Heart Disease Prediction from heart beat audio signals using Machine Learning and Network Analysis

Ligari D. • Alberti A. [1]

[1] *Department of Computer Engineering, Data Science, University of Pavia, Italy*
Course of Advanced Biomedical Machine Learning
Github page: https://github.com/DavideLigari01/advanced-biomedical-project
Date: June 30, 2024

Abstract — Heart disease remains one of the leading causes of mortality worldwide, making early diagnosis crucial. This study aims to predict heart diseases by analyzing heartbeat audio signals using machine learning and network analysis. We utilized a dataset from the PASCAL Classifying Heart Sounds Challenge 2011, which includes normal heart sounds, murmurs, extra heart sounds, extra systoles, and artifacts. Various preprocessing techniques such as noise reduction, resampling, and segmentation were applied to ensure data quality. Features were extracted using methods like Mel-Frequency Cepstral Coefficients (MFCC), Chroma, RMS, ZCR, and spectral features. Multiple machine learning models including LightGBM, XGBoost, CatBoost, Random Forest, and Multilayer Perceptron were trained and evaluated. The best performing model achieved high accuracy in distinguishing between different heart sound categories. This research highlights the potential of machine learning in cardiac diagnostics and provides a foundation for future advancements in the field.

Keywords ——TO BE DEFINED—

CONTENTS

1 Introduction **1**
1.1 Problem Domain . . . 1
1.2 Research Question . . . 1
1.3 Previous Research . . . 1

2 Methods **1**
2.1 Source of Data . . . 1
2.1.1 Type of sources . . . 2
2.1.2 Classes . . . 2
2.1.3 Data Distribution . . . 2
2.2 Data Preprocessing . . . 2
2.3 Feature Extraction . . . 2
2.3.1 Features Type . . . 2
2.3.2 Sampling Rate Selection . . . 3
2.3.3 Extraction Interval Selection . . . 3
2.3.4 Number of Features per Type . . . 4
2.4 Feature Selection . . . 4
2.5 Models . . . 4
2.5.1 Metrics . . . 4
2.5.2 Prevention Model . . . 4
2.5.3 Support Model . . . 4
2.5.4 Experimented Architectures . . . 4
2.6 Tools and Software . . . 4

3 Results **5**
3.1 Prevention Model . . . 5
3.1.1 Best Model Analysis . . . 5
3.2 Support Model . . . 6
3.2.1 Explainability . . . 6
3.3 Other Experiments . . . 6
3.3.1 CNNs . . . 6
3.3.2 Tiered Ensemble Model . . . 6
3.3.3 Data Augmentation . . . 6

4 Discussion **6**
4.1 Positioning in Existing Research . . . 6
4.2 Limitations . . . 6

5 Conclusion **6**
5.1 Overall Impression . . . 6
5.2 Future Work . . . 6

6 Appendix **7**

1. INTRODUCTION

1.1. Problem Domain

1.2. Research Question

1.3. Previous Research

2. METHODS

2.1. Source of Data

The dataset for this project was obtained from a Kaggle repository titled *Dangerous Heartbeat Dataset (DHD)* [1],

which in turn sources its data from the PASCAL Classifying Heart Sounds Challenge 2011 (CHSC2011) [2]. This dataset comprises audio recordings of heartbeats, categorized into different types of heart sounds. Specifically, the dataset consists of 5 types of recordings: Normal Heart Sounds, Murmur Sounds, Extra Heart Sounds, Extrasystole Sounds, and Artifacts. Data has been gathered from the general public via the iStethoscope Pro iPhone app and from a clinic trial in hospitals using the digital stethoscope DigiScope.

2.1.1. Type of sources

2.1.2. Classes

2.1.3. Data Distribution

To prepare the data several preprocessing operations were performed:

Noise Reduction: the audio data was already provided in a clipped format to minimize noise and irrelevant information.

Normalization: the audio are loaded using the *torchaudio.load()* function, which normalized the audio signals in the range [-1, 1].

Removal of Corrupted Files: corrupted files were identified and removed from the dataset to ensure data quality.

Outlier Detection and Removal: we investigated the average duration of each class and found the 'artifact' class to have a significantly larger average duration. This was due to a few long lasting audio recordings (see Figure 1). A large number of samples from the same audio might not be as informative, thereby we used IQR to detect and remove outliers.

Resampling: we evaluated two sampling rates to determine the optimal rate for heartbeat sounds and all audio files were resampled to a common frequency of 4000 Hz (see Section 2.3.2).

Segmentation: the audio data was segmented into 1-second intervals, identified as the optimal extraction interval (see Section 2.3.3), as it offered both good performance and dataset size increasing.

Hop and Window Size: the hop size determines the number of samples between successive windows, while the window size determines the number of samples considered. Each feature was extracted using the same window length and hop length facilitating a fair assessment of each feature's contribution to the classification task.

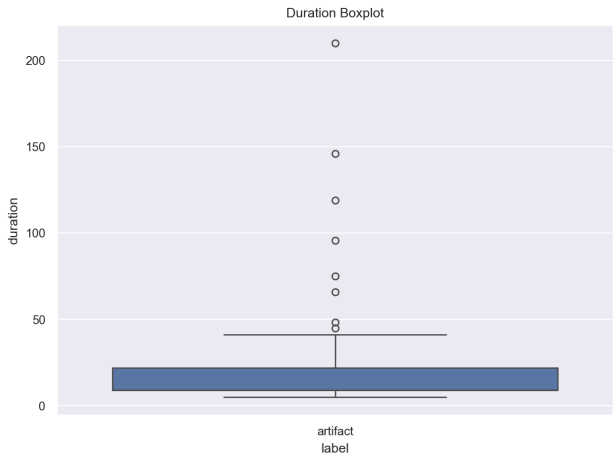

Fig. 1: Outliers in the Artifacts class.

2.2. Data Preprocessing

2.3. Feature Extraction

s demonstrated by [4] and [3], MFCCs are highly effective features for heartbeat classification. In addition to MFCCs, we incorporated other features to capture various characteristics of heart sounds, enhancing the classification accuracy. The features used are explained in the following section.

2.3.1. Features Type

MFCC
Mel-Frequency Cepstral Coefficients (MFCCs) are representations of the short-term power spectrum of sound. They are derived by taking the Fourier transform of a signal, mapping the powers of the spectrum onto the mel scale, taking the logarithm, and then performing a discrete cosine transform. MFCCs are effective in capturing the timbral texture of audio and are widely used in speech and audio processing due to their ability to represent the envelope of the time power spectrum. In heartbeat classification, MFFCs can reflect the different perceived quality of heart sounds, such as the presence of murmurs or other anomalies.

Chroma STFT
Chroma features represent the 12 different pitch classes of music (e.g., C, C#, D, etc.). They are particularly useful for capturing harmonic and melodic characteristics in music. By mapping audio signals onto the chroma scale, these features can identify pitches regardless of the octave, making them useful for analyzing harmonic content in heart sounds.

RMS
Root Mean Square (RMS) measures the magnitude of varying quantities, in this case, the amplitude of an audio signal. It is a straightforward way to compute the energy of the signal over a given time frame. RMS is useful in audio analysis for detecting volume changes and can help identify different types of heartbeats based on their energy levels. For example, in a given timeframe the RMS may be altered

by the presence of a murmur with respect to a normal heart sound.

ZCR
Zero-Crossing Rate (ZCR) is the rate at which a signal changes sign, indicating how often the signal crosses the zero amplitude line. It is particularly useful for detecting the noisiness and the temporal structure of the signal. In heartbeat classification, ZCR can help differentiate between normal and abnormal sounds by highlighting changes in signal periodicity.

CQT
Constant-Q Transform (CQT) is a time-frequency representation with a logarithmic frequency scale, making it suitable for musical applications. Since it captures more detail at lower frequencies, it may be useful for analyzing the low-frequency components of heart sounds.

Spectral Centroid
The spectral centroid indicates the center of mass of the spectrum and is often perceived as the brightness of a sound. It is calculated as the weighted mean of the frequencies present in the signal, with their magnitudes as weights. In heart sound analysis, a higher spectral centroid can indicate sharper, more pronounced sounds, while a lower centroid suggests smoother sounds.

Spectral Bandwidth
Spectral bandwidth measures the width of the spectrum around the centroid, providing an indication of the range of frequencies present. It is calculated as the square root of the variance of the spectrum. This feature helps in understanding the spread of the frequency components in the heart sounds, which can be indicative of different heart conditions.

Spectral Roll-off Spectral roll-off is the frequency below which a certain percentage of the total spectral energy lies. It is typically set at 85% and helps distinguish between harmonic and non-harmonic content. In heartbeat classification, spectral roll-off can be used to differentiate between sounds with a concentrated energy distribution and those with more dispersed energy.

2.3.2. Sampling Rate Selection

The sampling rate of the data were heterogeneous, ranging from 4000 Hz to 44100 Hz, with a majority of the data being sampled at 4000 Hz. To assess the impact of the sampling rate on the classification performance, we trained different models on different features, extracted at different sampling rates and from various intervals. Each model is then evaluated using different metrics, taking into account the class imbalance issue. We also considered a possible dependency between the sampling rate and the extraction interval, as shown in Algorithm 1.
The results, reported in Figure 2 showed no evident advantage to using a mix of sampling frequencies over a fixed resampled sample rate. Moreover, employing a fixed sample rate of 4000 Hz reduces the risk of introducing bias, enhances efficiency, and permits the use of a broader range of features and models.

Algorithm 1 Sampling rate and Interval choice

Input:
features = [`mfcc30 & 120`, `cqt30 & 70`, `chroma12`]
sampling_rates = [`mix`, `4000`]
extraction_intervals = [`0.5`, `1`, `2`, `3`]
models = [`rf`, `svm-rbf`, `lr`]
metrics = [`macrof1`, `mcc`]

for sr **in** sampling_rates **do**
  for interval **in** extraction_intervals **do**
    for feature **in** features **do**
      extract feature with interval at sr
      for model **in** models **do**
        train model with extracted feature
        for metric **in** metrics **do**
          evaluate model with metric
Output:
Given all the results, group by `model` and average the values of a specific `metric` across `features`

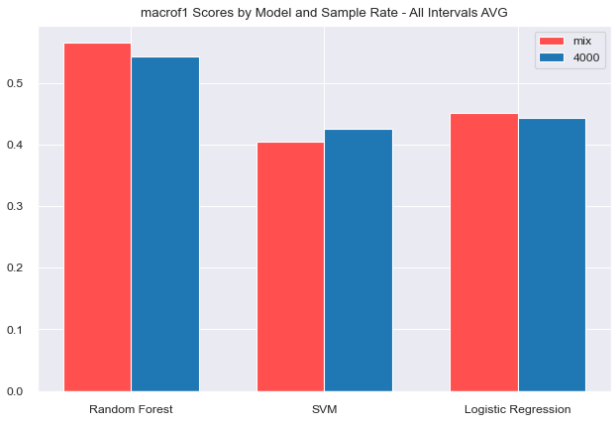

Fig. 2: Comparison of the macro F1 score for different sampling rates.

2.3.3. Extraction Interval Selection

The extraction interval refers to the duration of the audio segment from which the features are extracted. Using algorithm 1, we evaluated the performance of 0.5, 1, 2, and 3-second intervals on the classification task. It is important to note that the interval choice affects the number of samples available for training and evaluation, so in case of a limited number of samples, this choice should no be based solely on the performance of the model. The results, showed that a 2-second interval yielded the best performance, however it also reduced the number of samples, impeding a correct training and evaluation of the models. As a consequence, we picked a 1-second interval as a compromise.

2.3.4. Number of Features per Type

2.4. Feature Selection

2.5. Models

2.5.1. Metrics

2.5.2. Prevention Model

The goal of the prevention model is to provide an accessible tool for the early diagnosis of heart diseases, potentially usable by non-experts. Therefore, it is crucial to develop a model that minimizes the number of false normal predictions to accurately indicate the presence or not of disease or identify artifacts in the provided data.

To achieve this, different heart diseases were grouped together, transforming the problem into a 3-class classification task: normal, disease, and artifact. Grouping the diseases not only simplified the classification but also balanced the class distribution. The data was divided into training and testing sets in an 80-20 ratio, and various models were evaluated, as shown in Table 1.

The primary metrics for evaluating the models were the ROC-AUC score, false positive rate (FPR), and true positive rate (TPR), with F1-score and accuracy as secondary metrics. To adapt binary metrics for multi-class classification, the one-vs-rest strategy was employed. Specifically, we focused on the normal-vs-rest case to minimize false normal predictions.

In summary, each model was trained on the 3-class classification problem but was evaluated based on its binary classification performance (normal-vs-rest). The best model was selected based on its ROC-AUC score and performance at specific FPR levels (1%, 5%, 10%, and 20%). The objective was to minimize false normal predictions while maximizing true normals. A model predicting no cases as normal to achieve a 0% FPR would be ineffective.

2.5.3. Support Model

2.5.4. Experimented Architectures

The architectures of the models used in the experiments are detailed in Table 1. Special attention is given to the ensemble models, which combine predictions from multiple models to enhance overall performance.
All MLP_Ensemble models consist of the individual models listed in their architecture name. These models' predictions are combined using a soft voting strategy, where the final prediction is determined by the argmax of the sum of the predicted probabilities from each model. This approach is effective when the models are well-calibrated and exhibit complementary strengths and weaknesses.
The ALL_Ensemble model aggregates the predictions of all individual models using a majority vote strategy. In contrast, the CB_ALL_Ensemble model also considers all individual models but uses a CatBoost model to aggregate the predictions. This allows for a more flexible voting strategy, potentially leading to improved performance.

Name	Architecture (hidden layers)
Random Forest	-
XGBoost	-
CatBoost	-
LightGBM	-
MLP_Basic	(128, 64, 32)
MLP_Ultra	(512, 256, 128, 64, 32)
MLP_Large	(256, 128, 64, 32)
MLP_Small	(64, 32)
MLP_Tiny	(32, 16)
MLP_Reverse	(32, 64, 128, 256, 512, 256, 128, 64, 32)
MLP_Bottleneck	(512, 64, 32)
MLP_Rollercoaster	(512, 128, 256, 128, 256, 64, 32)
MLP_Hourglass	(512, 256, 128, 64, 32, 64, 128, 256, 512)
MLP_Pyramid	(1024, 512, 256, 128, 128, 128, 64, 32)
MLP_Wide	(1024, 1024)
MLP_WideUltra	(1024, 1024, 128, 32)
MLP_Sparse	(32, 16, 8)
MLP_Dropout	(128, 64, 32)
MLP_Ensemble1	MLP_Basic, Large, Ultra
MLP_Ensemble2	RandomForest, MLP_Ultra
MLP_Ensemble3	MLP_Rollercoaster, Large
MLP_Ensemble4	MLP_Rollercoaster, Large, Ultra
MLP_Ensemble5	RandomForest, MLP_Ultra, Rollercoaster
MLP_Ensemble6	MLP_Rollercoaster, Large, Ultra, Wide
ALL_Ensemble	All models majority vote
CB_ALL_Ensemble	All models CatBoost

Table 1: Models names and architectures.

2.6. Tools and Software

This study utilized several powerful libraries and tools for data processing, model training, and evaluation:

- **Scikit-learn**: Used for MLP, RF, and metrics such as F1, Balanced Accuracy, Accuracy, MCC, ROC, AUC, permutation importance, train-test split, confusion matrix and voting classifiers.
- **Torchaudio**: Used to load the audio, for MFCC extraction and audio resampling.
- **Librosa**: Used for other features extraction and audio processing and augmentation.
- **Imblearn**: Applied for handling imbalanced datasets with techniques such as undersampling, oversampling, and SMOTEN.
- **Numpy**: Essential for numerical computations and array manipulations.
- **Pandas**: Crucial for data manipulation and analysis.
- **Matplotlib**: Employed for creating visualizations.
- **Seaborn**: Used for statistical data visualization.
- **Scipy**: Utilized for scientific and technical computing.
- **XGBoost**: Implemented for gradient boosting models.
- **CatBoost**: Applied for gradient boosting on decision trees.
- **PyTorch**: Used for developing and training CNN models, specifically VGG16_bn.

- **TensorFlow**: Used for building and training deep learning models, including CNNs.
- **Keras**: High-level API for building and training neural networks on TensorFlow.
- **Shap**: Utilized for model interpretability.
- **Other Utility Libraries**: Includes `joblib` for model serialization, and `os`, `sys` for system operations and file handling.

3. Results

3.1. Prevention Model

The initial evaluation is presented using the ROC curves of the normal class versus the others for selected models. According to Figure 3, the MLP models outperformed the other models, as indicated by the higher AUC values. Specifically, *MLP_Ensemble5* achieved the highest AUC value of 0.96, followed by *MLP_Ultra*, *MLP_Rollercoaster*, *MLP_Ensemble2*, and *MLP_Ensemble4*, all with an AUC of 0.95.

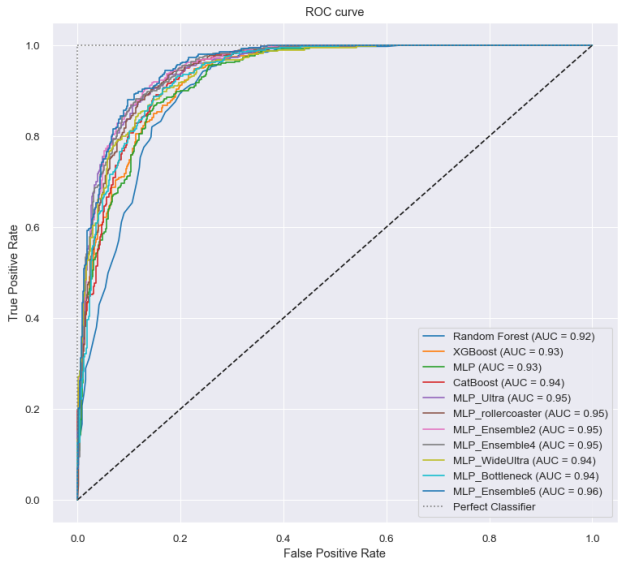

Fig. 3: ROC curves for the normal class against the rest of the classes across all models.

To further analyze model performance, we selected four significative FPR levels (1%, 5%, 10%, 20%) and calculated the corresponding TPR. The consolidated results are shown in Figure 4.

At each FPR level, *MLP_Ensemble5* outperformed the other models, achieving TPRs of 43.4%, 74.3%, 86.6%, and 95.8% at the 1%, 5%, 10%, and 20% FPR levels, respectively. Excluding *MLP_Ensemble5*, the best-performing model varied by FPR level: *MLP_WideUltra* at 1%, *MLP_Ultra* at 5%, *MLP_Ensemble2* at 10%, and *MLP_Ensemble4* at 20%.

These outcomes highlight the task's challenges in creating a model that performs well across all FPR levels and demonstrate the efficacy of a well-built ensemble model, which leverages the strengths of different models to achieve optimal performance.

3.1.1. Best Model Analysis

To understand the performance of the ensemble model (detailed architecture depicted in Figure 6), we compared its confusion matrix against the confusion matrices of the individual models that compose it (Figure 5).

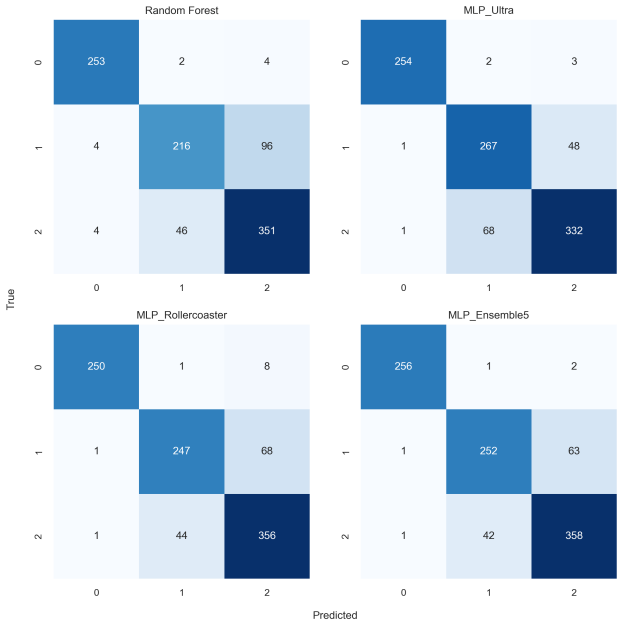

Fig. 5: Confusion matrices of the individual models and the ensemble model.

In the confusion matrix, class 2 represents normal heartbeats, class 1 represents abnormal heartbeats, and class 0 represents artifacts. The Random Forest and MLP_Ultra models exhibit complementary strengths: the Random Forest works better in recognizing normal samples, while the MLP_Ultra model performs better in identifying abnormal samples. The ensemble model effectively combines these strengths.

The contribution of the MLP_Rollercoaster model to the ensemble's performance is less apparent but has been experimentally demonstrated. This may be due to its ability to correctly classify some samples that are misclassified by the other models.

Notably, the MLP_Ultra model classifies fewer abnormal heartbeats as normal compared to the MLP_Ensemble5 model. However, this is because the MLP_Ultra model simply classifies fewer samples as normal. Minimizing the false positive rate (FPR) for the normal class by classifying all samples as abnormal would result in a very low true positive rate (TPR) for the normal class. This highlights the importance of analyzing FPR and TPR together.

In conclusion, the ensemble model is the most effective for classifying normal heartbeats, abnormal heartbeats, and artifacts, achieving an optimal trade-off between FPR and TPR.

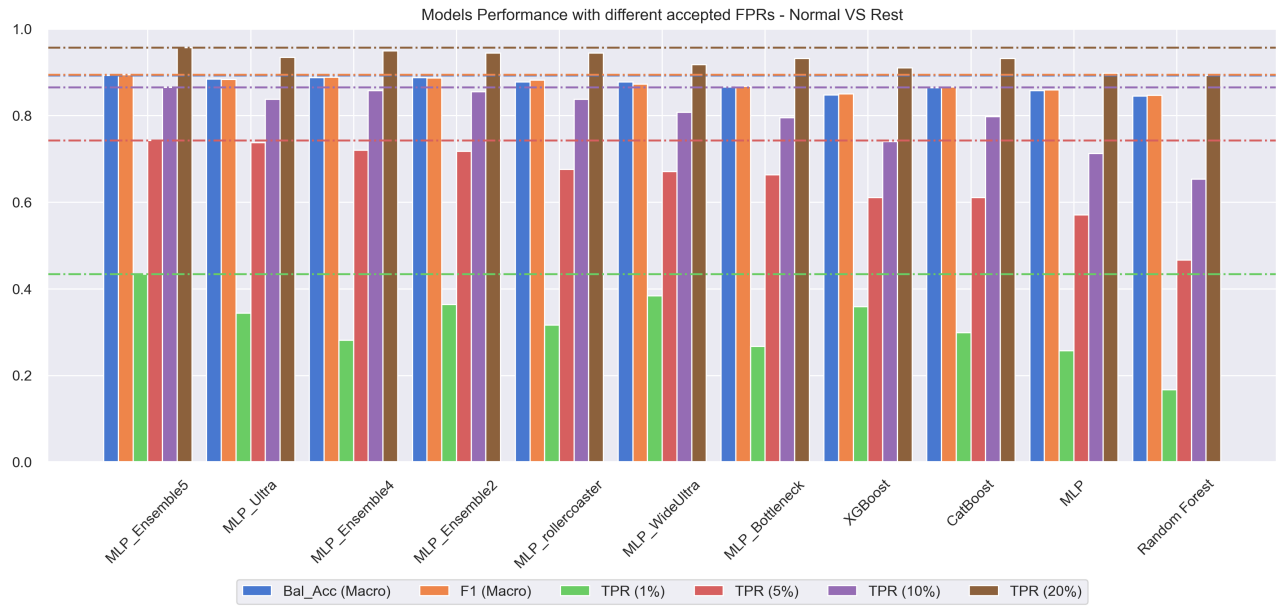

Fig. 4: TPR at different FPR levels for all models.

3.2. Support Model

3.2.1. Explainability

3.3. Other Experiments

3.3.1. CNNs

3.3.2. Tiered Ensemble Model

3.3.3. Data Augmentation

4. Discussion

4.1. Positioning in Existing Research

4.2. Limitations

5. Conclusion

5.1. Overall Impression

5.2. Future Work

6. APPENDIX

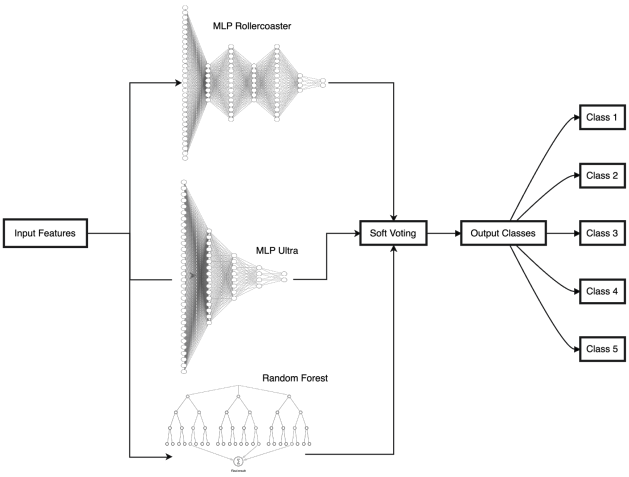

Fig. 6: MLP Ensemble5 Architecture

REFERENCES

[1] en. URL: https://www.kaggle.com/datasets/mersico/dangerous-heartbeat-dataset-dhd.

[2] P. Bentley et al. *The PASCAL Classifying Heart Sounds Challenge 2011 (CHSC2011) Results*. URL: http://www.peterjbentley.com/heartchallenge/index.html.

[3] Wei Chen et al. "Deep Learning Methods for Heart Sounds Classification: A Systematic Review". In: *Entropy* 23.6 (May 2021), p. 667. ISSN: 1099-4300. DOI: 10.3390/e23060667.

[4] Ali Raza et al. "Heartbeat Sound Signal Classification Using Deep Learning". en. In: *Sensors* 19.2121 (Jan. 2019), p. 4819. ISSN: 1424-8220. DOI: 10.3390/s19214819.