Heart sound classification

Shahil Manoj Dhotre
Ashwini Muralidharan
Purushothaman Saravanan
Iype Eldho

1 INTRODUCTION AND BACKGROUND

1.1 Problem Statement

Cardiovascular disease has a significant impact on the world population - it leads to a high number of premature deaths in people of all age groups. According to the World Health Organization (WHO), an estimated 17.9 million people die each year from cardiovascular diseases, accounting for 31% of all deaths worldwide. Cardiovascular disease places a significant burden on healthcare systems around the world as well. The cost of treating cardiovascular disease is high, and it can strain healthcare resources, particularly in low and middle-income countries.

Owing to the several life-threatening consequences of cardiovascular diseases and the enormous impact they have on the human population, the need for automated heartbeat classification techniques is imperative. This project aims to develop a pipeline that takes raw heartbeat sounds as input, processes it and classifies it into 4 classes - Normal, Murmur, Extra Heart Sound, Artifact. After pre-processing, the signal is passed to a feature extraction engine that performs feature engineering by extracting the necessary handcrafted feature set, contributing to the novelty of the project. It is followed by a deep learning classifier that performs the prediction. The analysis thus retrieved can be used to serve the first level of screening of cardiac pathologies - both in a hospital environment by a doctor (using a digital stethoscope) and at home by the patient (using a mobile device)

1.2 Related Works

Several works are related to the pre-processing, feature extraction, and classification of heart sound signals. [1] used Hidden Semi-Markov Model to solve the problem of accurate segmentation of the heart sound in the noisy real heart sound recording. The method used in [2] extracted six audio features (spectral centroid, zero crossing rate, energy entropy, spectral roll-off, volume, and spectral flux) and selected the most distinguishing statistical features. Finally, the most prominent features are used to distinguish between normal and abnormal heart sounds. In addition, many different models have been proposed for the detection of heart diseases from heart sounds. In [3], a hidden Markov model was constructed for heart sound classification and achieved promising experimental results. In [4], the classification was done with a stacked autoencoder network and obtained very efficient and effective results.

2 METHOD

2.1 Novel Aspect(s)

As mentioned in Section 1.1, this project aims to develop a pipeline that pre-processes data and passes it to a feature extraction engine. This engine performs feature engineering, which involves extracting relevant features or characteristics from the data. These features are handcrafted, meaning they are selected or transformed based on domain-specific knowledge or expertise. The use of handcrafted features contribute to the novelty of the project, as it may involve unique or innovative approaches to identifying informative features from the data.

Further, the utilization of the Long Short-Term Memory (LSTM) model in heart sound classification using audio data introduces a sense of novelty. By leveraging LSTMs, heart sound classification models can effectively capture the unique patterns and characteristics of heart sounds in audio data, allowing for the accurate classification of different heart conditions. This novel approach to heart sound classification using LSTMs enables more precise and automated identification of heart conditions, which has the potential to revolutionize the field of cardiology by enhancing diagnosis accuracy and efficiency.

2.2 Rationale

To perform a comparative analysis, Support Vector Machine (SVM) - a popular supervised classification algorithm was chosen to serve as a baseline model. The Long Short-Term Memory (LSTM) method is expected to perform well and outperform other reasonable baseline methods, such as SVM for heartbeat classification due for several key reasons:

- Ability to capture temporal dependencies: Heartbeat sounds are sequential data that exhibit temporal dependencies, as the duration, intensity, and frequency of heart sounds change over time. LSTMs are specifically designed to handle sequential data and can effectively capture long-term dependencies, which are critical for accurately modeling heartbeat sounds. On the other hand, SVM is a type of traditional machine learning algorithm that does not inherently consider temporal dependencies, which may limit its ability to capture the dynamic nature of heartbeat sounds.
- Handling variable-length sequences: Heartbeat sounds can vary in length, depending on factors such as heart rate, respiratory rate, and patient age. LSTMs can handle variable-length sequences, as they can adaptively learn to process input data of different lengths. In contrast, SVM typically requires fixed-length input features, and handling variable-length sequences may require additional pre-processing steps, such as feature extraction or data padding, which can introduce complexities and potentially reduce classification performance.
- Automatic feature extraction: LSTMs can learn to automatically extract relevant features from raw heartbeat sound data, without the need for explicit feature engineering. This

can be advantageous, as heartbeat sounds can be complex and may contain subtle patterns or characteristics that are not easily discernible by human experts. SVM, on the other hand, typically relies on handcrafted features, which may not capture all the relevant information in heartbeat sounds, potentially limiting its classification accuracy.

- Adaptability to noisy data: Heartbeat sounds can be affected by various factors, such as ambient noise, patient movement, and recording artifacts, which can introduce noise and variability in the data. LSTMs are known to be robust to noisy data, as they can learn to model patterns even in the presence of noise. SVM, on the other hand, may be more sensitive to noise, as it relies on explicit feature engineering and may not be as adaptive to noisy data.

Hence, LSTM is a promising method for heartbeat sound classification and can potentially outperform other reasonable baseline methods, such as SVM, in this context.

2.3 Approach

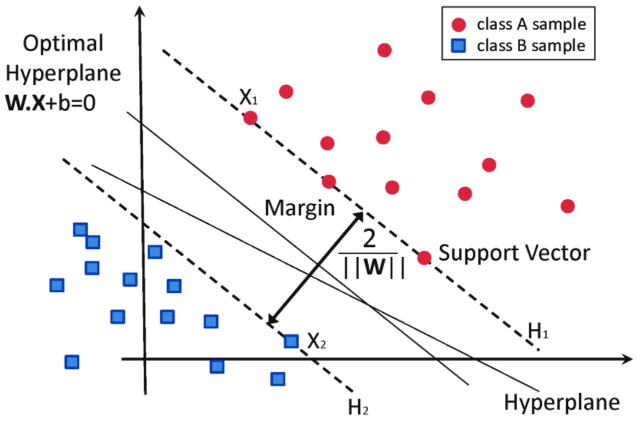

Figure 1: Schematic diagram of SVM

2.3.1 Baseline Model. SVM was established as a baseline model and trained on the dataset to perform heart sound classification. SVMs are a type of supervised machine learning algorithm used for classification and regression analysis. The SVM algorithm finds the hyperplane that maximizes the margin between the closest data points of each class, known as support vectors, as shown in 1. The margin is the distance between the hyperplane and the support vectors. SVM aims to maximize the margin because it helps to reduce the risk of misclassification.

If the data is not linearly separable, SVM uses kernel functions to map the data into a higher-dimensional space where the data can be separated by a hyperplane. The most commonly used kernel functions include linear, polynomial, and radial basis function (RBF) kernels.

SVM is a powerful algorithm that can handle complex datasets and is widely used in various applications, such as image classification, text classification, and bioinformatics. SVM was trained on the feature subset and was used to perform the prediction.

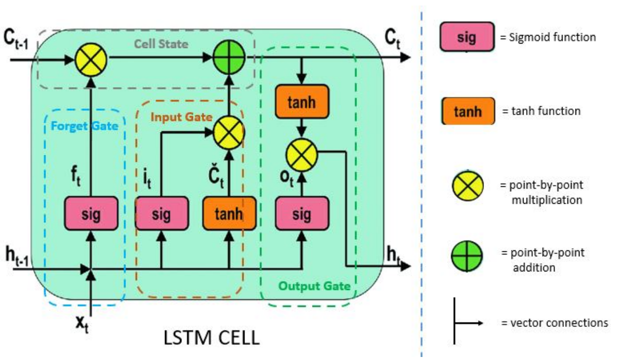

Figure 2: Schematic diagram of LSTM

2.3.2 Proposed LSTM Model. LSTM networks are a modified version of the recurrent neural networks (RNN). Generalized from a feedforward neural network, RNNs have internal memory, hence, are recurrent in nature. The network takes information from previous inputs to influence the current input and output. However, RNNs suffer from vanishing and exploding gradient problems. To combat this, LSTM networks are used. The LSTM architecture aims to provide a short-term memory for RNN that can last thousands of timesteps, thus "long short-term memory". LSTM networks are well-suited to classifying, processing, and making predictions on time series data.

The key components of an LSTM architecture are the memory cells and the gates, as shown in Fig. 2, which are responsible for controlling the flow of information during computation. LSTMs typically have three main gates: the input gate, the forget gate, and the output gate.

The memory cell in an LSTM acts like a "black box" that stores information that is passed through time steps and is updated at each time step based on the input data and the gate activations. The memory cell has the ability to retain information from earlier time steps and propagate it to later time steps, allowing the LSTM to capture long-term dependencies in the data. The input gate determines how much new information should be added to the memory cells at the current time step. It takes as input the current input data and the previous hidden state, and applies a sigmoid activation function to produce an output between 0 and 1. This output is then multiplied element-wise with a candidate activation vector, which is obtained by passing the input data and the previous hidden state through a tanh activation function. The resulting vector is added to the memory cells, allowing the LSTM to selectively update its memory based on the importance of the new information.

The forget gate controls how much information should be retained or discarded from the memory cells. It takes as input the current input data and the previous hidden state, and applies a sigmoid activation function to produce an output between 0 and 1. This output is then multiplied element-wise with the current memory cells, allowing the LSTM to selectively forget or retain information in its memory.

The output gate regulates the flow of information from the memory cells to the output of the LSTM. It takes as input the current input data and the previous hidden state, and applies a sigmoid activation function to produce an output between 0 and 1. This output is then multiplied element-wise with the current memory cells, which are passed through a tanh activation function. The resulting vector is the output of the LSTM at the current time step, which can be used for prediction or further processing.

The combination of the memory cells and gates in LSTMs enables them to capture and maintain long-term dependencies in sequential data. The gating mechanism allows LSTMs to selectively update, forget, and output information, making them well-suited for tasks such as sequence prediction, time series forecasting, and language modeling, where the ability to capture long-term dependencies is critical for accurate modeling of the data.

Overall, while both SVMs and LSTMs are effective machine learning algorithms for time series classification, LSTMs have several advantages in handling the unique characteristics of time series data, making them a popular choice for this task, the results of which are explained in Section 5.

3 PLAN AND EXPERIMENT

3.1 Dataset Description

The dataset contains labeled audio files of the heart sounds (lub dub) in the .wav format. Each audio file is of varying length between 1 second to 28 seconds [5]. We had to update the dataset, from the previous report and we added more samples into our dataset, since our LSTM model was performing rather poorly for the initial smaller dataset. Heart sounds are classified into 4 types in the DataSet

- Normal: normal, healthy heart sounds
- Murmur: whooshing, roaring, rumbling, or turbulent fluid noise captured when the heart is not working properly
- Extra Heart Sound Category: heart sound captured during early symptoms of the heart disorder sound
- Artifact: This contains noise and indicates the instructor to capture the heart sound again.

Table 1: Dataset Description

Category	No. of files	Size (MB)	Min	Max	Avg
Artifact	149	10.2	1.34	27.87	9.01
Extra Heart Sound	46	2.05	1.87	13.38	5.86
Murmur	67	3.87	0.86	24.16	7.66
Normal	149	6.67	0.76	14.68	4.35

The waveplots above in Fig. 3 are the pictographic representation of different types of heart sounds labels used in this project.

3.2 Data preprocessing

The data needs to be preprocessed so that the relevant features could be extracted from the raw data to facilitate the prediction. The *librosa* python package is used for music and audio analysis. It provides the building blocks necessary to create audio information retrieval systems [6]. The *load* command loads the audio files as a NumPy floating point time series, with the default sample rate of 22050 Hz. The following features are extracted from the NumPy array:

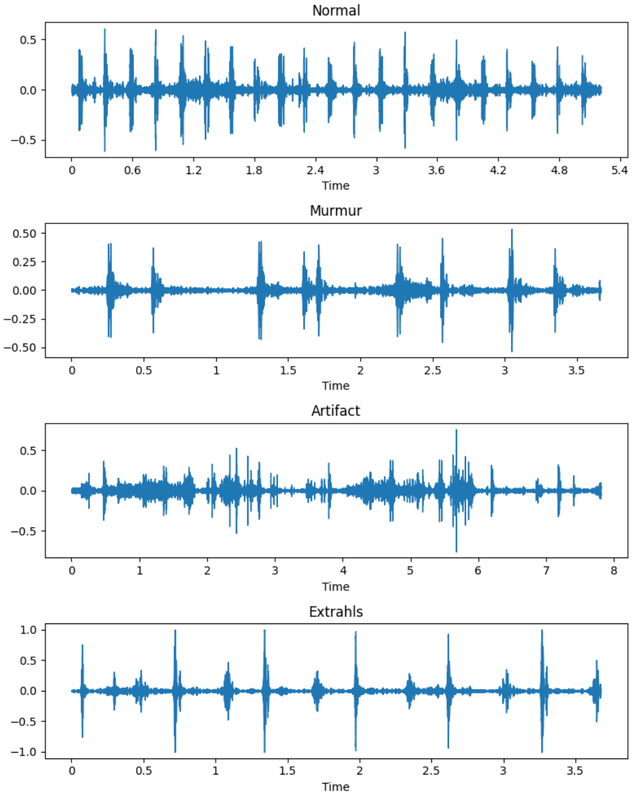

Figure 3: Waveplots of different heart sounds

1. *STFT - Short-time Fourier transform*: The STFT represents a signal in the time-frequency domain by computing discrete Fourier transforms (DFT) over short overlapping windows
2. *MFCC - Mel-frequency cepstral coefficients*: MFCCs accurately represent the envelope of the short-time power spectrum, in which the shape of the vocal tract is manifested
3. *Chroma*: Computes a chromagram from a waveform or power spectrogram
4. *Contrast*: Each frame of a spectrogram S is divided into sub-bands. For each sub-band, the energy contrast is estimated by comparing the mean energy in the top quantile (peak energy) to that of the bottom quantile (valley energy). High contrast values generally correspond to clear, narrow-band signals, while low contrast values correspond to broad-band noise
5. *Tonnetz*: Computes the tonal centroid features. This representation uses the method of [7] to project chroma features onto a 6-dimensional basis representing the perfect fifth, minor third, and major third each as two-dimensional coordinates

The STFT is not a directly used feature, but it is used to compute Chroma and Contrast. These four features are horizontally stacked into a single array. These arrays are further vertically stacked for each file for every folder containing each class, thus forming the preprocessed dataset. The labels for the preprocessed dataset are created by using a NumPy array with the following labels:

(0) Normal
(1) Murmur
(2) Artifact
(3) Extra Heart Sound

Further data pre-processing was needed for LSTM. The data was split into sequence length of 5, which yielded the final data to be of shape (312, 65, 1) which served as input to the LSTM model. The labels were One-Hot Encoded.

3.3 Hypothesis

The primary hypothesis investigated in this study is the classification of heart sounds accurately into four categories. Based on previous research and experimentation, the other hypothesis is that Long Short-Term Memory (LSTM) models will perform better than other traditional machine learning algorithms, specifically SVM for tasks involving sequential data analysis. The relevance of this problem has already been mentioned in the problem statement section. Further, classes Murmur and Normal are hard to distinguish and it is expected that the model ill perform poorly on the Murmur class.

4 EXPERIMENTAL DESIGN

Raw data were pre-processed as mentioned in Section 3.2. The pre-processed dataset was split 80:20 into train and test.

Hyperparameters are model parameters whose values are set before training. The hyperparameters of a model should be tuned because their optimal values are not known in advance. A model with different hyperparameters is a different model so it may have a lower performance. Grid search is the simplest algorithm for hyperparameter tuning. The domain of the hyperparameters is divided into a discrete grid. Then, every combination of values of this grid is used to calculate some performance metrics using cross-validation. The point of the grid that maximizes the average value in cross-validation, is the optimal combination of values for the hyperparameters. Grid search is an exhaustive algorithm that spans all the combinations, so it can actually find the best point in the domain.

Cross validation works by splitting the dataset into random groups, holding one group out as the test, and training the model on the remaining groups. This process is repeated for each group being held as the test group, then the average of the models is used for the resulting model.

Using cross validation and grid search provides a more meaningful result when compared to the original train/test split with minimal tuning. Cross validation is a very important method used to create better fitting models by training and testing on all parts of the training dataset.

The following hyperparameters were used to implement the SVM Model, with the following values for Grid Search Cross-Validation.

(1) C: *[0.01, 0.1, 1, 10, 100, 1000]*
C parameter adds a penalty for each misclassified data point.
(2) Kernel: *RBF*
Kernelized SVM computes decision boundaries in terms of similarity measures in a high-dimensional feature space without actually doing a transformation. The Radial Basis Function is used here.
(3) Gamma: *[1, 0.1, 0.01, 0.001, 0.0001, 0.00001]*
Gamma parameter of RBF controls the distance of influence of a single training point.

For the final phase of this project, LSTM was implemented. The model contained two LSTM layers and a linear layer which acts as a classifier. It is this layer that predicts the class. The details of this architecture is shown in Fig. 4.

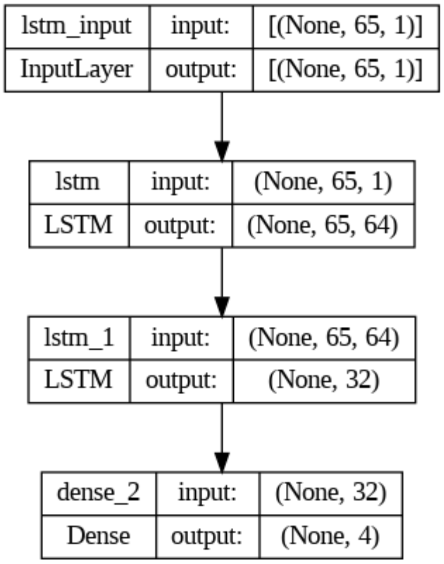
lstm_input | input: | [(None, 65, 1)]
InputLayer | output: | [(None, 65, 1)]
lstm | input: | (None, 65, 1)
LSTM | output: | (None, 65, 64)
lstm_1 | input: | (None, 65, 64)
LSTM | output: | (None, 32)
dense_2 | input: | (None, 32)
Dense | output: | (None, 4)

Figure 4: LSTM Architecture

5 RESULTS

There are several standard metrics used for evaluating the performance of classification models. The most basic and commonly used metric is accuracy, which measures the percentage of correctly predicted labels out of all the labels. However, accuracy alone may not always provide a complete picture of a model's performance. Two other commonly used metrics are precision and recall. Precision measures the proportion of true positives (correctly predicted positive labels) out of all positive predictions, and is useful in cases where false positives have significant consequences. Recall, on the other hand, measures the proportion of true positives out of all actual positives, and is useful in cases where false negatives have significant consequences. The F1 score is another commonly used metric that combines both precision and recall, providing a single value that represents the harmonic mean of the two metrics. The F1 score is useful in cases where you want to balance precision and recall, and is often used when the dataset is imbalanced. Together, accuracy, precision, recall, and F1 score provide a comprehensive overview of a classification model's performance. The Grid Search Cross Validation reveals that the SVM Model performs the best for the following values of the hyperparameters:
$C = 1000$
$Gamma = 0.00001$

The results are shown in the normalized confusion matrix in Fig. 5

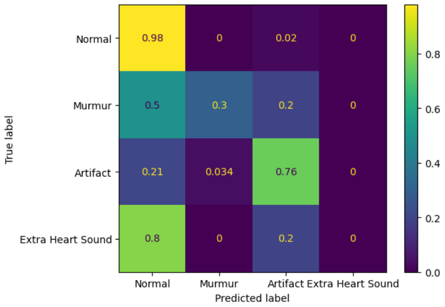

Figure 5: Confusion Matrix of SVM

Table 2: Classification report of SVM

Label	Precision	Recall	F1-Score	Support
0	0.76	0.98	0.86	49
1	0.75	0.3	0.43	10
2	0.85	0.76	0.80	29
3	0.00	0.00	0.00	5
Accuracy			0.78	93
Macro Average	0.59	0.51	0.52	93
Weighted Average	0.75	0.78	0.75	93

Table 3: Classification report of LSTM

Label	Precision	Recall	F1-Score	Support
0	0.77	0.96	0.85	48
1	0.78	0.44	0.56	16
2	1.00	0.89	0.94	9
3	0.00	0.00	0.00	5
Accuracy			0.78	78
Macro Average	0.64	0.57	0.59	78
Weighted Average	0.75	0.78	0.75	78

Similarly, the results for LSTM are tabulated in Table 3, and the confusion matrix is shown in Fig. 6. The accuracy and losses per epoch are also depicted in Fig. 7 and Fig. 8 respectively.

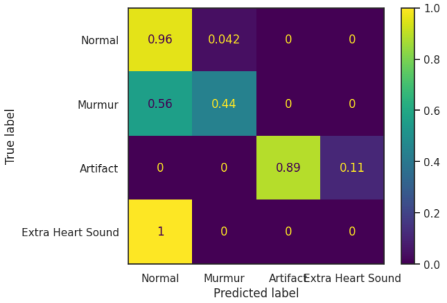

Figure 6: Confusion Matrix of LSTM

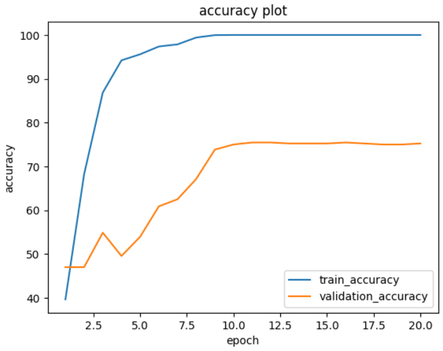

Figure 7: Accuracy for every epoch

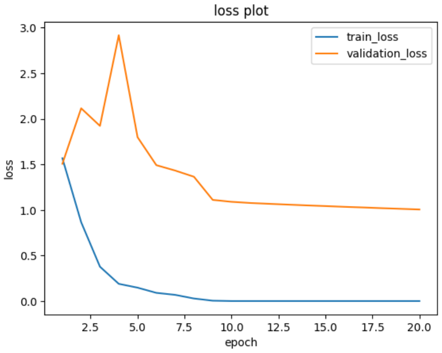

Figure 8: Loss per epoch

5.1 Discussion

As the confusion matrix from figure 5 of SVM shows, the labels are predicted accurately most of the time. From Table 2, the SVM model is the best at predicting the *Normal* label, with a 86% accuracy. This could be because *Normal* contains more samples compared to the other labels, thus it is easier to predict.

The SVM model completely fails for the*Extra Heart Sound* category. This could possibly be the case because it captures the sound when the early symptoms of the heart disorder are detected. Thus, it is confused with the *Normal* category 80% of the time.

The *Normal* and *Murmur* categories have F1-scores of 86% and 43% respectively. Since, *Murmur* is more misidentified than *Normal*, the model would have more false negatives than false positives and this is a problem since mistakenly identifying a normal sample as a disorder is better than being unable to identify a sample with heart problems, here it is the opposite.

Similarly, figure 6 shows the confusion matrix for the LSTM model. The f1-scores of *Murmur* and *Artifact* have improved, compared with the SVM model. The Macro-Average of the LSTM is also higher than the SVM, by 7%.

Just like the SVM model, LSTM completely fails for the *Extra Heart Sound* category, identifying it as the *Normal* category 100% of the time.

6 CONCLUSION

In conclusion, this project has been able to successfully develop a novel model that classifies heart sound, which would have beneficial implications in being able to detect heartbeat irregularities before the health condition of the patient worsens. The baseline model - Support Vector Machine is a traditional machine learning model performs at an accuracy of 78%, however the evaluation metric varies a lot based on which label is to be predicted.

Long Short Term Memory (LSTM) Network, which are a modified version of Recurrent Neural Network was proposed to supersede Support Vector Machines as LSTMs have the ability to capture long-term dependencies, thus it was deemed to be appropriate for handling sequential data. The LSTM model did perform fairly well as 78% accuracy - just as well as the SVM model. As the confusion matrices show, it performs better for the artifact label than the SVM model. We hypothesize that LSTM did not supersede the SVM model since the dataset for the task was very small. Moreover, the average length of each dataset was only a couple of seconds. This could be due to the following reasons.

(1) *Limited context*: LSTMs rely on capturing contextual information from previous time steps to make predictions at the current time step. Short durations of data provide limited context for the LSTM to capture long-range dependencies accurately. With insufficient context, the LSTM may struggle to model the underlying patterns in the data effectively.
(2) *Overfitting*: LSTMs are prone to overfitting, especially when the training data is limited. When the duration of data is short, the model may be at risk of overfitting, as there may not be enough diverse examples for the LSTM to learn from. Overfitting can lead to poor generalization performance, where the LSTM may not perform well on unseen data.
(3) *Lack of pattern complexity*: If the duration of the data is short, it may not contain enough complex patterns for the LSTM to learn. LSTMs are designed to capture long-range dependencies and complex temporal patterns in data. If the data duration is short and does not contain such patterns, the LSTM may not be able to fully exploit its capabilities, leading to suboptimal performance.
(4) *Hyperparameter tuning*: LSTMs have various hyperparameters that need to be tuned, such as the number of LSTM layers, the number of LSTM units, the learning rate, and others. When the duration of the data is short, it may be challenging to find the optimal hyperparameter settings, which can impact the performance of the LSTM.
(5) *Data quality and noise*: Short-duration data may be more prone to noise or inconsistencies, which can negatively impact the performance of the LSTM. If the data quality is low or contains a high level of noise, the LSTM may struggle to extract meaningful patterns from the data.

REFERENCES

[1] David B Springer, Lionel Tarassenko, and Gari D Clifford. Logistic Regression-HSMM-Based heart sound segmentation. *IEEE Trans Biomed Eng*, 63(4):822–832, September 2015.
[2] Anjali Yadav, Anushikha Singh, Malay Kishore Dutta, and Carlos M Travieso. Machine learning-based classification of cardiac diseases from PCG recorded heart sounds. *Neural Computing and Applications*, 32(24):17843–17856, December 2020.
[3] Classification of heart sound signals using autoregressive model and hidden markov model. *Journal of Medical Imaging and Health Informatics*, 7(4), 2017.
[4] Omer Deperlioglu. Heart sound classification with signal instant energy and stacked autoencoder network. *Biomedical Signal Processing and Control*, 64:102211, 2021.
[5] P. Bentley, G. Nordehn, M. Coimbra, and S. Mannor. The pascal classifying heart sounds challenge 2011 (chsc2011) results, 2011.
[6] Brian McFee et al. librosa/librosa: 0.10.0.post2, March 2023.
[7] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, AMCMM '06, page 21–26, New York, NY, USA, 2006. Association for Computing Machinery.

Shahil Manoj Dhotre, Ashwini Muralidharan, Purushothaman Saravanan, Iype Eldho„