Design of a digital stethoscope-type device for the detection of carotid artery bruits for early stroke prediction

Signal Processing and Machine Learning Data Processing Pipeline

(Classification: Normal vs Murmur vs Artifact)

SOTA methods of stroke risk prediction

Imaging Techniques: CT angiography, ultrasound, MRI

Invasive Option: Catheter angiography for special cases

Additional Tools: ECG, echocardiography, cardiac MRI (cMRI)

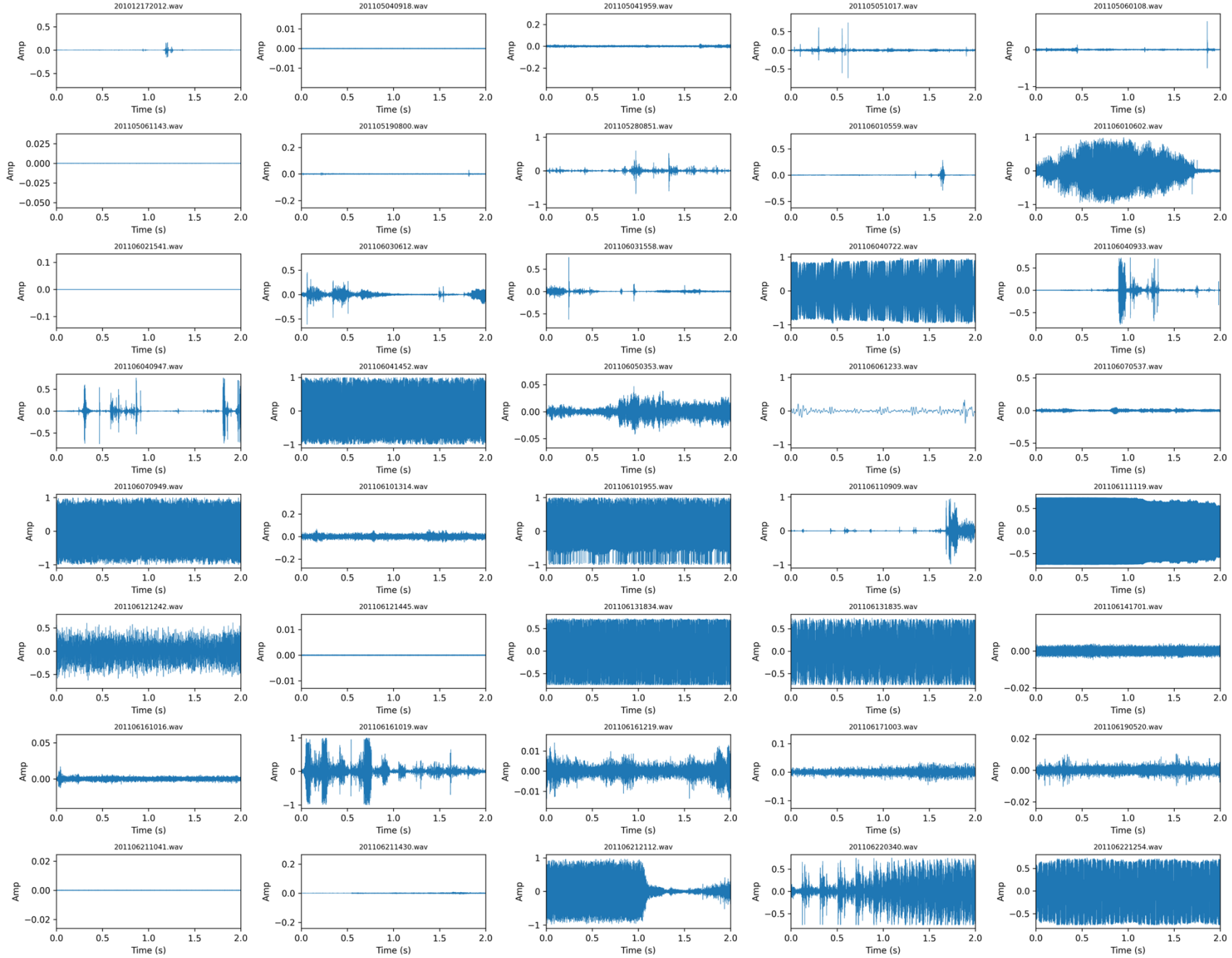
Challenges:

- High cost and complex hardware
- Requires trained technicians and expert interpretation
- Limited to major medical centers
- Inaccessible in many underserved areas

Unaddressed need:

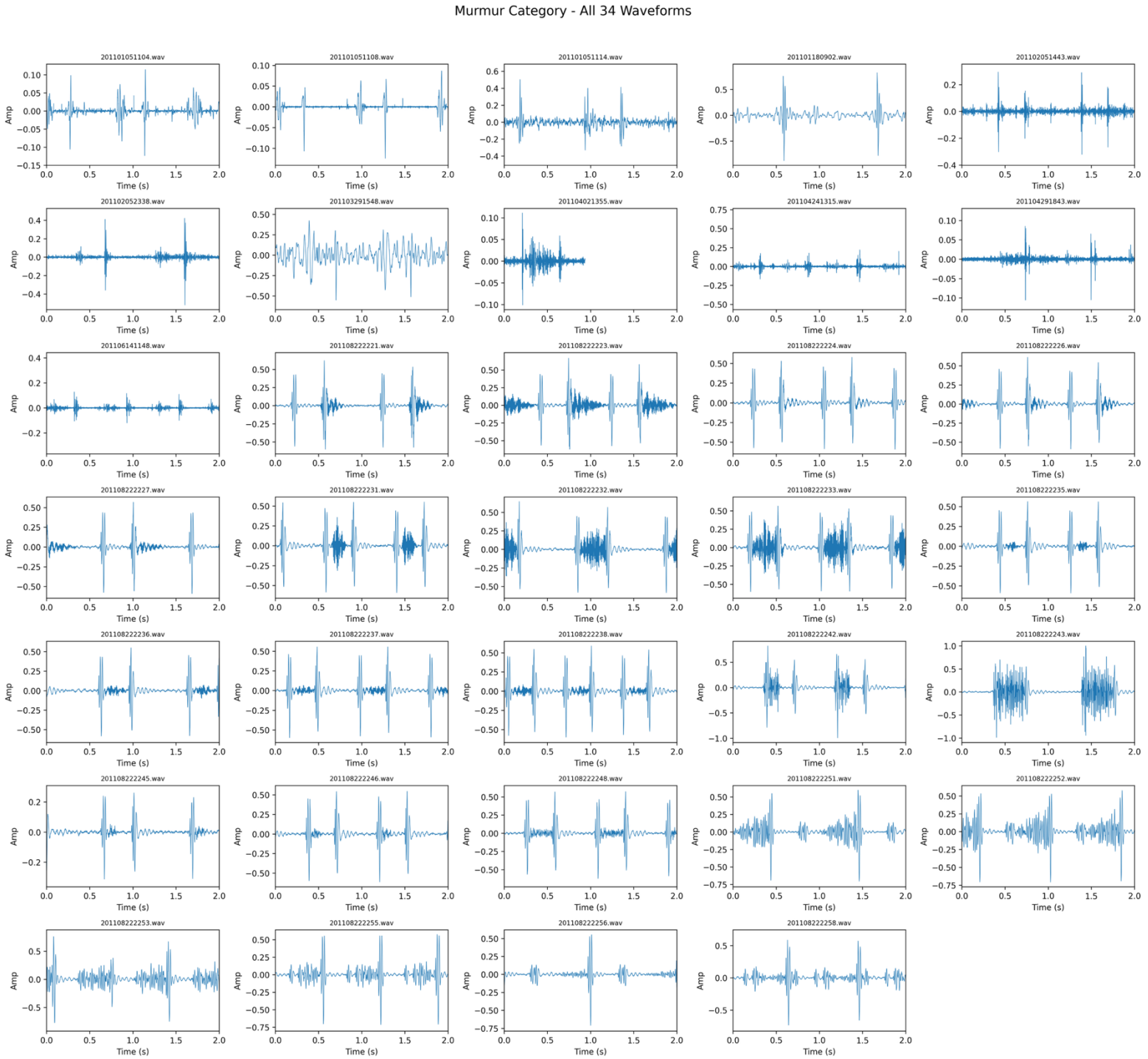
- Diagnostic gaps in early detection and treatment of stroke risk

Data Processing Pipeline: Overview



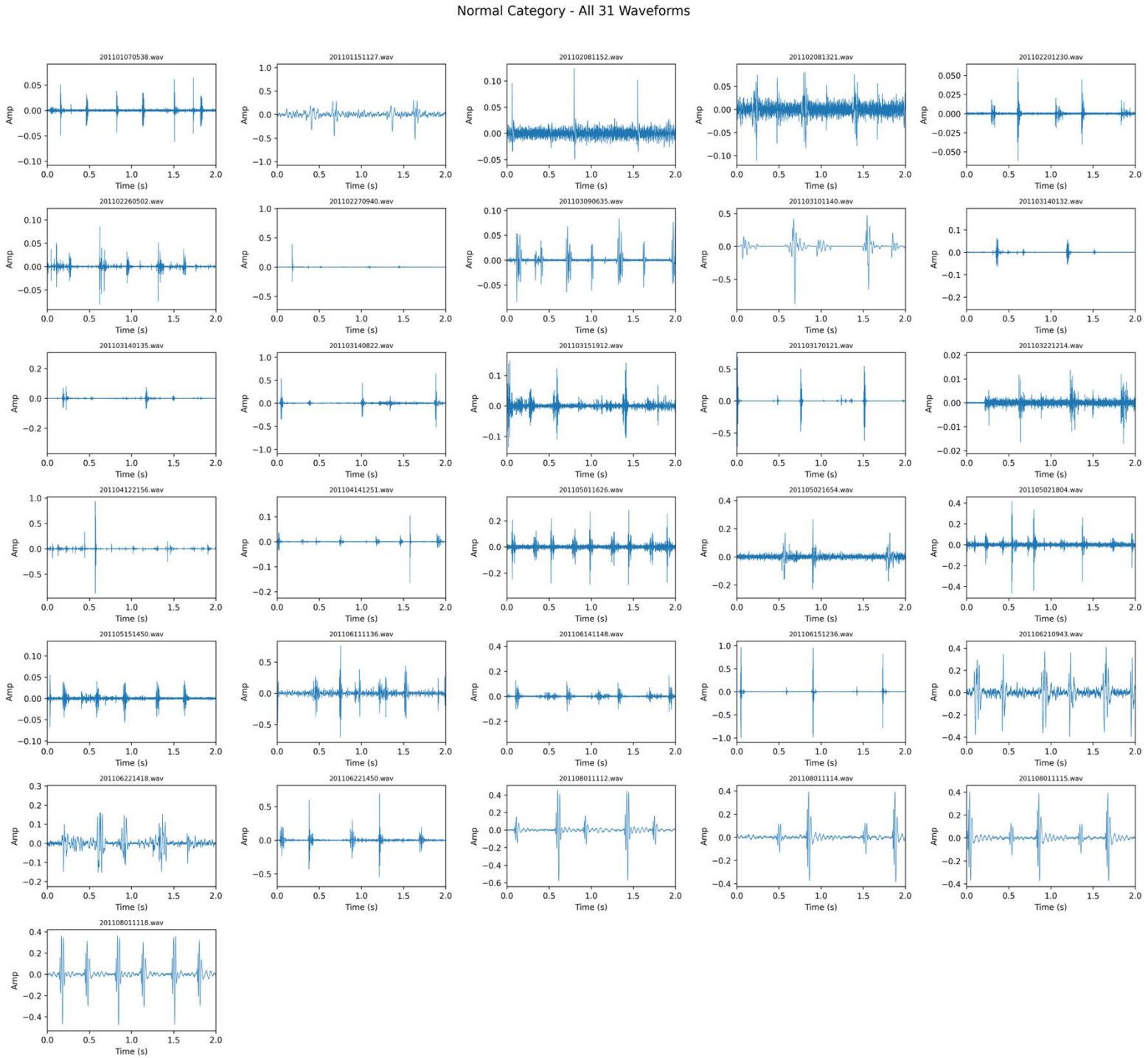
Artifact

- Raw plots (up to 2 sec)
- Just to visualize the general structure of the waveforms
- Some waveforms just have no structure
- Makes them readily distinguishable using anomaly detection algorithms



Murmur

- Raw plots (up to 2 sec)
- Just to visualize the general structure of the waveforms

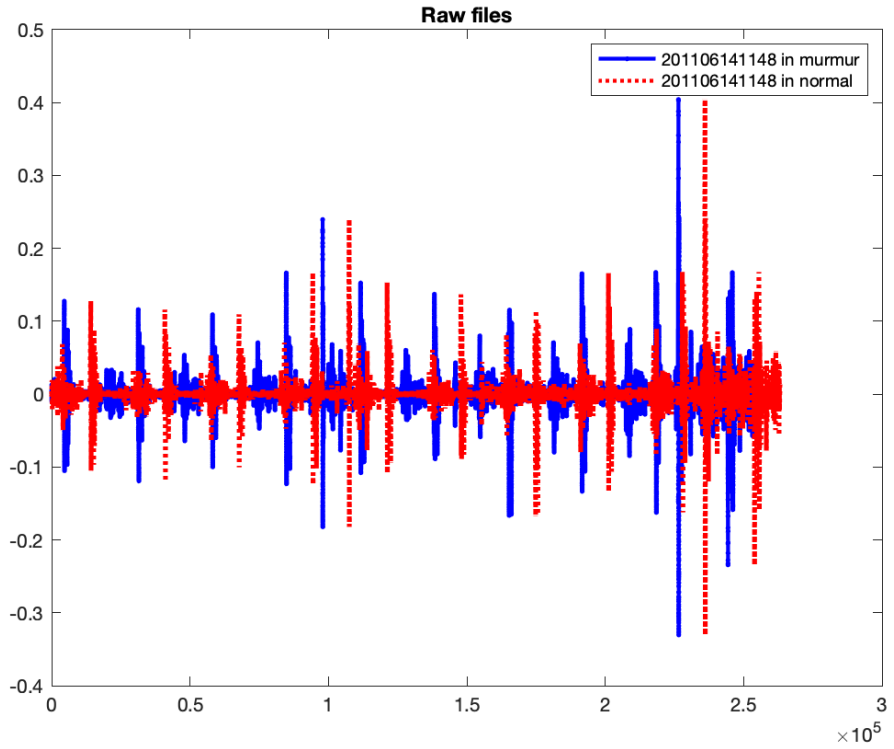


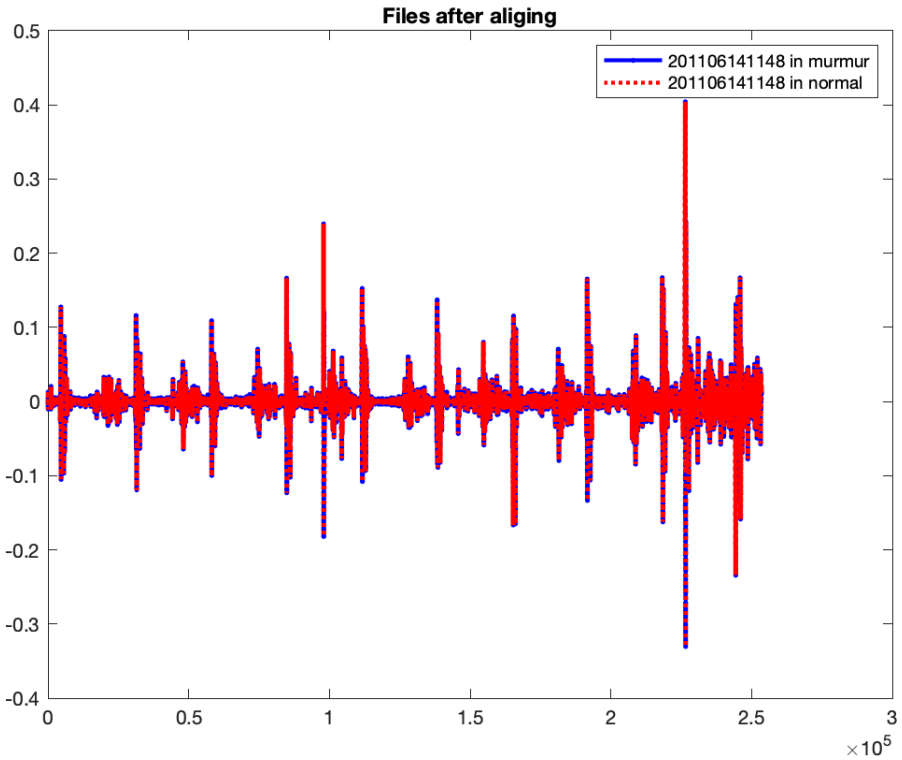
Normal

- Raw plots (up to 2 sec)
- Just to visualize the general structure of the waveforms

Quality check

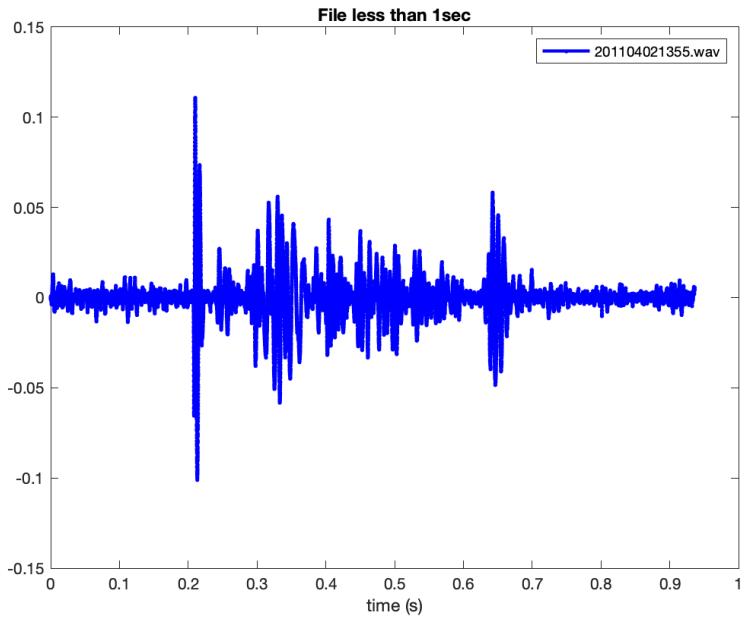
- Same filename in two different folders:
  - File 201106141148 (in normal folder) and 201106141148 (in murmur folder)
  - Closer inspection revealed one to be a delayed version of the other → Removed it from murmur (as shown below)





Quality check

- Removed file 201104021355 from murmur (recording less than 1sec)



Task 1: Classification of sounds into S1 and S2

Preprocessing:

Downsampled the signal to 1 kHz

- Applied anti-aliasing filter: Used a 4th-order Butterworth low-pass filter to remove high-frequency noise before down-sampling.
- Downsampled safely: Applied zero-phase filtering (filtfilt) to avoid signal distortion and then performed down-sampling to reduce the data size.
- Bandpass filtering: Then, designed a 4th-order Butterworth bandpass filter between 20 Hz - 200Hz and applied zero-phase filtering (filtfilt) to denoise the signal without introducing phase distortion

Task 1: Classification of sounds into S1 and S2

- Used the Logistic Regression (LR) - Hidden Semi-Markov Models (HSMM) algorithm proposed by [1]
- The algorithm performs heart sound segmentation using a combination of:
  - Feature extraction from PCG
  - LR for emission probability estimation
  - HSMMs to incorporate temporal duration
  - Extended Viterbi Algorithm to decode heart sound states over time

[1] Springer, David B., Lionel Tarassenko, and Gari D. Clifford. "Logistic regression-HSMM-based heart sound segmentation." *IEEE transactions on biomedical engineering* 63.4 (2015): 822-832.

Step 1: Preprocessing

- Downsample the waveform to 1000 Hz
- Denoise using wavelet filters
- Extract 4 features per time point
  - homomorphic envelope
  - hilbert envelope
  - wavelet envelope
  - power spectral density between 40–60 Hz

Step 2: Normalize and Resample

- Normalize each feature: subtract mean and divide by standard deviation
- Downsample the feature set to 50 Hz (1 frame every 20ms) to speed up processing

Step 3: Logistic Regression

- A logistic regression model is trained to classify at a point in time how likely it is that this is S1, S2, systole, or diastole.

- This gives observation probabilities (called emission probabilities in the paper)
  - Tells how likely a time point belongs to each state

Step 4: Duration Modeling in HSMM

- Each heart cycle is assumed to transition through:
- S1 → systole → S2 → diastole → S1...
- The HSMM adds:
  - Duration models (Gaussian) for how long each state lasts (e.g., S1 = 50 ms, systole = 300 ms).
  - So HSMM adds memory of how long each state typically lasts which follow a rhythm (vital for heartbeats)

Step 5: Decode State Sequence with Extended Viterbi Algorithm

A dynamic programming technique that finds the most likely sequence of heart states (S1, systole, S2, diastole) based on:

- What state the logistic regression thinks it is
- The likely duration of each state from HSMM
- The order: S1 → systole → S2 → diastole → repeat

Once decoded, each time frame is labeled as either S1, systole, S2, diastole

Task 1: Classification of sounds into S1 and S2

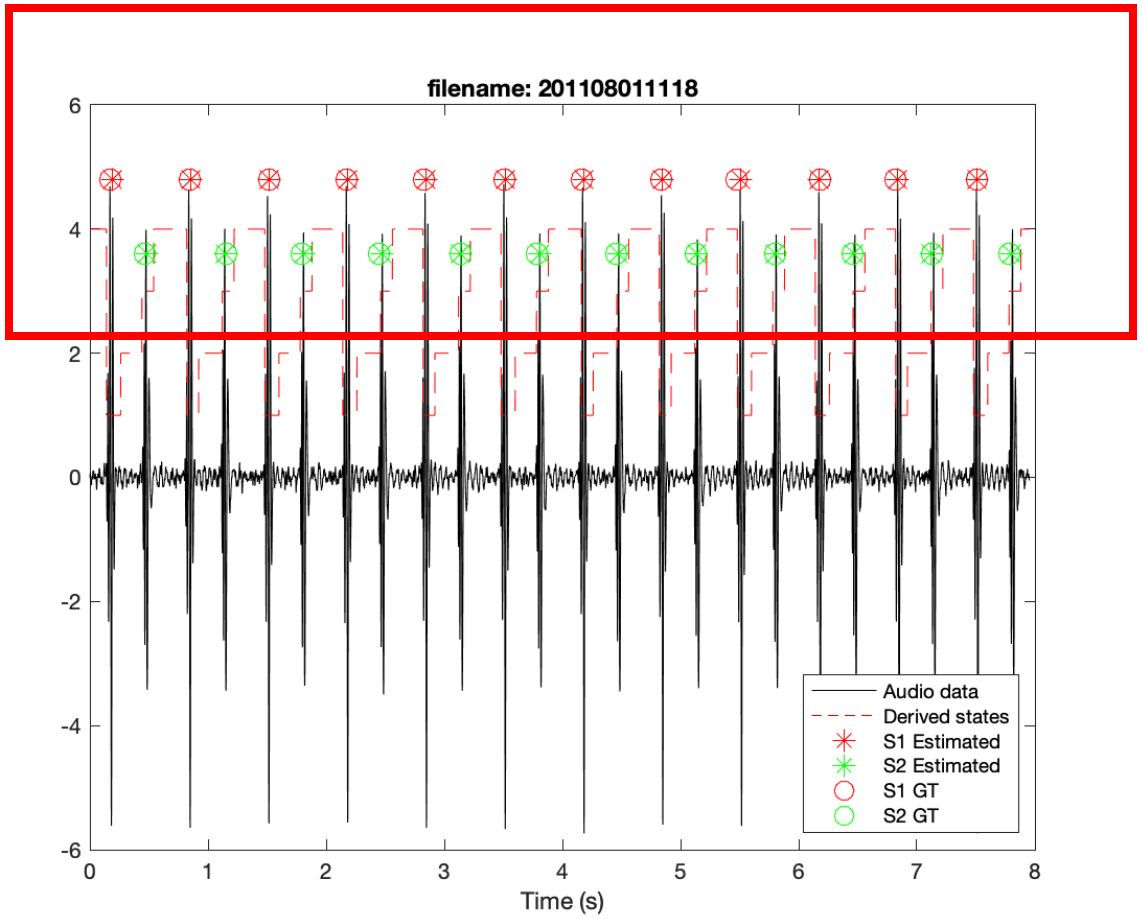
- The open-source algorithm implementation by the author (D. Springer) was downloaded from the author's website [2]
- It was adapted inside my algorithm for the prediction of S1 and S2 intervals, that
  - (Previously: Normalizes and computes the Shannon energy + smooths using a moving average + Detects peaks based on height and spacing; problem: prone to height and spacing parameters)
  - Now, I detect the maximum value in the interval identified as S1 or S2

[2] https://github.com/davidspringer/Springer-Segmentation-Code

Task 1: Classification of sounds into S1 and S2

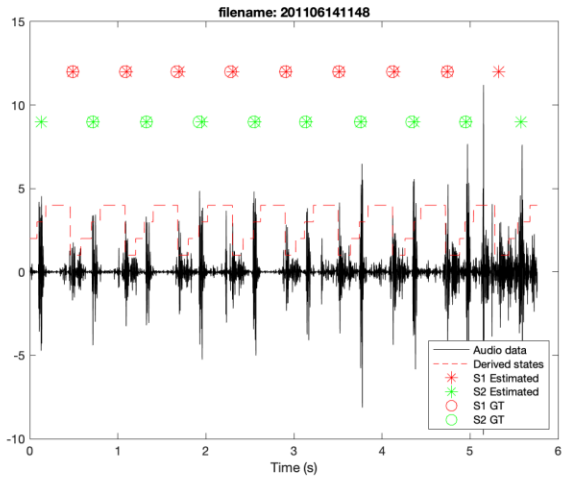
- Performance was checked on the annotated dataset given for performance
- 18/21 available files were correctly segmented from the Normal_timestamps.csv
- 85.7% accuracy rate
- Other fine-grained metrics (with segmentation tolerance) could also be computed

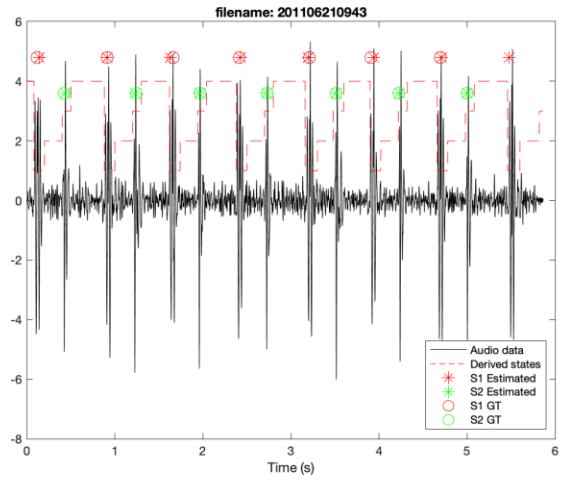
Example of correct S1 and S2 intervals identification

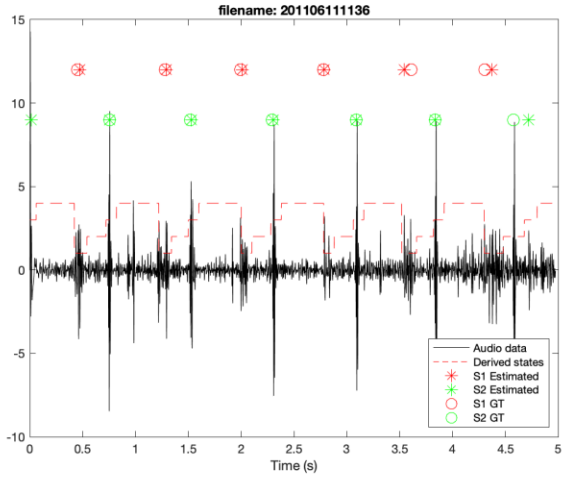


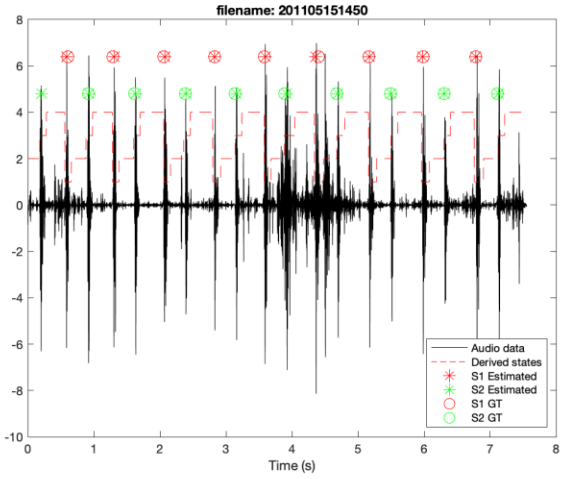
The algorithm correctly estimated the ground truth S1 and S2 location indicated by overlapping GT and estimated labels (see figure legend)

In most cases, the algorithm was able to identify S1 and S2 intervals correctly

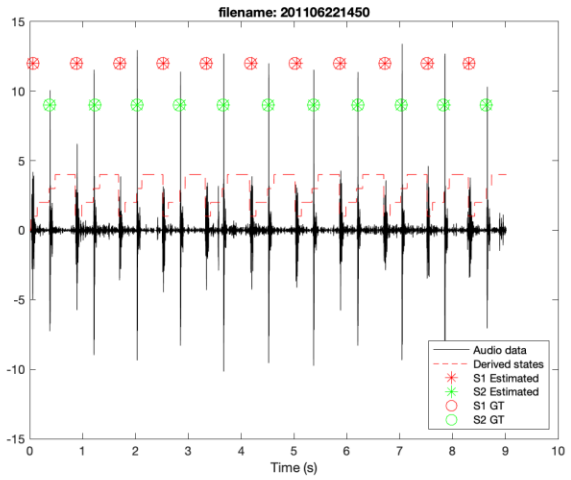


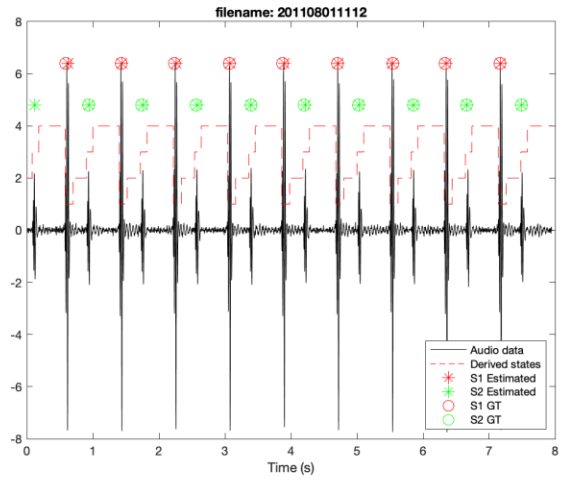


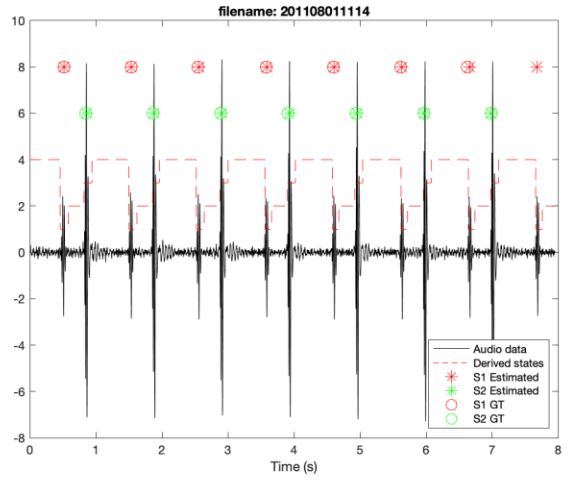


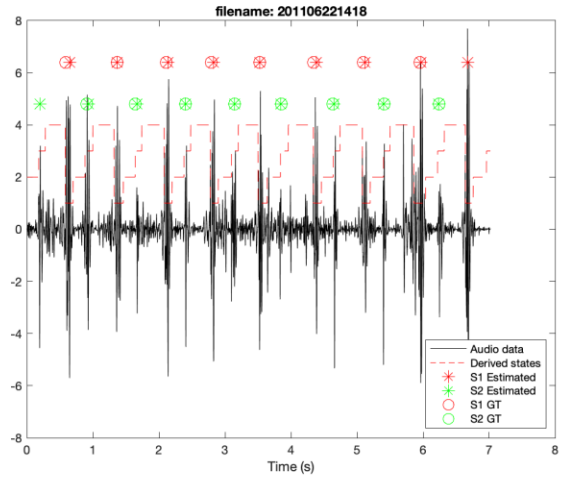


In most cases, the algorithm was able to identify S1 and S2 intervals correctly

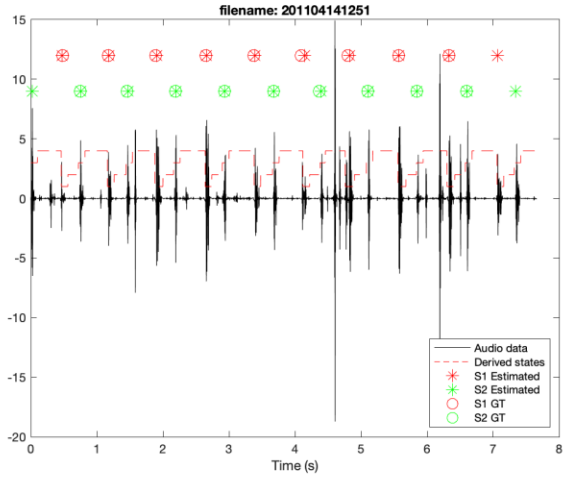


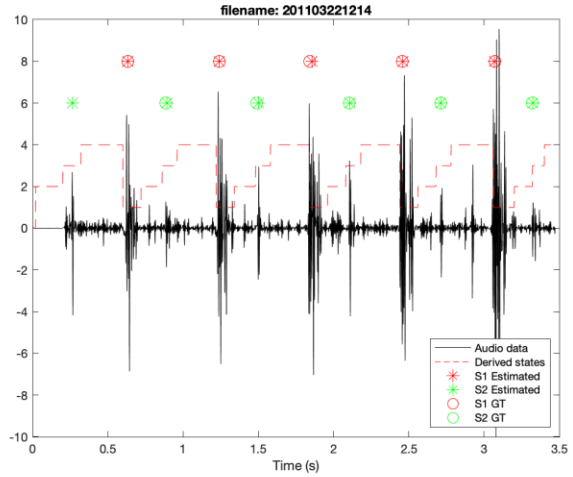


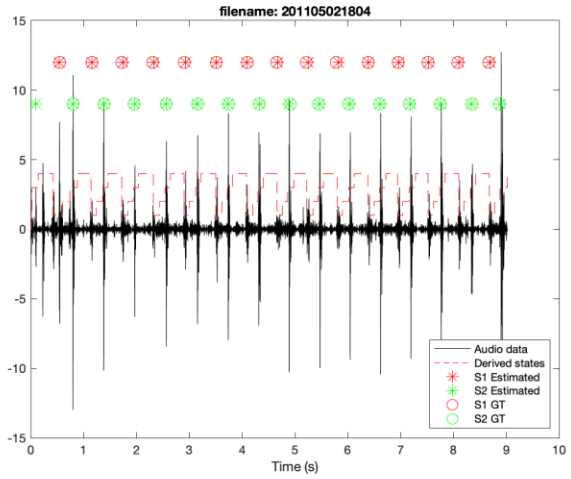


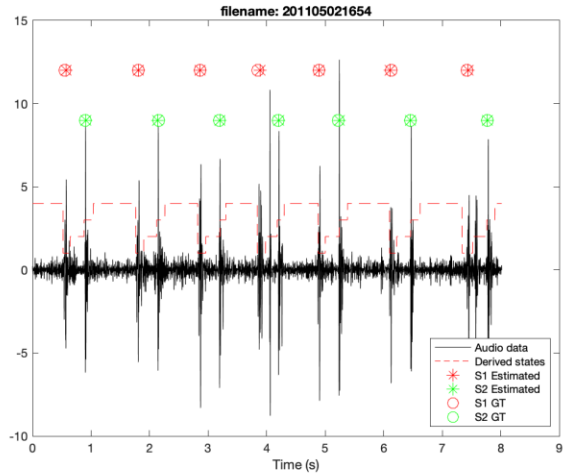


In most cases, the algorithm was able to identify S1 and S2 intervals correctly

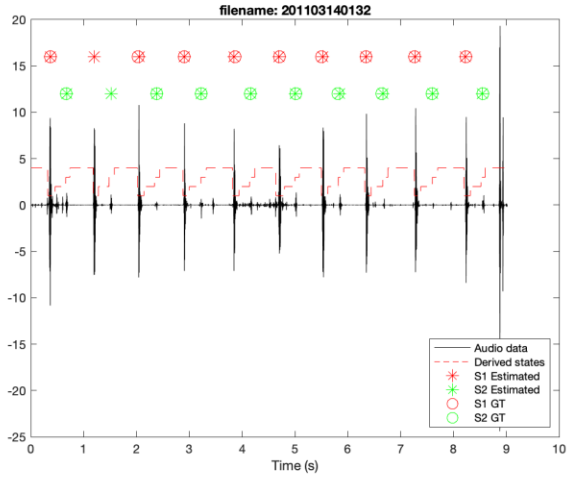


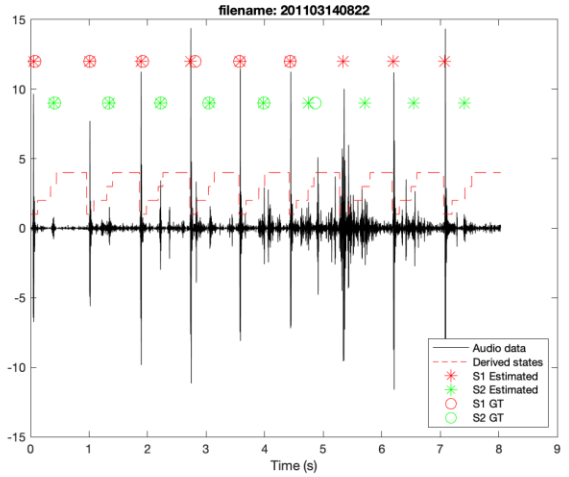


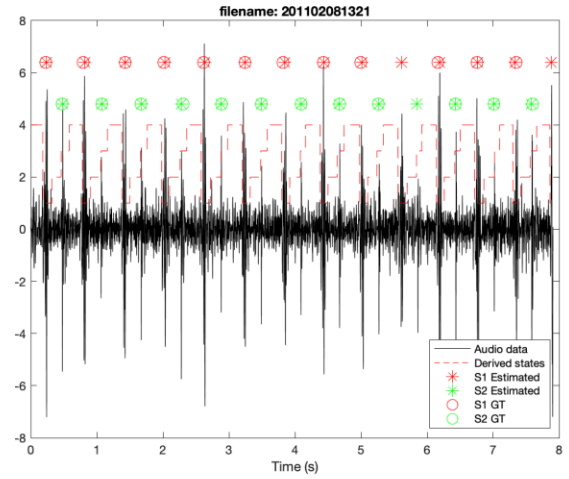


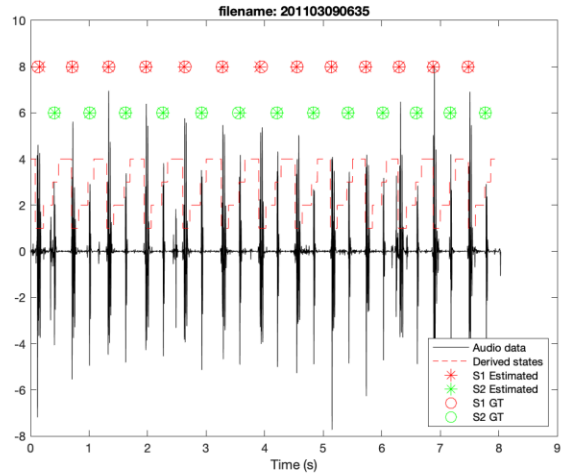


In most cases, the algorithm was able to identify S1 and S2 intervals correctly

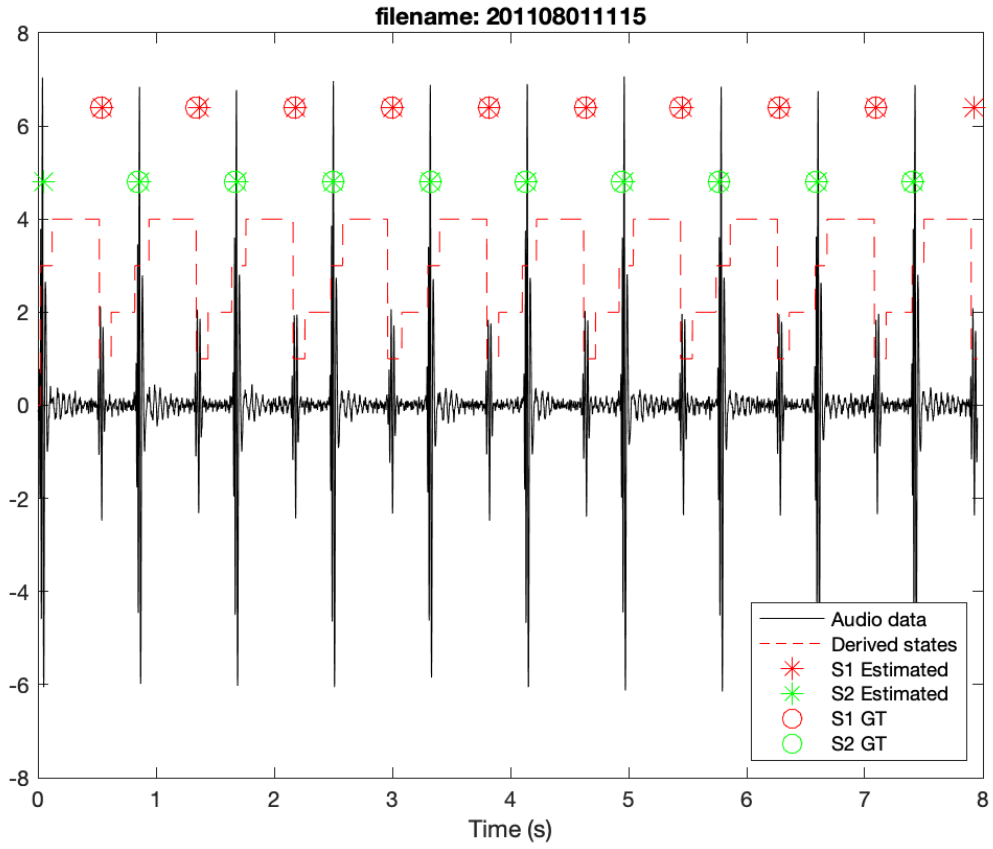




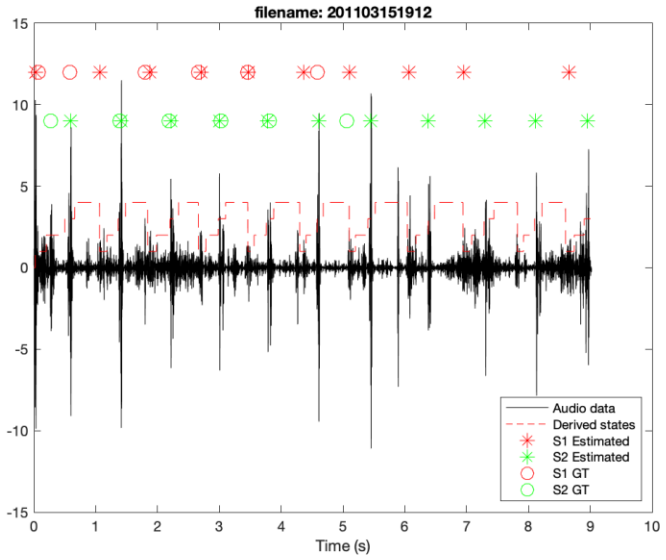


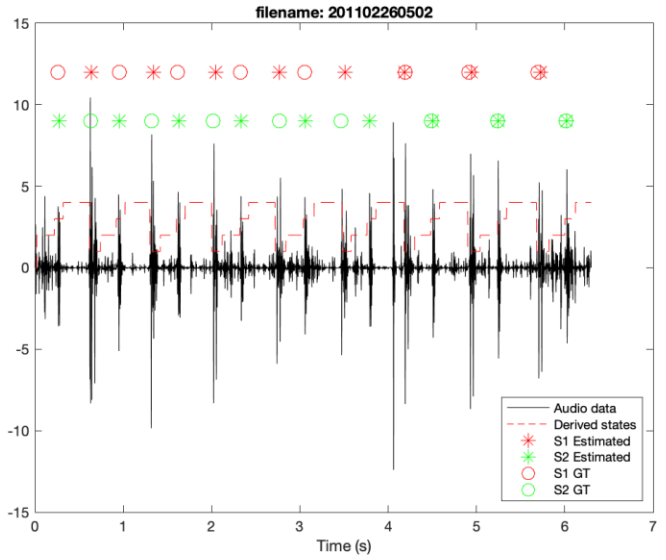


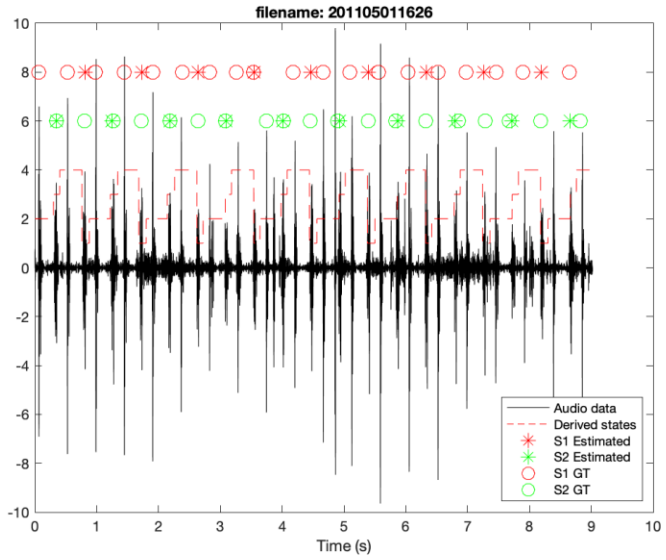
In most cases, the algorithm was able to identify S1 and S2 intervals correctly



In certain cases, there were some errors







Limitations and suggestions for improvement

- Limitations:
  - Only files for normal are given as GT; actual performance on murmur heartbeats is unknown
- Suggestions for improvement:
  - Also benchmark the algorithm with heartbeats with murmurs
  - HSMM algorithm doesn’t seem to perform as well on fast heartbeats
  - Incorporate a convolutional NN for further improvement. See for reference [3]

[3] Jang, Yeongul, et al. "Fully Convolutional Hybrid Fusion Network with Heterogeneous Representations for Identification of S1 and S2 from Phonocardiogram." *IEEE Journal of Biomedical and Health Informatics* (2024).

Traditional ML Classification

Over view of Signal Processing Pipeline

- Original signal → Anti-alias filter → Down-sample → Denoise using bandpass→ Feature extraction → Normalize to 0 mean & unit variance → Label encoding →Test-train split →Validation
- This signal was used in feature extraction, described next

Time-domain features: on S1 and S2 related intervals

- First calculated the following metrics:
  - S1 intervals = difference between successive S1 peaks
  - S2 intervals = difference between successive S2 peaks
  - mean_RR = mean(S1 intervals)
  - Systole duration = time from each S1 to the next S2 after it
  - Diastole duration = time from each S2 to the next S1 after it
- Some time domain features were inspired by [4]
- This yielded 14 time-domain features, explained next

[4] Potes, Cristhian, et al. "Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds." *2016 computing in cardiology conference (CinC)*. IEEE, 2016.

Time-domain features: on S1 and S2 related intervals

- Then calculated the following features using the metrics calculated previously

Feature	Signifies
Mean and standard deviation of S1 interval	Average & variance of S1-to-S1 intervals
Mean and standard deviation of S2 interval	Average & variance of S2-to-S2 intervals
Mean and standard deviation of systole	Average & variance of systole (S1→S2)
Mean and standard deviation of diastole	Average & variance of diastole (S2→S1)
Mean of RR = S1 to S1 length	Average heartbeat cycle length
Systole/RR	Fraction of cycle spent in systole
Diastole/RR	Fraction of cycle spent in diastole
Systole/Diastole	Balance between systole and diastole
(# of S1)/duration, (# of S2)/duration	Number of S1 or S2 per second
(# of S1)/ time of detected S1s	S1s per unit time over span of detected S1s

Freq-domain features: on S1 and S2 related intervals

- Frequency domain features were inspired by [4]
- Power spectrum of each heart sound state (S1, S2, systole, diastole) was computed using a Hamming window and DTFT
- Median power was calculated across 9 frequency bands (i.e., 25-45, 45-65, 65-85, 85-105, 105-125, 125-150, 150-200, 200-300, 300-400 Hz) for each state in every cardiac cycle
- The average of these medians across all cycles yielded 36 frequency-domain features (9 bands × 4 states)

[4] Potes, Cristhian, et al. "Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds." *2016 computing in cardiology conference (CinC)*. IEEE, 2016.

ML Training

- Split data into 80% training and 20% testing while preserving class distribution using stratify for balanced evaluation
- Evaluated multiple classification models (Logistic Regression, SVM, Random Forest, Naive Bayes, KNN) for comparison
- Performed 5-fold cross-validation on the 80% training dataset to select the best model

Various performance metrics

$$\text{Accuracy} = \frac{Correct\ Predictions}{Total\ Predictions}$$

$$\text{Precision} = \frac{True\ positives}{True\ positives + False\ Positives}$$

$$\text{Recall} = \frac{True\ positives}{True\ positives + False\ Negative}$$

$$\text{F1-score} = 2\ \frac{Precision * Recall}{Precision + Recall}$$

Macro average = Unweighted average across all the metrics (precision, recall, and F1-score)

Weighted average = Weighted (by the number of samples present) average across all the metrics (precision, recall, and F1-score)

Results of training on a single fold of 80% training, 20% testing

```
Model: Logistic Regression
Train Accuracy: 0.8902
              precision    recall  f1-score   support

    artifact       1.00      1.00      1.00         8
      murmur       0.86      0.86      0.86         7
      normal       0.83      0.83      0.83         6

    accuracy                           0.90        21
   macro avg       0.90      0.90      0.90        21
weighted avg       0.90      0.90      0.90        21

ROC-AUC: 0.9858
Cross-Validation Accuracy: 0.7926
```

```
Model: SVM
Train Accuracy: 0.8902
              precision    recall  f1-score   support

    artifact       1.00      1.00      1.00         8
      murmur       0.86      0.86      0.86         7
      normal       0.83      0.83      0.83         6

    accuracy                           0.90        21
   macro avg       0.90      0.90      0.90        21
weighted avg       0.90      0.90      0.90        21

ROC-AUC: 0.9858
Cross-Validation Accuracy: 0.8551
```

```
Model: Random Forest
Train Accuracy: 1.0000
              precision    recall  f1-score   support

    artifact       1.00      1.00      1.00         8
      murmur       0.83      0.71      0.77         7
      normal       0.71      0.83      0.77         6

    accuracy                           0.86        21
   macro avg       0.85      0.85      0.85        21
weighted avg       0.86      0.86      0.86        21

ROC-AUC: 0.9680
Cross-Validation Accuracy: 0.8559
```

```
Model: Naive Bayes
Train Accuracy: 0.9146
              precision    recall  f1-score   support

    artifact       1.00      1.00      1.00         8
      murmur       0.86      0.86      0.86         7
      normal       0.83      0.83      0.83         6

    accuracy                           0.90        21
   macro avg       0.90      0.90      0.90        21
weighted avg       0.90      0.90      0.90        21

ROC-AUC: 0.9486
Cross-Validation Accuracy: 0.8779
```

```
Model: KNN
Train Accuracy: 0.8780
              precision    recall  f1-score   support

    artifact       1.00      1.00      1.00         8
      murmur       0.86      0.86      0.86         7
      normal       0.83      0.83      0.83         6

    accuracy                           0.90        21
   macro avg       0.90      0.90      0.90        21
weighted avg       0.90      0.90      0.90        21

ROC-AUC: 0.9893
Cross-Validation Accuracy: 0.8662
```

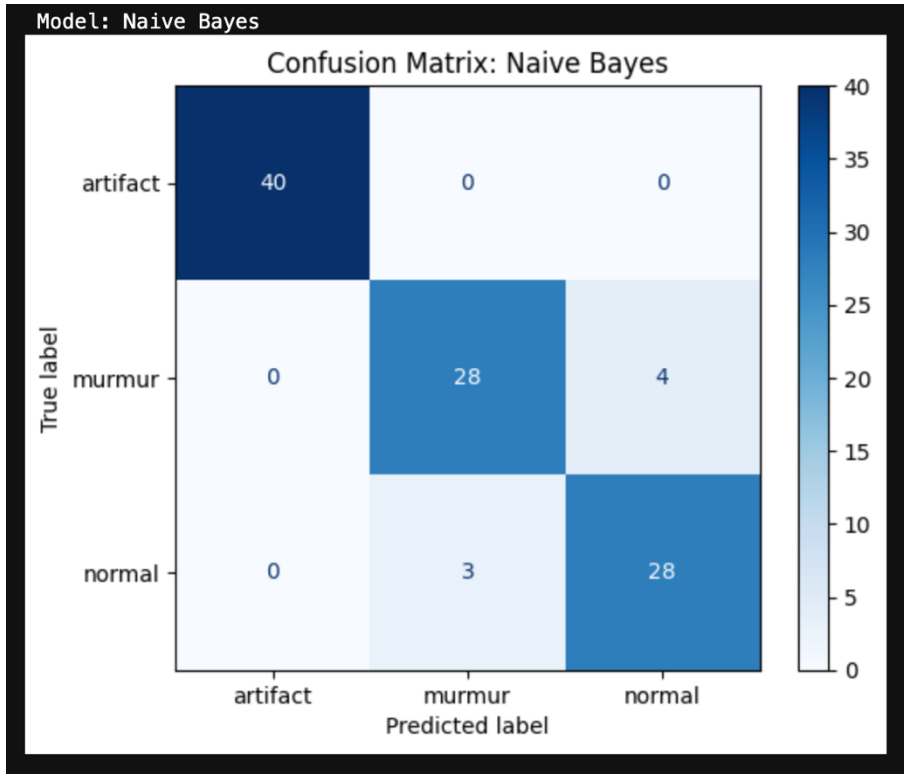
- Best model selection: Naïve Bayes (based on cross-validation score)
- Test accuracy < train accuracy →denotes model likely not over-fitting + they are close (so model also not underfitting)

K-fold on dataset: Performance and generalization

- Then performed 5-fold cross-validation on the complete dataset to assess the performance and generalization capability

```
Model: Logistic Regression
Cross-Validation Accuracy: 0.8638

Model: SVM
Cross-Validation Accuracy: 0.8543

Model: Random Forest
Cross-Validation Accuracy: 0.8638

Model: Naive Bayes
Cross-Validation Accuracy: 0.9319

Model: KNN
Cross-Validation Accuracy: 0.8552
```

- **Naïve Bayes performs best with 93.19% accuracy**



Future work

- Incorporate additional hand-crafted features (skewness, kurtosis, mel frequency features, etc.)
- Perform hyperparameter tuning
- Incorporate ensemble methods to increase robustness
- Augment deep learning architectures with hand-crafted features to improve performance and interpretability