is identified and each data point is rescaled relative to this value, resulting in the peak amplitude normalized to a maximum 1, or -1 in the case of negative values. First the time series' maximum peak is identified, mathematically denoted as $\max(|T|)$ for a time series T of Eq. 1. Each data point is then rescaled relative to the identified peak, resulting in the peak amplitude normalized to a maximum 1, or -1 in the case of negative values. We used the well-known Min-Max and Mean-Variance Scaler from tslearn library.

- **Min-Max Scaler** effectively maintains the original shape by scaling within a predefined range [a,b], chosen here as [-1,1]. It is sensitive to extreme values, skewing the waveform and not centering around zero. For a time series T the new value t_i' for a sample t_i is calculated as:

$$t_i' = a + (t_i - t_{\min}) \cdot \frac{(b-a)}{t_{\max} - t_{\min}} \tag{2}$$

- **Mean-Variance Scaler** calculates the z-Score of each sample, indicating how many standard deviations a data point deviates from the series mean. It addresses amplitude offsets, adjusting the mean of the time series to 0 and its standard deviation to 1. While effective for managing outliers, it can amplify noise and reduce interpretability due to

altered amplitude values. For time series T the new value t_i' for a sample t_i is calculated as:

$$t_i' = \frac{t_i - \mu}{\sigma} \tag{3}$$

where μ and σ are the mean and standard deviation of the series, respectively.

3.2. Euclidean k-means and Dynamic Time Warping (DTW) with Barycenter Averaging (BA) k-means

Our research focuses on the k-means algorithm [6] for distance-based clustering, which partitions data into a predetermined number of clusters k while minimizing the intra-cluster sum of squares. It operates by initially selecting k cluster centroids, positioned to be maximally distant from each other. It iteratively assigns time series to the nearest centroid and recalculates centroids until either a specified number of iterations is reached or the algorithm converges. K-means is chosen for its efficiency and simplicity, making it suitable for large datasets. Within our clustering framework, we explore TimeSeriesKMeans, a variant of the k-means algorithm provided by the tslearn library:

Euclidean k-means assumes equal-length time series. For time series $x = \{x_0, \ldots, x_{n-1}\}$ and $y = \{y_0, \ldots, y_{n-1}\}$, each being n steps long, the Euclidean distance d_{ed} is calculated as:

$$d_{\text{ed}}(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4}$$

This value quantifies similarity, favoring speed and simplicity with a time complexity of $O(n)$.

DTW Barycenter Averaging k-means addresses the limitations of the Euclidean case, accommodating time series of different lengths and phase-shifted events. It initializes a cost matrix M of size $n \times m$, where each element (i, j) represents the distance d between the respective data points. The optimal path π, representing the smallest distance among all possible paths, is continuous and increases gradually. DTW has a time complexity of $O(n^2)$. The DTW distance d_{DTW} between two time series is given by the following optimization formulation:

$$d_{\text{DTW}}(x, y) = \min_{\pi} \sqrt{\sum_{(i,j)\in\pi} d(x_i, y_j)^2} \tag{5}$$

The search for an optimal path can be restricted to a Sakoe-Chiba band [14]. Generally, constraints with a 10% width band relative to time series length are used, which speed up the calculation by a constant factor. In our implementation, we have expanded the Sakoe-Chiba band to 20% for wider data comparisons, particularly useful for capturing the sporadic nature of crackles and wheezes in respiratory sounds. The centroid calculation, or barycenter $\mathbf{b}$, for a group C of time series TS_i is an iterative process. Known as DTW Barycenter Averaging (DBA) [15], it is formalized as:

$$\min \sum_{i} d_{\text{DTW}}(\mathbf{b}, TS_i)^2 \tag{6}$$

The barycenter is refined in a loop until the squared distance is at a minimum, with each iteration finessing the output. The final barycenter average is the most representative of all series with minimal information loss.

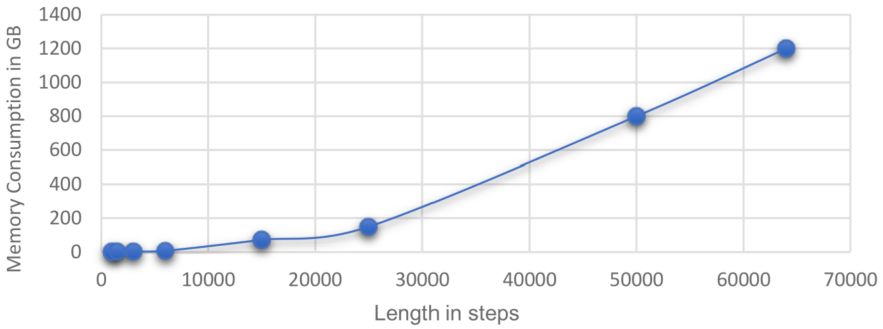


Figure 1: Memory consumption for DTW calculation for 10 time series with different resampling settings (length of time series in time steps)

3.3. Piecewise Aggregate Approximation

To tackle the computational challenges posed by the dataset's substantial size and high data density, especially for metrics like Dynamic Time Warping in a web application context that relies on CPU support, we employ Piecewise Aggregate Approximation (PAA) [16] which compresses a time series segment-wise – balancing between representing complex short sound events and computational efficiency. The original time series T (as defined in Eq. 1) with n data points is transformed into a new dimensionality of n' data points, where $1 \leq n' \leq n$. The compression rate c is defined as $c = \frac{n}{n'}$. Therefore, the speedup achieved can be quantified as the ratio of the original clustering complexity $O(n^2)$ to the reduced complexity of $O(n'^2)$. PAA offers several advantages, including a balance between representational accuracy and computational efficiency, reducing memory usage and algorithm's time complexity, while smoothing out noise and ringing effects, that occur in equal-length representation.

To demonstrate the necessity of PAA, ten random respiratory cycles from the ICBHI dataset were chosen and resampled to sizes ranging from 1,000 to 64,000 time steps. This test was only intended to measure speed and memory efficiency without evaluating the relevance or quality of the clustering results. The memory consumption of DTW is (without PAA) is shown in Figure 1. We experienced a "failed random access memory storage" error at 64,000 time steps. As this part of the files only accounts for 1/700 of the total dataset, it highlights the need for aggregation techniques to be used. Furthermore, the time complexity approached unacceptable levels, such as 36 minutes to analyze three cycles with 40000 steps each, making PAA essential.

In addition to computational considerations, a visual inspection was conducted to evaluate how varying the number of segments influences the proper capture of underlying trends, given the complex waveform characteristic of breath sounds. In Fig. 2 the longest cycle (a), which is most affected by a reduction in the number of segments and a randomly chosen one with rapid wave deflections due to vocal and background noise (b) were visualized. The number of segments used for the visual inspections varied from 1,000 to 2,500, with intervals incremented by 500. The use of PAA results in significant benefits. The dataset, previously unmanageable, can be fully clustered using the DTW metric within reasonable memory and time considerations.

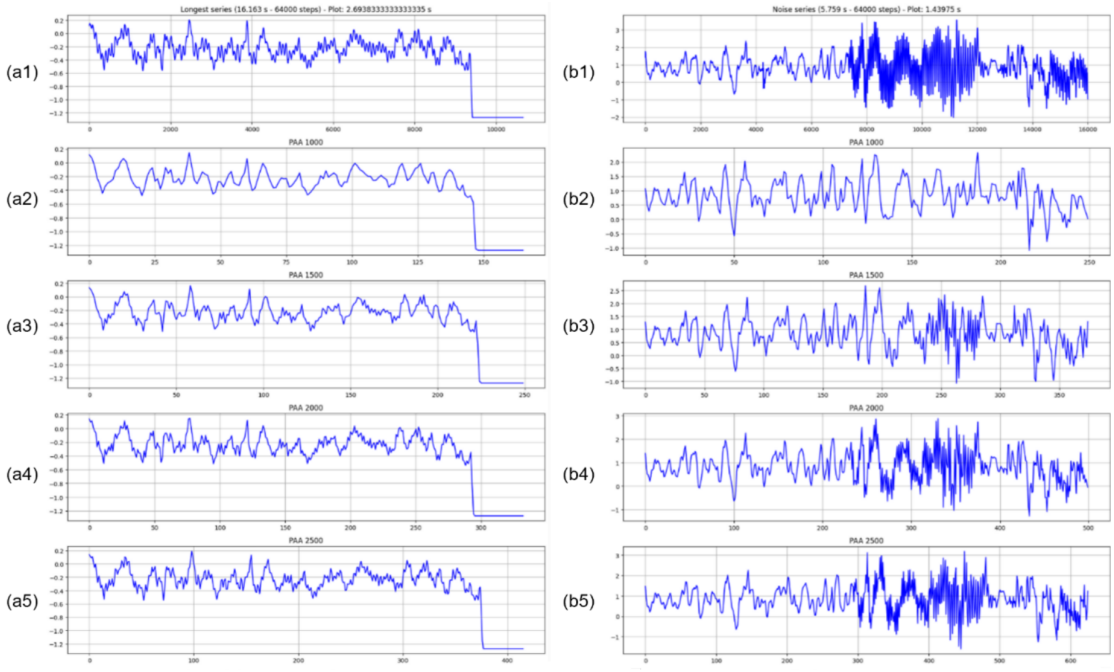


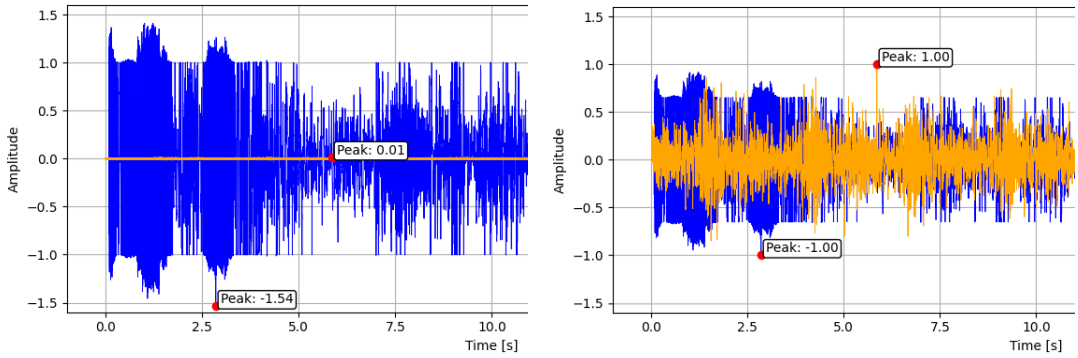
Figure 2: Respiratory cycles before and after PAA with different number of segments: (a) longest cycle; (b) noisy cycle; (1) resampled; (2) 1000 segments; (3) 1500 segments; (4) 2000 segments; (5) 2500 segments used for the respiratory cycle.

4. Implementation

4.1. Data Upload

Initiating with the data upload, users input necessary audio data and metadata. The first step allows users to upload desired or all audio files together with patient information (demographic data, diagnoses, detailed acquisition information, and annotation notes) for each recording. The following approach ensures high audio faithfulness and compatibility with the application's features, considering the diversity of audio file parameters.

File Format Considerations and Uniformity: The audio files in WAV format show variations in sample rates and bit depths (refer to Table 1). To standardize the subsequent analysis, all recordings are processed at a predefined sample rate of 16 kHz while being uploaded. This decision is aligned with the Nyquist-Shannon sampling theorem [17, 18]: for accurate signal representation, the sampling frequency should be at least twice as high as the signal's highest frequency component. The highest frequency component in lung sounds is around 4 kHz, so that a sampling rate of 8 kHz and higher effectively captures the relevant frequency range, without information loss.



(a) Before peak normalization

(b) After peak normalization

Figure 3: Peak normalization on two time series with marked initial peaks: (a) before and (b) after peak normalization. Before normalization, the orange series appears flat. Comparing both in a common plot would require extensive zooming in and out.

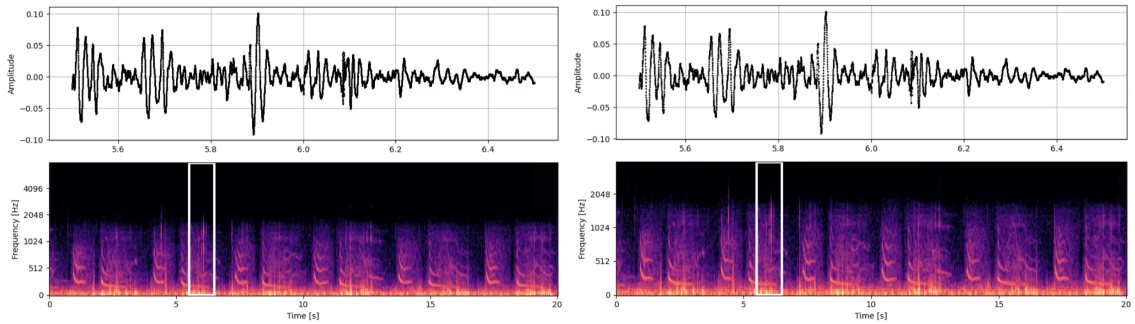
Library Selection and Rationale: Our selection of the librosa [19] library was based on its robust handling of WAV files and efficiency in converting them into numerical arrays suitable for subsequent analysis. Compared to wave [20] or SciPy [21], librosa offers a balance of functionality and performance, particularly in dealing with the challenges posed by 24-bit audio and the need for a standardized sample rate. Furthermore, it is known for its wide range of functionalities, including but not limited to signal preprocessing and spectral analysis, which are valuable for the next tasks in our application's workflow.

4.2. Downsampling

Since the essential frequencies in respiratory sounds are below 4 kHz, an 8 kHz sample rate effectively captures the pertinent audio information, thus maintaining necessary detail while reducing the load on computational resources. Utilizing the librosa library, downsampling to 8 kHz ensures data integrity for both single and multi-file visualizations, optimizing the trade-off between audio quality and visualization performance. Our approach employs band-limited sinc interpolation [22], suitable for digital audio processing. It relies on the sinc function, centered on the corresponding time position, scaled accordingly and summed to reconstructs a continuous signal. The sinc function itself acts as an exemplary low-pass filter, suppressing frequencies above the Nyquist frequency to negligible levels, mitigating aliasing effects and ensuring distortion-free visualization of the audio data (cf. Figure 4).

4.3. Domain Exploration

Our app features two visualization domains. *Time domain:* It illustrates how the signal's amplitude varies over time, with each sample in the numerical array corresponding to a specific time point in the recording. This domain is suitable for intuitively analyzing aspects such as the duration, onset, and decay of respiratory sounds events. *Time-Frequency domain:* The time-



(a) 16 kHz sample rate (b) 8 kHz sample rate

Figure 4: Audio segment with crackle and wheeze sounds, and the corresponding mel spectrogram of the full recording, with the audio segment highlighted by a white rectangle. (a) 16 kHz sample rate; (b) 8 kHz sample rate.

frequency domain provides a two-dimensional perspective, achieved through the Short-Time Fourier Transform [23]. This process segments the signal into short, overlapping intervals applying windowing. The relationship between time and frequency resolution involves a trade-off primarily determined by the window size. Spectral leakage, caused by abrupt changes at signal edges, can introduce artifacts such as ripples [24]. To counteract this, a Hann-shaped window of 64ms is applied, resulting in a frequency resolution of approximately 15.7 Hz. The mel spectrogram [24], utilized here (see Fig. 4), aligns with human auditory perception by converting the standard spectrogram's frequency axis to the mel scale, which is effectively linear in lower frequencies and logarithmic in higher ones.

4.3.1. Library Selection

Three visualization libraries were considered: Plotly (GO and Express) [25], Bokeh [26] and Matplotlib [27].Bokeh proved optimal for time domain visualizations due to its excellent performance in processing large data points, fast rendering speed, and interactive exploration features such as panning, zooming, range tool and point hovering. These capabilities significantly enhance user experience. For the time-frequency domain, where interactivity is less critical, Matplotlib showed to be efficient, due to its integration with librosa. We integrated additional interactive elements, such as checkboxes and forms, to engage users with mel spectrogram plots more effectively. Our library choices align with our aim to maximize performance and ensure a smooth user experience in the web application.

4.4. Web Application User Interface

The homepage, illustrated in Figure 6 in Appendix A, serves as a guide, providing essential information about the application's capabilities, outlining necessary file types, data usage and link to the relevant database. The sidebar updates in real-time (Figure 7, Appendix A), enriching the user experience. Each page follows a systematic workflow, starting with an introduction to its functionalities, followed by automatic background checks ensuring all

necessary preprocessing is complete. Next, different columns are displayed, each containing a separate form for performance efficiency, in which options regarding the functionality are given. When the submit button is triggered, the user receives real-time feedback via progress bars, status updates and sidebar updates. **Upload Files** page enables file selection and upload, either choosing from recordings stored on the server side, made accessible via a *multiselect* widget or WAV audio files directly from their devices via a *file upload* widget. Following this, a *submit button* confirms the user's choice and stores files as time series in numerical array representation. Before submission, the chosen files can be reviewed, removed, or even added. **Data Entry and Extraction** sections enable users to access data, specifically relevant to their uploaded files, and store it in the session state using dictionaries and lists. A customized *tab* is integrated, facilitating user interactivity through informative hints and interactive buttons. Upon submit, a comprehensive DataFrame is created containing all patient-relevant data, which serves as a repository for future tasks reference and use. **Extract respiratory cycles** page mirrors the structure of others. Users can select the desired length in seconds for equal-length resizing; however, the value of four seconds is recommended. With a simple click, the process begins. The user gets feedback in real time upon each extracted cycle through a progress bar, as well as a permanent notification on the sidebar, regarding the exact number of extracted cycles and utilized method. **Clustering** page offers a comprehensive suite of preprocessing functionalities. Within two separate forms (see Figure 8(a) and Figure 8(b) in Appendix A), users can prepare the dataset for clustering, deciding on preprocessing steps such as scaling (None, Min-Max, Z-Score) and Piecewise Aggregate Approximation (number of segments), as well as configure the clustering algorithm TimeSeriesKMeans, choosing the distance function, number of clusters, initialization method and Sakoe-Chiba band radius. Upon completion, textual and graphical representations of cluster cardinality, including label distribution per cluster are output (see Figure 8(c), Appendix A). Interactive plots display cluster centroids and previews of the initial 20 time series in each cluster. A table displays metadata for each file in every cluster, leveraging information from the "Data Entry and Extraction" page. An evaluation metric set is provided, alongside model-specific metrics like the number of iterations, inertia, and computation times. **Visualization Waveform & Visualization Spectrogram** are two distinct web pages that share a similar structure. Each visualization task, either single- or multi-file, is accompanied by a structured form to enable file selection and other domain-specific options. At the core of the rendered image is either the time-domain waveform (see Fig. 9) or mel-scaled spectrogram (see Fig. 10), complemented by an audio playback function, conveniently located on top of the graph. This combined approach ensures that users have multi-sensory experience. For instance, wheezes can be both heard as a musical sound and seen as a sinusoidal waveform and horizontal lines in the mel spectrogram. Accompanying text tables provide patient information, and annotations within the visualization detail respiratory cycle specifics like start and end times and color-sound categories. Specifically in time domain interactive elements such as zoom, move, pan, range and hover functionalities, allow users to engage the waveform as desired and understand subtle details in the audio signal. The time-frequency domain offers user-adjustable frequency ranges, enlarging and saving figures for later use. These features facilitate a deeper understanding of the respiratory sounds and support comprehensive analysis by medical professionals.

Table 1
Detailed information about the ICBHI Respiratory Sound Database [7]

Parameter	Details
Number of recordings	920
Sample rate	4 kHz; 10 kHz; 44.1 kHz
Bits per sample	16; 24
Recording duration	21.5±8.3 s (mean ± standard deviation)
Recording equipment	AKG C417L Microphone; 3M Littmann Classic II SE Stethoscope; 3M Littmann 3200 Electronic Stethoscope; WelchAllyn Meditron Master Elite Electronic Stethoscope
Number of participants	126
Diagnosis (number of participants)	COPD (64); Healthy (26); URTI (14); Bronchiectasis (7); Pneumonia (6); Bronchiolitis (6); LRTI (2); Asthma (1)
Sex* (# participants)	Male (79); Female (46)
Age of children*	4.8±4.6 years (mean ± standard deviation)
Age of adults*	67.6±11.5 years (mean ± standard deviation)
Patient information*	Adult: Body Mass Index; Child: Height and Weight
# respiratory cycles	6,898
Annotation (# cycles)	Normal (3,642); Crackle (1,864); Wheeze (886); Both (506)
Recording duration	2.7±1.2 s (mean ± standard deviation)
*Note: Instances of missing or unavailable information.	

5. Experiments and Results

The application was implemented in Python with a Streamlit frontend[1] and deployed in a Docker container and executed on a high-performance computational platform including an AMD EPYC 7742 64-Core Processor and 1.0 TiB of RAM with an Ubuntu operating system. We focused on evaluating the effect of the scaling and compression techniques discussed in Section 3 on clustering algorithms' effectiveness.

A dedicated pipeline is set up, commencing with extraction of respiratory cycles from the uploaded audio files, using accompanying annotations. Slicing the DataFrames according to start and end times, we create a dataset comprising 6,898 respiratory cycles for subsequent clustering. We explored the impact of various scaling techniques on clustering outcomes for cluster count $k = 2$ (normal versus adventitious sounds); the results are summarized in Table 2. For the Euclidean metric, a maximum iteration parameter of 300 was used. On the contrary, for the DTW with DBA, PAA was integrated with 2000 segments, 50 iterations for initialization and barycenter averaging, and a Sakoe-Chiba band was set at 20% width. This aggregating strategy allows for detailed representation, while significantly speeding up the algorithm's running time. Consequently, the previously unmanageable database is fully clusterable using the DTW metric, within reasonable memory and time constraints, and with reduced space complexity for the cost matrix. PAA also aligns well with our goal, maintaining the numerical integrity of time series, where cycles are aggregated so that at least one segment encompasses a duration of 5ms. PAA's application to our extensive dataset demonstrated its efficacy in managing computational

[1]Source code on Github: https://github.com/sindiveliko/DigitaLung_WebApp.

challenges. PAA successfully smoothed out artifacts while maintaining the integrity of key respiratory events like crackles and wheezes, enabling efficient dataset clustering.

Unscaled Data: Both metrics faced challenges with unscaled data, particularly with skewed cluster cardinality. For the Euclidean case, this results in significantly imbalanced clusters some of which contain very few to single cycles, often distorted or clipped due to loud and incorrect auscultation. This might be useful for outlier detection but requires careful examination.

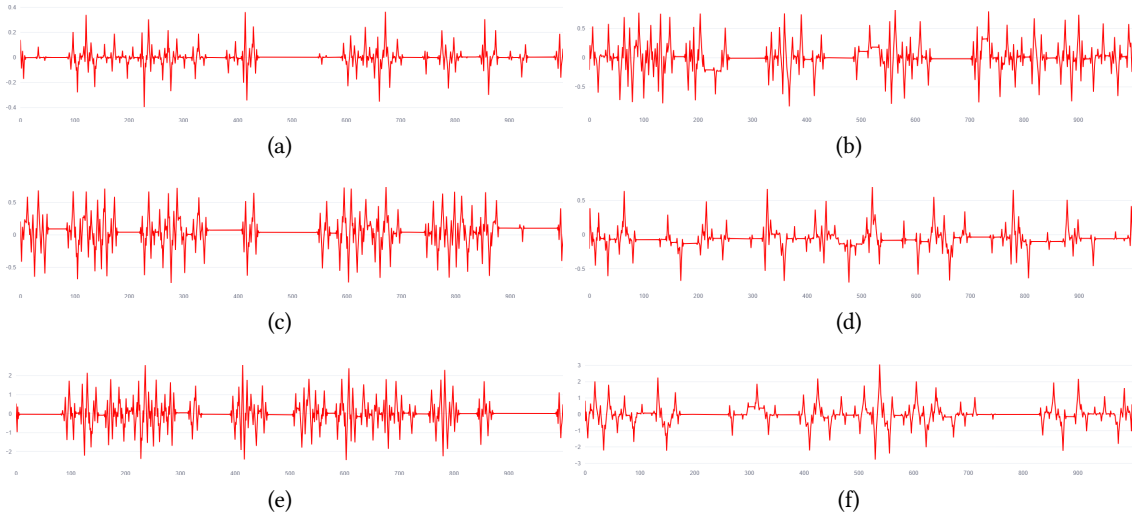


Figure 5: Centroids of adventitious (5(a)) and normal (5(b)) sounds resulting from dataset clustering with DTW without scaling; adventitious (5(c)) and normal (5(d)) sounds with Min-Max Scaler; adventitious (5(e)) and normal (5(f)) sounds with Mean-Variance Scaler.

DTW, in contrast, achieved a more balanced distribution across clusters. Clusters with a majority of normal respiratory sounds typically exhibited simpler centroid waveform with amplitudes capped at 0.5, whereas that of adventitious sounds presented higher amplitudes and irregular shapes. This hints at the influence of the raw signal amplitude on clustering, as the algorithm initially seems to be driven by these amplitude differences, highlighting a potential risk of misclassification due to noise as the clustering progresses. The silhouette score points to a better partitioning in comparison to other scaling methods.

Min-Max Scaler: The use of Min-Max scaling evenly distributed time series across clusters, with cardinalities around 70%-30% range for Euclidean and 60%-40% range for DTW. Notably, under DTW, the cluster centroids showed marked differences in their shapes, though they generally fell within the same amplitude range. A decrease in silhouette scores points to amplitude differences being eliminated. This shift suggested that Min-Max is influenced more by the shape of the time series, rather than amplitude values. Consequently, the clusters may appear less compact than those formed without scaling, as the time series previously considered "far apart" in their original form are now assigned to the same group. This scaler might be more impactful for results in combination with DTW, than other ones.

Mean-Variance Scaler: Z-Score normalization significantly impacted cluster formation, aligning with expected distributions, but complicated interpretation. A sharp drop in the silhouette

score to nearly a quarter of the non-normalized case, along with a sharp rise in inertia by a factor of 10 suggest that time series were spread far around their respective centroids and lied close to the boundaries between clusters. This makes it challenging to discern clean groupings. Furthermore, the centroids after Z-Score normalization varied considerably, ranging between -3 and 3 units. The range of individual time series varied greatly, making it challenging to discern distinct groupings. This variability pointed to the complexity of applying Z-Score normalization in this context. The differences discussed above are illustrated in Figure 5.

Table 2
Clustering results for $k = 2$.

	Scaling Method	Silhou. Score	Inertia	Rand Index	Purity Average	Clust. 1 / Clust. 2
k-means	None	0.33	3205.6	0.00024	52%	8% / 92%
k-means	Min-Max	0.062	6023.1	-0.000032	55%	75% / 25%
k-means	Z-Score	0.0049	63664.8	-0.000066	53%	51.3% / 48.7%
DTW	None	0.46	11.4	0.022	57.6%	70.9% / 29.1%
DTW	Min-Max	0.11	19.08	0.024	57.7%	48.7% / 51.3%
DTW	Z-Score	0.099	194.3	0.030	59.3%	56.7% / 43.3%

6. Conclusion and Future Work

Unsupervised learning methods offer potential in respiratory sound data analysis. Our web app utilizes the distinctive patterns in individual breath cycle waveforms for differentiation with the aim to develop and validate a machine learning-based digital auscultation system to enhance patient care. We presented features that are implemented directly in the web app, eliminating the need for external infrastructure. Overall, our analysis indicated that while unscaled data and the Euclidean metric might be more suitable for outlier detection, DTW, especially when combined with scaling techniques like Min-Max, provided more refined clustering. However, Mean-Variance Scaler seemed to complicate the clustering landscape, potentially due to its alteration of the original time series characteristics. The presence of outliers significantly influences the overall outcome, necessitating careful considerations of such anomalies. These insights highlight the impact of scaling techniques on time series clustering. Moreover, the initial random centroid selection could introduce bias. In future work we will extend our approach to supervised machine learning [28, 11]. In sum, our web app shows the potential of applications that offer services to process and interpret lung sound data, while in the future we aim to adapt our web service to support heart sounds as well.

Acknowledgements

This work was partially funded by the BMBF under project number 13GW0554B (*Digitalung*) and partially supported by the Fraunhofer Internal Programs under Grant No. Attract 042-601000.

References

[1] C. Qiu, W. Zeng, W. Tian, J. Xu, Y. Tian, C. Zhao, H. Liu, Wearable stethoscope for lung disease diagnosis, Sensors & Diagnostics 3 (2024) 281–286.

[2] J. B. Soriano, P. J. Kendrick, K. R. Paulson, V. Gupta, et al., Prevalence and attributable health burden of chronic respiratory diseases, 1990 – 2017: a systematic analysis for the global burden of disease study 2017, The Lancet Respiratory Medicine 8 (2020) 585–596. doi:10.1016/S2213-2600(20)30105-3.

[3] K. Priftis, L. Hadjileontiadis, M. L. Everard, Breath Sounds: From Basic Science to Clinical Practice, Springer, 2018. doi:10.1007/978-3-319-71824-8.

[4] P. Esling, C. Agon, Time-series data mining, ACM Computing Surveys 45 (2012) 1–34. doi:10.1145/2379776.2379788.

[5] S. Aydin, Time series analysis and some applications in medical research, Journal of Mathematics and Statistics Studies 3 (2022) 31–36. doi:10.32996/jmss.2022.3.2.3.

[6] T.-C. Fu, A review on time series data mining, Engineering Applications of Artificial Intelligence 24 (2011) 164–181. doi:10.1016/j.engappai.2010.09.007.

[7] B. M. Rocha, D. Filos, L. Mendes, G. Serbes, et al., An open access database for the evaluation of respiratory sound classification algorithms, Physiological Measurement 40 (2019) 035001. URL: 10.1088/1361-6579/ab03ea. doi:10.1088/1361-6579/ab03ea.

[8] Y. Alsouda, S. Pllana, A. Kurti, A machine learning driven iot solution for noise classification in smart cities, arXiv preprint arXiv:1809.00238 (2018).

[9] J. Han, K. Qian, M. Song, Z. Yang, et al., An early study on intelligent analysis of speech under covid-19: Severity, sleep quality, fatigue, and anxiety, arXiv preprint arXiv:2005.00096 (2020).

[10] J. S. Park, K. Kim, J. H. Kim, Y. J. Choi, K. Kim, D. I. Suh, A machine learning approach to the development and prospective evaluation of a pediatric lung sound classification model, Scientific Reports 13 (2023) 1289.

[11] I. Topaloglu, P. D. Barua, A. M. Yildiz, T. Keles, S. Dogan, M. Baygin, H. F. Gul, T. Tuncer, R.-S. Tan, U. R. Acharya, Explainable attention resnet18-based model for asthma detection using stethoscope lung sounds, Engineering Applications of Artificial Intelligence 126 (2023) 106887.

[12] Y. Zeinali, S. T. A. Niaki, Heart sound classification using signal processing and machine learning algorithms, Machine Learning with Applications 7 (2022) 100206.

[13] T. Xia, J. Han, C. Mascolo, Exploring machine learning for audio-based respiratory condition screening: A concise review of databases, methods, and open issues, Experimental Biology and Medicine 247 (2022) 2053–2061.

[14] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE transactions on acoustics, speech, and signal processing 26 (1978) 43–49.

[15] F. Petitjean, A. Ketterlin, P. Gançarski, A global averaging method for dynamic time warping, with applications to clustering, Pattern Recognition 44 (2011) 678–693. doi:10.1016/j.patcog.2010.09.013.

[16] E. J. Keogh, M. J. Pazzani, Scaling up dynamic time warping for datamining applications, in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000, pp. 285–289. doi:10.1145/347090.347153.

[17] V. A. Kotelnikov, On the transmission capacity of the “ether” and wire in electrocommunications, in: Modern Sampling Theory: Mathematics and Applications, Springer, 2001, pp. 27–45.
[18] C. E. Shannon, Communication in the presence of noise, Proceedings of the IRE 37 (1949) 10–21.
[19] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, O. Nieto, librosa: Audio and music signal analysis in python, in: Proceedings of the 14th python in science conference, volume 8, 2015, pp. 18–25.
[20] Python Software Foundation, wave — read and write wav files, Python 3.10 Documentation, 2023. URL: https://docs.python.org/3.10/library/wave.html, accessed on 04.09.2023.
[21] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, et al., Scipy 1.0: Fundamental algorithms for scientific computing in python, Nature Methods 17 (2020) 261–272.
[22] J. O. Smith, Digital Audio Resampling Home Page, http://www-ccrma.stanford.edu/~jos/resample/, 2002.
[23] J. O. Smith, Spectral Audio Signal Processing, http://ccrma.stanford.edu/~jos/sasp/, 2011. Online book, 2011 edition.
[24] M. Müller, Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications, 1 ed., Springer Cham, 2015. doi:10.1007/978-3-319-21945-5.
[25] Plotly Technologies Inc., Collaborative data science, Plotly Technologies Inc., Montreal, QC, 2023. Retrieved September 10, 2023, from https://plotly.com/python/.
[26] Bokeh Development Team, Bokeh: Python library for interactive visualization, http://bokeh.org/, 2022.
[27] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering 9 (2007) 90–95.
[28] F.-S. Hsu, S.-R. Huang, C.-W. Huang, C.-J. Huang, Y.-R. Cheng, C.-C. Chen, J. Hsiao, C.-W. Chen, L.-C. Chen, Y.-C. Lai, et al., Benchmarking of eight recurrent neural network variants for breath phase and adventitious sound detection on a self-developed open-access lung sound database—hf_lung_v1, PLoS One 16 (2021) e0254134.

A. Screenshots

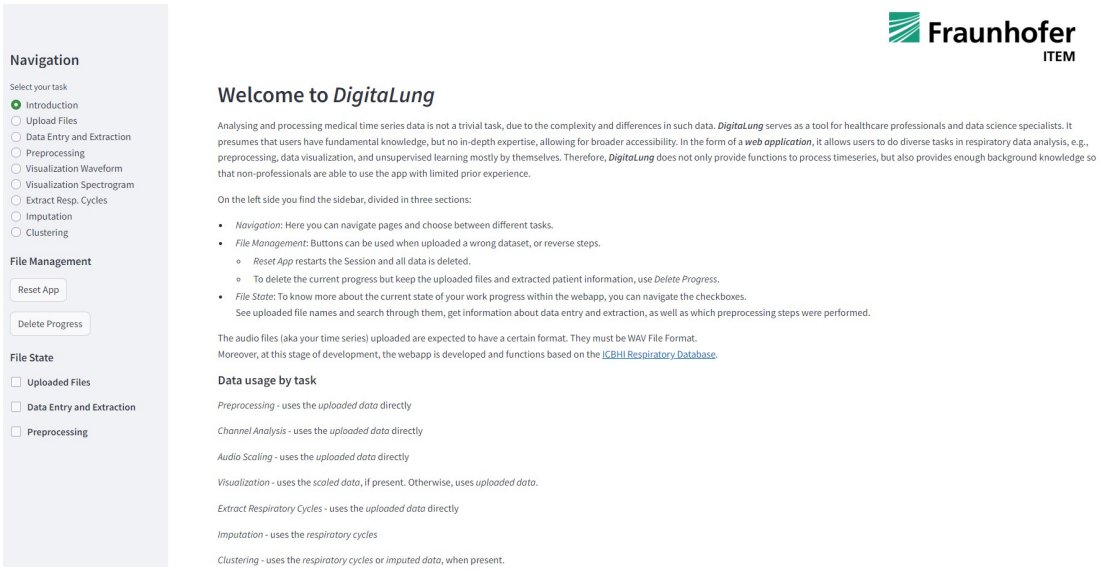
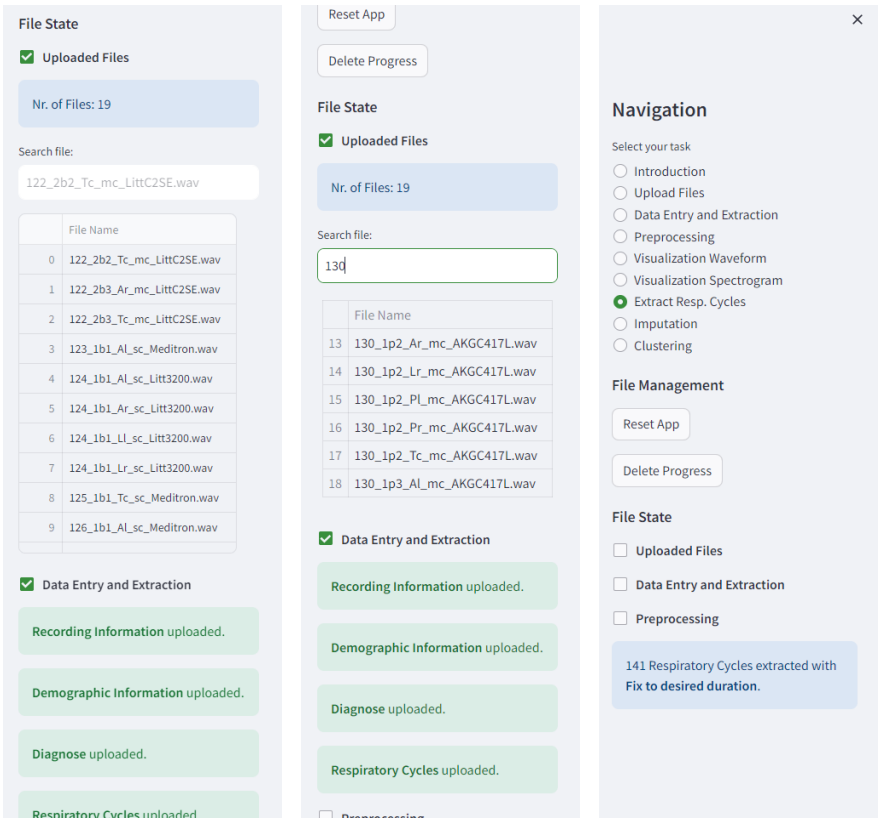
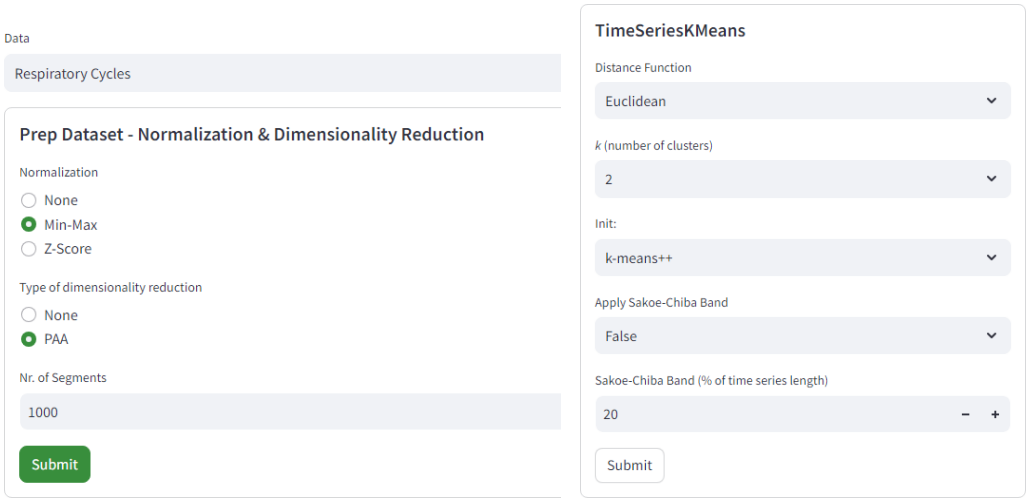


Figure 6: Homepage of the web application



(a) (b) (c)

Figure 7: Sidebar with various outputs: (a) audio files and patient data are uploaded, and information is displayed; (b) user searches for patient ID “130” and the table updates upon pressing “Enter”; (c) there is a notification regarding extracted respiratory cycles.



(a) Preparation Form

(b) Algorithm Parameter Form

(c) Page output

Figure 8: Clustering page: (a) Status element and Form for dataset preparation; (b) Form for TimeSeriesKMeans algorithm and initiating clustering; (c) a section of the clustering output.

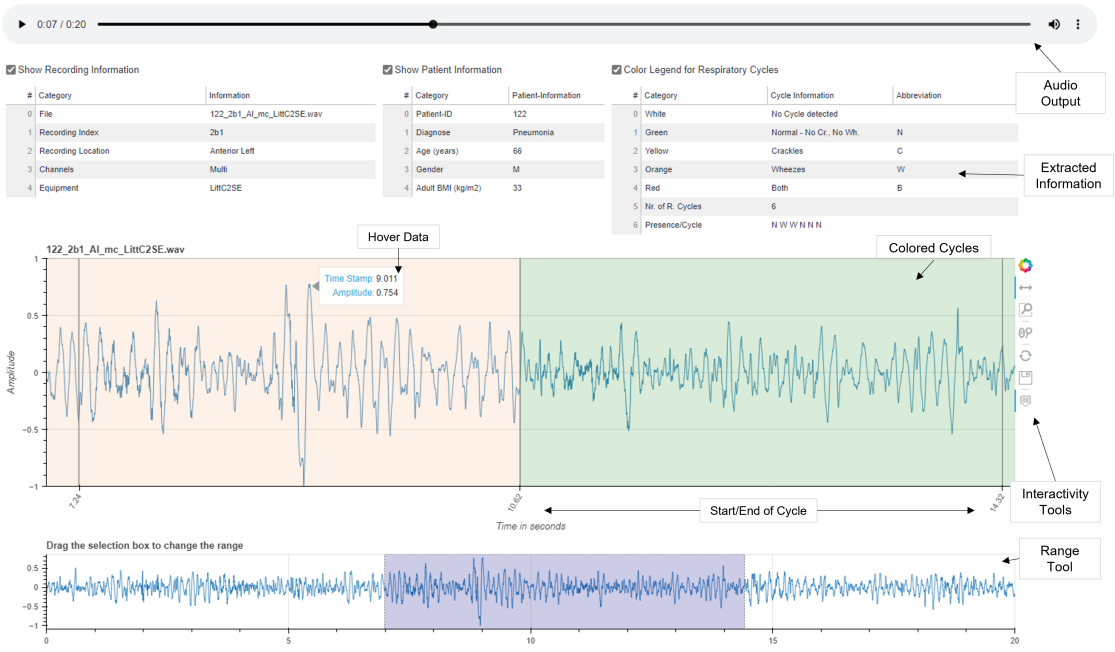


Figure 9: Time domain visualization: Single-file plotting with all options.

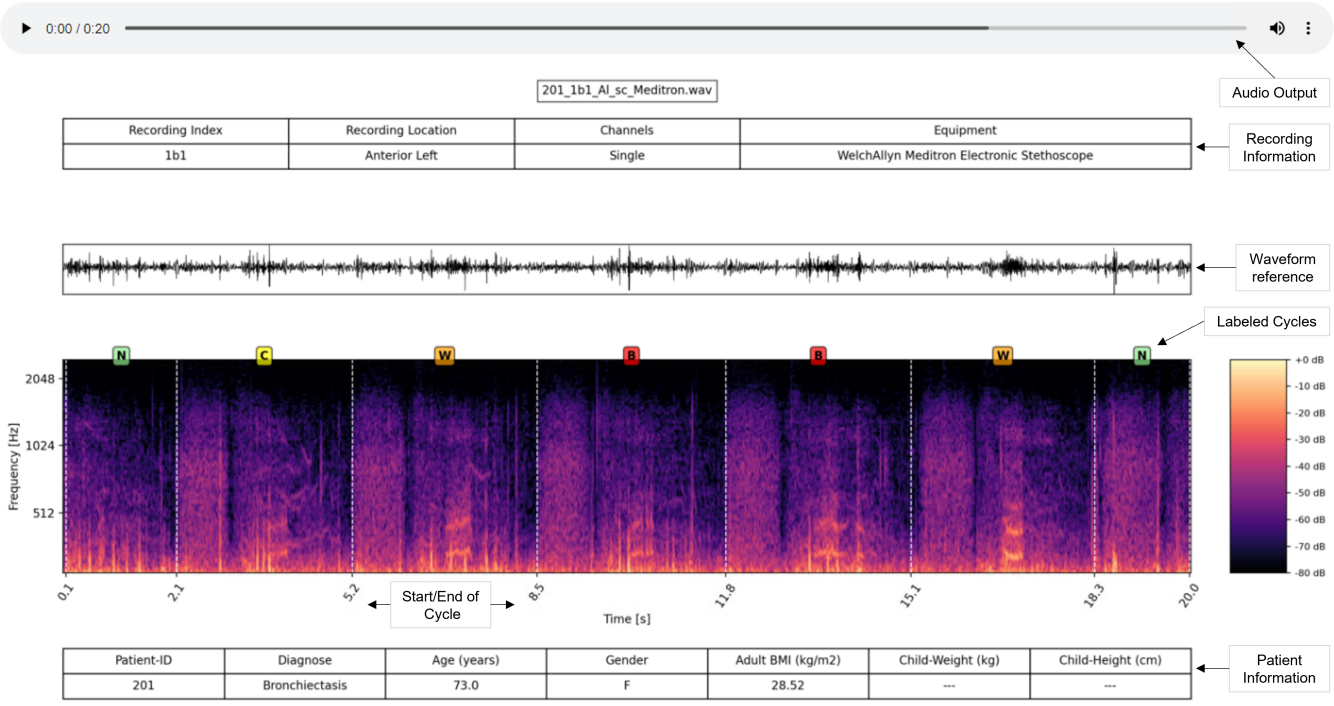


Figure 10: Time-Frequency Domain: Single-file plotting with all options.