Binary Classification of Heart Sounds Using a 1D Convolutional Neural Network

Luca Lamperti
Politecnico di Milano

Abstract

In this project, I present a 1D Convolutional Neural Network (CNN) for binary classification of heart sound recordings as normal or abnormal. The raw signals undergo preprocessing steps including downsampling, normalization, and segmentation. The model is trained and evaluated on a labeled dataset, achieving promising results in classifying phonocardiogram (PCG) signals.

1 Introduction

Cardiovascular diseases are a leading cause of death globally. Phonocardiograms (PCGs), which capture the acoustic activity of the heart, can aid in the early detection of such conditions.

Heart sound auscultation is a routinely used physical examination in clinical practice to identify potential cardiac abnormalities. However, accurate interpretation of heart sounds requires specialized training and experience, which limits its generalizability. Deep learning, a subset of machine learning, involves training artificial neural networks to learn from large datasets and perform complex tasks with intricate patterns. Over the past decade, deep learning has been successfully applied to heart sound analysis, achieving remarkable results and accumulating substantial heart sound data for model training.

This project aims to automatically classify heart sounds into normal or abnormal using a 1D CNN architecture.

Convolution is a commonly used operation in deep learning, especially in CNN for tasks such as image recognition, speech recognition, and natural language processing. The convolution operation can be thought of as a special type of weighted averaging operation, which mainly works by convolving input data with a set of learnable convolutional kernels. The convolutional layer typically consists of multiple convolutional kernels, which can extract different feature information from the data.

1.1 Dataset

The 2016 PhysioNet Challenge dataset includes heart sound recordings from multiple participants from around the world, including healthy subjects and pathological patients in clinical or non-clinical environments. The challenge training set consists of five databases (a to e), totaling 3240 heart sound recordings, with durations ranging from 5 s to 120 s, and containing two labels: 'healthy' and 'unhealthy'. All recordings have been resampled to 2000 Hz and provided in .wav format.

2 Preprocessing

Preprocessing of heart sound signals is performed to remove noise, improve signal quality, and normalize the input for subsequent signal processing and analysis. Here are some methods that i used

- **Downsampling**: Downsampling refers to reducing the sampling rate of a signal to a lower frequency. Since the original heart sound signal usually has a high sampling rate and large time duration, it is computationally expensive and increases the complexity of the model, leading to slow training and overfitting issues. Therefore, downsampling can effectively reduce the time scale of the signal, making it easier to handle, while also reducing the computation and improving training efficiency and generalization ability.

- **Normalization**: Normalization refers to scaling the amplitude range of a signal to a fixed range so that different signals can be compared and processed. Normalization is very common in signal processing, especially in deep learning. Due to the presence of activation functions, the amplitude range of the signal has an important impact on the training and performance of neural networks.

  Specifically, in heart sound signal classification, normalization can help to solve the following two problems:

  - Eliminating the amplitude difference between signals from different devices or environments: Since heart sound signals collected from different devices or environments have different amplitude ranges, this can affect the processing and classification of the signals. Therefore, normalization of the signals is required to eliminate this amplitude difference so that the signals can be compared and processed.
  - Improving the stability and training effectiveness of neural networks: In neural networks, the role of the activation function is to map the input signal to a certain range. If the amplitude range of the input signal is too large, it will cause problems such as activation function saturation and gradient disappearance, which will affect the training effectiveness and stability of the neural network. Therefore, normalization of heart sound signals can help to scale the amplitude range of the signals to an appropriate range, thereby improving the training effectiveness and stability of the neural network. The normalization formula is
  $$\frac{x - \text{mean}(x)}{\text{std}(x)}.$$

- **Segmentation**: Segmentation refers to directly cutting the heart sound signal into fixed-length segments without detection. It can facilitate subsequent feature extraction and model training.

  The heart sound signal is usually a very long time series signal containing many complex cardiac events and physiological information, which may be distributed in different positions throughout the signal. If the entire signal is classified directly, it is difficult to make full use of this information, while also increasing the complexity and computational resources of the model. Therefore, dividing the signal into fixed-length segments can effectively improve the processing efficiency of the signal and the generalization ability of the model, as well as increase the number of datasets.

A heart sound signal cycle is approximately 0.8 s, but due to individual differences, it is set to 2.5 s, which ensures at least 1–2 heart cycle periods. In addition, to increase the amount of data, the overlap of the sliding window is set to 50%.

- **Filtering**: Heart sound signals are important physiological signals that have significant implications for the diagnosis and treatment of cardiovascular diseases. However, during the process of acquisition and transmission, they are often subject to various types of interference and noise, such as respiratory noise, motion noise, environmental noise, etc. These noises can affect the accuracy and reliability of the signals, making the diagnosis and treatment of cardiovascular diseases difficult.

  Therefore, before classifying heart sound signals, it is necessary to perform denoising processing on the signals. Denoising is an important step in signal processing, which aims to remove the noise component from the signal to improve the signal-to-noise ratio and accuracy. The frequency of heart sound is mainly concentrated in 20–150 Hz; to obtain more information of heart sound signal, this frequency band signal needs to be highlighted. In this study, a fourth-order Butterworth bandpass digital filter with a passband of 25 Hz to 400 Hz was used for filtering as explained in [**?**].

- Tools/libraries used 'kagglehub', ‘librosa‘, ‘jax‘, 'scipy', 'tensorflow', 'keras', 'matplotlib', 'seaborn', 'sklearn'.

3 Model Architecture

The first layer of the network is a one-dimensional convolutional layer with 16 filters of size 5 and ReLU activation. This layer scans the signal with sliding kernels, extracting short-term temporal features such as local oscillations or sharp transitions that may correspond to physiological events like heartbeats or murmurs. By applying multiple filters in parallel, the model learns different types of local patterns from the raw input.

Immediately after, a max pooling operation with pool size 2 reduces the dimensionality of the extracted feature maps. This step not only decreases the computational cost but also makes the learned features more invariant to small temporal shifts in the signal, which is desirable when dealing with biological signals that can exhibit natural variability.

The second convolutional block consists of another 1D convolutional layer with 32 filters, again of size 5, followed by a second max pooling layer. This deeper convolutional layer captures more abstract and higher-level features by combining the local patterns identified in the first stage, while the pooling step further compresses the representation and emphasizes the most salient characteristics of the signal.

After the convolutional and pooling stages, the multi-dimensional feature maps are flattened into a one-dimensional vector. This transformation allows the transition from localized feature extraction to global reasoning about the entire signal. The flattened representation is then passed to a fully connected dense layer with 64 units and ReLU activation. This dense layer integrates the previously extracted features, enabling the model to learn complex interactions and combinations that are indicative of normal versus abnormal heart sounds.

The final layer of the architecture is a dense output neuron with a sigmoid activation function. This choice is appropriate for binary classification tasks, as it produces a probability score between 0 and 1 that represents the likelihood of the input signal belonging to the abnormal class. A decision threshold is then applied to map this probability into a discrete class label.

3.1 Custom Loss Function and Accuracy Metric

The model is trained using a custom loss function and evaluation metric, specifically designed for **grouped binary classification**. This setup assumes that multiple training samples share a common identifier (`input_id`), and the model should make predictions at the group level rather than per individual sample.

4 Grouped Binary Crossentropy Loss for Segmented Audio Inputs

4.1 Function Description

We define a custom loss function called `custom_grouped_bce`, specifically designed for datasets where each original audio sample is segmented into fixed-length chunks before being passed through a convolutional neural network (CNN).

4.2 Motivation and Advantages

In many audio classification tasks, input signals have variable lengths. To handle this, we segment each audio signal into fixed-length windows, which are individually processed by the CNN. However, training directly on segment-level labels is suboptimal because:

- All segments derived from the same original signal share the same ground-truth label.
- Segment-level predictions are noisy and may not individually reflect the correct class.
- Segmenting introduces a many-to-one mapping: multiple segments correspond to a single original data point.

Therefore, instead of computing the loss at the segment level, I adopt a **grouped prediction strategy**. The idea is to:

1. Predict a value for each segment independently.
2. Aggregate predictions at the original audio level using the mean.
3. Compare this aggregated prediction to the true label of the original audio file.

This approach provides several benefits:

- **Improved robustness**: Aggregating predictions reduces the variance caused by noisy or ambiguous segments.
- **Correct supervision scale**: The model is penalized based on the prediction quality at the audio level, which aligns with the labeling granularity.
- **Better generalization**: Encourages the model to learn patterns consistent across multiple segments, rather than overfitting to individual segment artifacts.
- **Flexibility**: Easily integrates into mini-batch training where each batch may contain segments from multiple original inputs.

By aligning the loss computation with the granularity of the true labels, the `custom_grouped_bce` loss function ensures that training remains faithful to the original structure of the dataset. This is especially important in domains like audio, where segmenting data is often a technical necessity rather than a semantic one.

5 Training and Evaluation

5.1 Training Setup

The model was trained using the Stochastic Gradient Descent (SGD) optimizer, which offers precise control over the optimization dynamics. After experimenting with various optimizers, SGD produced the most stable and interpretable results for this task, especially when paired with an appropriate learning rate and batch configuration.

During training, audio recordings were first segmented into fixed-length chunks to accommodate input constraints of the CNN. However, loss computation was performed at the original audio level by aggregating predictions from all segments belonging to the same audio file. This was implemented using a custom loss function (see Section 3.1) that groups segment predictions and computes binary cross-entropy against the true label of the full recording.

To ensure that each batch contained meaningful and label-consistent data, a custom data generator was used to create batches by grouping segments by their corresponding original audio ID. I used a **batch size of 1** in terms of original input audio, meaning that each batch includes all segments of a single audio file. This choice was found to be optimal, yielding the most stable convergence and best validation results.

5.2 Evaluation Metrics

The model was evaluated using a set of standard classification metrics, computed **after aggregating predictions at the audio level**. Specifically, the segment-level predictions were averaged per input audio, and a threshold of 0.5 was applied to obtain the final binary prediction.

The following metrics were computed:

- **Accuracy**: Measures the overall proportion of correctly classified audio files.
- **AUC (Area Under ROC Curve)**: Captures the model's ability to discriminate between classes across all classification thresholds, especially relevant for imbalanced data.
- **Precision**: Indicates the proportion of predicted positive cases that are actually positive.
- **Recall (Sensitivity)**: Measures the model's ability to correctly identify true positives (i.e., diseased cases).
- **F1-Score**: Harmonic mean of precision and recall, balancing both aspects in a single metric.

These metrics are particularly important in the context of medical audio classification. In this domain, **recall** is critical to minimize false negatives (i.e., failing to identify a pathological heart sound), while **precision** helps to reduce false alarms. The **F1-score** offers a balanced assessment when both types of errors are costly. Additionally, the **AUC** score provides a threshold-independent view of performance, useful in clinical scenarios where decision thresholds might vary.

6 Results

In this section, we present and compare the results obtained from four different experimental configurations:

1. **No Filter**
2. **No Filter + Noise**
3. **Filter**
4. **Filter + Noise**

6.1 Quantitative Evaluation

Figure 1 shows the comparison of standard evaluation metrics across the four configurations. The metrics include accuracy, AUC, precision, recall, and F1-score. The model trained with **Filter** (without added noise) consistently achieved the best overall performance, with the highest accuracy and balanced precision/recall, resulting in the strongest F1-score. In contrast, the **No Filter + Noise** configuration produced the lowest results across all metrics, indicating that the model is less robust to noisy input when no filtering is applied in the preprocessing phase.

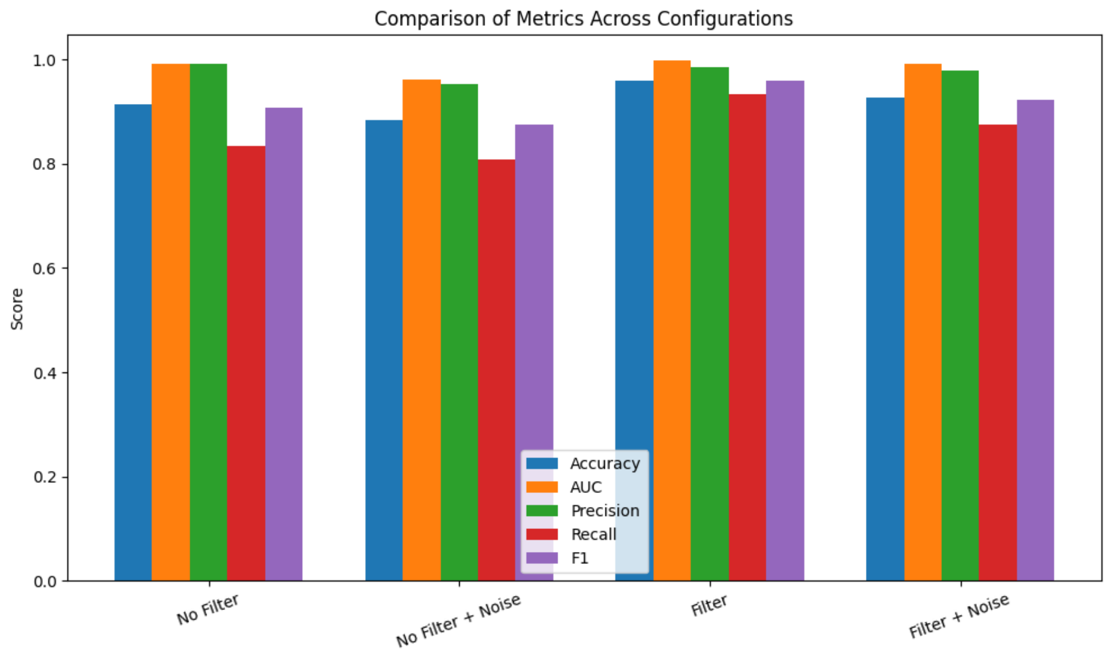

Figure 1: Comparison of metrics across configurations.

6.2 Training and Validation Loss

Figures 2 and 3 present the training and validation loss curves, respectively. The training loss decreased smoothly across all configurations, showing that the models were able to learn meaningful representations of the input signals. However, the validation curves highlight important differences:

- **No Filter + Noise** exhibited the highest and most unstable validation loss, suggesting difficulties in generalization.
- **Filter** showed the most stable and lowest validation loss, further confirming it as the best-performing configuration.
- **Filter + Noise** improved over the noisy unfiltered case but still displayed fluctuations, reflecting reduced robustness.

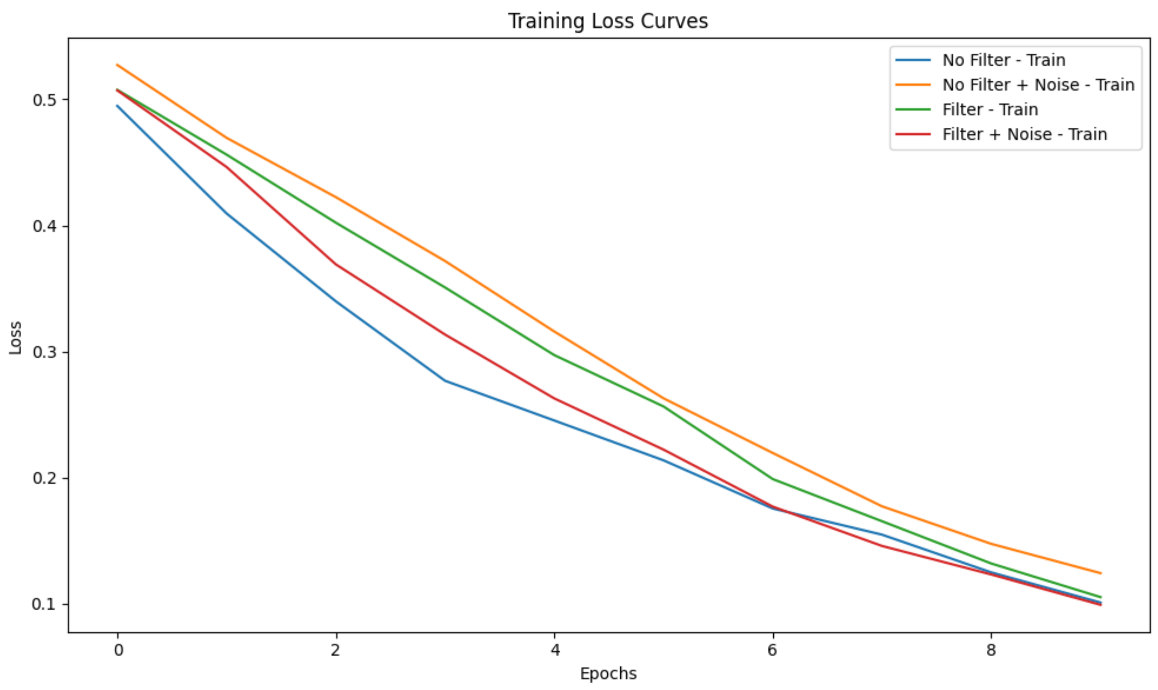

Figure 2: Training loss curves across different configurations.

6.3 Best Model Performance: Filter without Noise

The configuration that achieved the best overall performance was the **Filter** model without added noise. Table 1 summarizes the aggregated evaluation metrics, while Table 2 provides the detailed classification report.

Metric	**Score**
Accuracy	0.9734
AUC	0.9993
Precision	1.0000
Recall	0.9470
F1-score	0.9728

Table 1: Aggregated performance metrics for the best model (Filter, no noise).

These results highlight that the model achieved perfect precision for the *Unhealthy* class, meaning that all abnormal cases were correctly identified without false positives. The recall for this

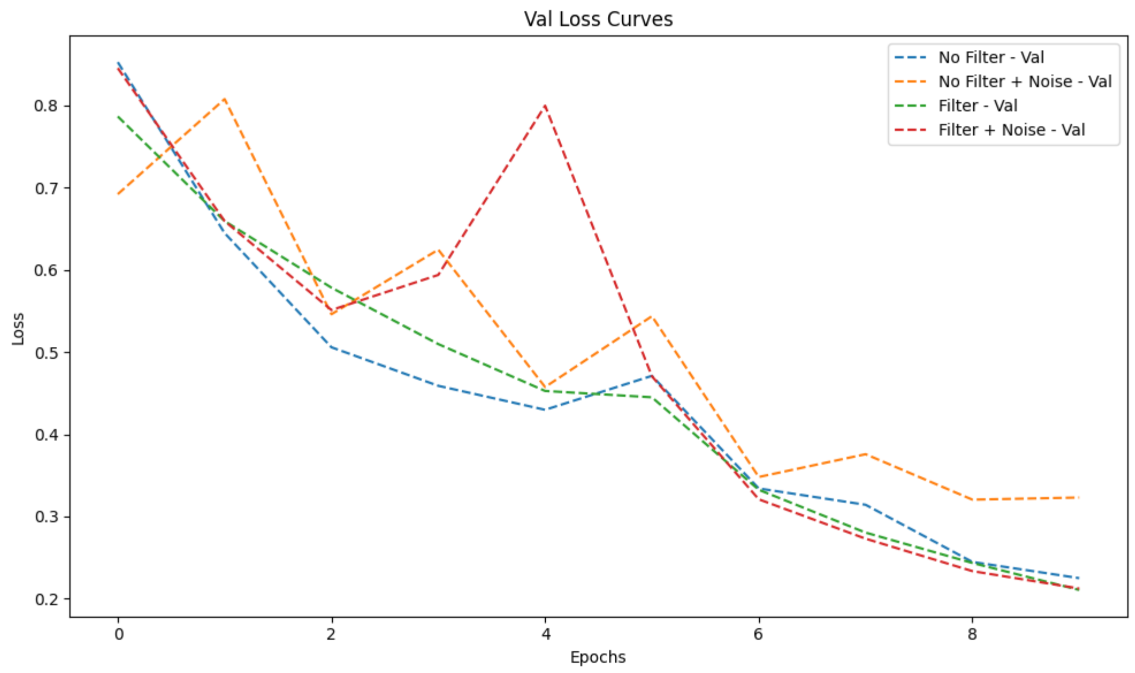

Figure 3: Validation loss curves across different configurations.

Class	Precision	Recall	F1-score	Support
Healthy	0.95	1.00	0.97	150
Unhealthy	1.00	0.95	0.97	151

Table 2: Classification report for the best model (Filter, no noise).

class was slightly lower (0.95), indicating that a small fraction of abnormal samples were missed. Conversely, the *Healthy* class achieved perfect recall (1.00) but slightly lower precision (0.95). This trade-off between sensitivity and specificity is typical in medical signal classification tasks, but the overall F1-score (0.97) demonstrates strong and reliable performance.

6.4 Effect of Noise Injection

To further investigate robustness, we applied artificial white noise to the signals using the function `add_noise`. This experiment showed that models trained without filtering suffered the most when noise was added, confirming their sensitivity to input corruption. On the other hand, applying a filtering step before segmentation made the model more robust, as reflected by the improved metrics in the **Filter + Noise** configuration.

6.5 Discussion

Overall, the results demonstrate that:

- Preprocessing plays a crucial role in classification performance.
- Filtering improves both accuracy and robustness, especially in noisy conditions.

- Artificial noise significantly degrades model performance, but its effect can be partially mitigated by appropriate preprocessing.

Among all tested models, the **Filter** configuration without noise yielded the best balance between generalization and stability, making it the optimal choice for further evaluation.

References

[1] Zhao Q, Geng S, Wang B, Sun Y, Nie W, Bai B, Yu C, Zhang F, Tang G, Zhang D, Zhou Y, Liu J, Hong S. Deep Learning in Heart Sound Analysis: From Techniques to Clinical Applications. Health Data Sci. 2024 Oct 9;4:0182. doi: 10.34133/hds.0182. PMID: 39387057; PMCID: PMC11461928.

[2] Cheng J, Sun K. Heart Sound Classification Network Based on Convolution and Transformer. Sensors (Basel). 2023 Sep 29;23(19):8168. doi: 10.3390/s23198168. PMID: 37836998; PMCID: PMC10575162.

[3] Chen W, Sun Q, Chen X, Xie G, Wu H, Xu C. Deep Learning Methods for Heart Sounds Classification: A Systematic Review. Entropy (Basel). 2021 May 26;23(6):667. doi: 10.3390/e23060667. PMID: 34073201; PMCID: PMC8229456.