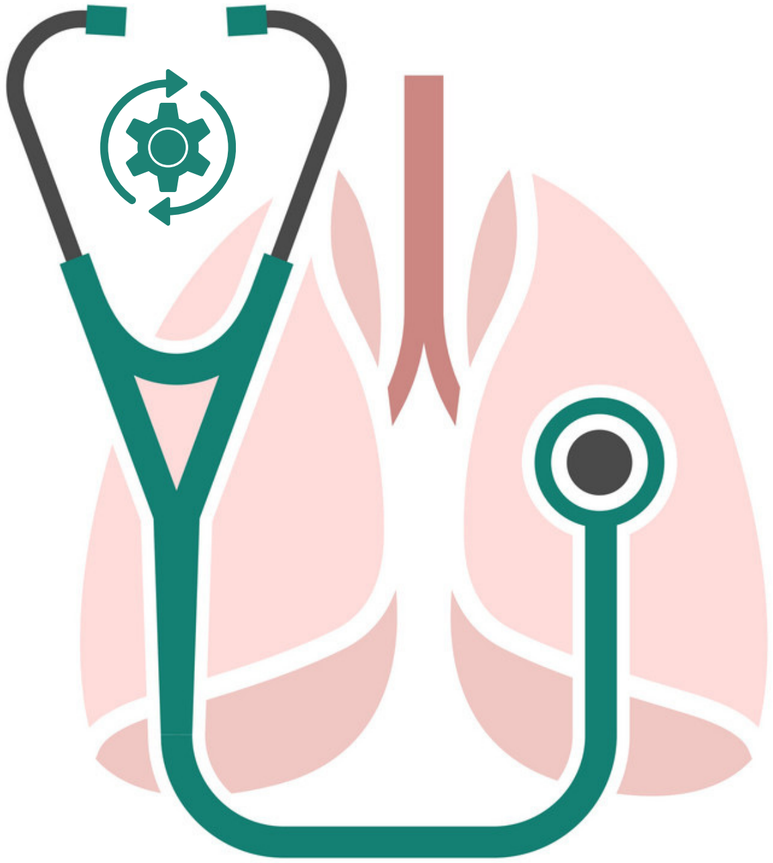


LUNG SOUND RECORDER

The Quad Chips

Team 38

Sai Praneeth - 2022101097

Jahnavi - 2022101118

Vaishnavi Priya - 2022101108

Sai Divya - 2022101090

Associated Professor - Abhishek Srivatsava

Associated TA - Santhoshini Thota

ABOUT THE STETHOSCOPE

The stethoscope is a medical device for auscultation

The chestpiece usually consists of two sides:

- Diaphragm (plastic disc) : Ex.Body Sounds

Diaphragm transmits higher frequency sounds

- Bell (hollow cup) : Ex.Skin vibrations

Bell transmits low frequency sounds.

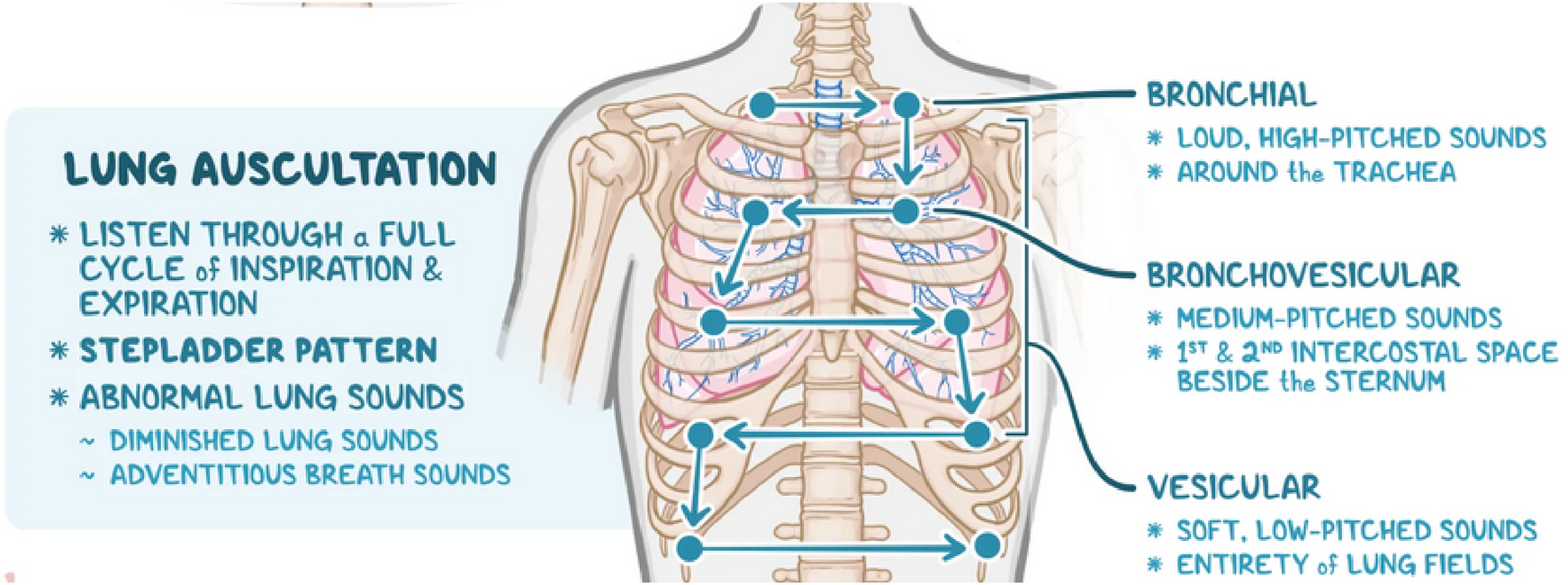
The Acoustic pressure waves travel up the tubing to the listener's ears.

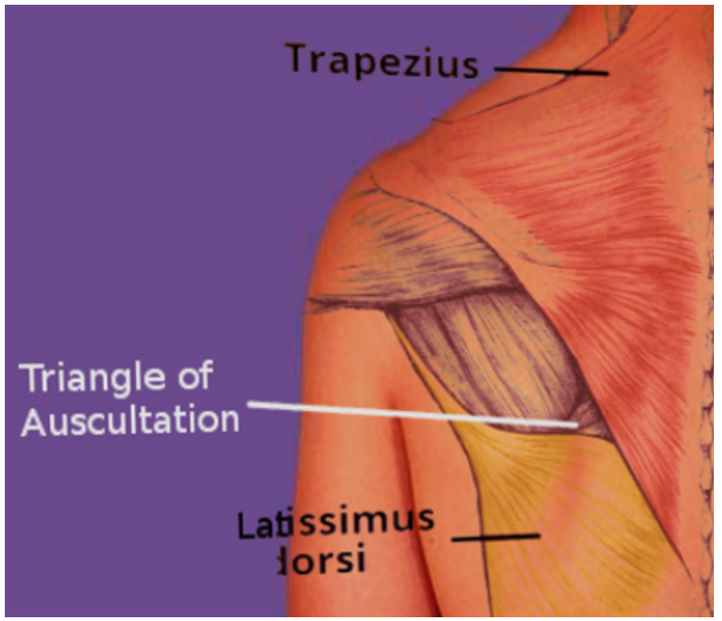
The tube connecting into the chamber between bell and diaphragm is open on only one side and can rotate. Rotating the tube 180 degrees in the head connects it to the diaphragm.

resource:https://en.wikipedia.org/wiki/Stethoscope

HOW TO USE IT FOR LUNG SOUNDS?

Auscultation of lungs sounds should be conducted over the anterior and posterior chest walls.





The auscultation points of the lungs coincide with the type of breath sounds heard and include the area around the trachea, the area between the 1st and 2nd intercostal space on both the anterior and posterior sides of the chest, and each lateral lung field

Auscultating the lungs,we will listen for a full cycle of inspiration and expiration using a stepladder pattern

The triangle of auscultation allows us to listen better to the patient's lungs and guide them to the proper diagnosis-Main Auscultation point.

resource:https://www.physio-pedia.com/Auscultation

Different Methods to classify sounds

1) CNN (Convolutional Neural Networks)
2) SVM’s (Support Vector Machines)
3) Random forests
4) HMM’s (Hidden Markov models)

CNNs are able to extract features from the data and learn complex patterns.

SVMs, random forests, and HMMs are also viable methods for classifying lung sounds, but they may not be as accurate as CNNs, especially for large datasets.

Hence using CNN’s are more accurate.

Characteristic	CNN	SVM	Random forest	HMM
Type of Model	Deep Learning	Machine Learning	Ensemble learning	Statistical
Strength	Able to extract features and learn complex patterns from data	Good at finding hyperplanes to separate data points into classes	Robust to overfitting	Good at modeling sequential data
Weakness	Can be computationally expensive to train	Can be sensitive to the choice of kernel function and other hyperparameters	Can be difficult to interpret the results	Can be difficult to train on small datasets
Best Suited for	Large datasets with labeled data	Smaller datasets with labeled data	Smaller datasets with labeled data	Smaller datasets with labeled data, especially sequential data

source:https://www.sciencedirect.com/science/article/abs/pii/S0933365717302051

SPECTRAL SUBSTRACTION

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

def spectral_subtraction(input_file, noise_file, output_file):

    fs, audio_data = wavfile.read(input_file)
    fs_noise, noise_data = wavfile.read(noise_file)

    if len(audio_data.shape) == 2:
        audio_data = np.mean(audio_data, axis=1)
    if len(noise_data.shape) == 2:
        noise_data = np.mean(noise_data, axis=1)

    spectrum_audio = np.fft.fft(audio_data)
    spectrum_noise = np.fft.fft(noise_data)

    magnitude_audio = np.abs(spectrum_audio)
    phase_audio = np.angle(spectrum_audio)
    magnitude_noise = np.abs(spectrum_noise)
    phase_noise = np.angle(spectrum_noise)

    noise_magnitude = magnitude_noise

    enhanced_magnitude = np.maximum(magnitude_audio - noise_magnitude, 0)
    enhanced_spectrum = enhanced_magnitude * np.exp(1j * phase_audio)

    enhanced_audio = np.fft.ifft(enhanced_spectrum).real.astype('int16')

    wavfile.write(output_file, fs, enhanced_audio)

input_file = "recording-lung.wav"
noise_file = "recording-new.wav"
output_file = "output_audio.wav"

spectral_subtraction(input_file, noise_file, output_file)
```

Spectral substraction has been used to remove the continuous noise recorded from the microphone while recording the lung sound.

```python
import numpy as np
import librosa
import os
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
```

```python
# Step 1: Feature Extraction with Padding
def extract_features(audio_file, max_length):
    y, sr = librosa.load(audio_file, sr=None)
    mfccs = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13)
    spectral_centroid = np.mean(librosa.feature.spectral_centroid(y=y, sr=sr))
    spectral_bandwidth = np.mean(librosa.feature.spectral_bandwidth(y=y, sr=sr))
    zero_crossing_rate = np.mean(librosa.feature.zero_crossing_rate(y=y))

    if mfccs.shape[1] < max_length:
        pad_width = max_length - mfccs.shape[1]
        mfccs = np.pad(mfccs, ((0, 0), (0, pad_width)), mode='constant')
    else:
        mfccs = mfccs[:, :max_length]

    return np.concatenate((mfccs.flatten(), [spectral_centroid, spectral_bandwidth, zero_crossing_rate]))
```

Resource for datasets-https://doi.org/10.17632/jwyy9np4gv.3

WEB-SOCKETS

We have implemented web-sockets for seamless interactions between the client & server.

- Normal HTTP requests have to be initiated by a client, and server can’t respond with no request.
- This issue is tackled by the web-sockets, where it enables two-way communication by emitting & listening to events.
- This becomes an advantage when we want to initiate / control the recording from the client (user / doctor).

Integration of UI with audio processing

We already have coded the audio processing part. It takes a audio file from the directory and processes it. After we have enabled the transfer of audio file to the web server, now, it goes through all the audio processing.

- With the integration of UI, user can initiate the recording from ESP and can access the audio file from the UI itself.
- Updated some UI files to access data from the database.

Further Plans

Developing a front-end interface: Data visualization, admin controls - for setting the thresholds.

Real-time audio integration: Currently, audio is processed on some trigger. We will try to automate that by recording audios with some delay and processing it in between (alternating). And build an alerting system based on the real-time processed data.

Classifying diseases: Each disease has certain set of lung sounds. Based on similarities with the reference audios and thresholds, we classify.

Localized processing on a remote device: If time and resources permit

Thank you!

Team 38, Group 4