






# Leveraging Liquid Time-Constant Neural Networks for ECG Classification: A Focus on Pre-Processing Techniques

Lisa-Maria Beneke<sup>1</sup>, Michell Boerger<sup>1</sup> <sup>a</sup>, Philipp Lämmel<sup>1</sup> <sup>b</sup>, Helene Knof<sup>1</sup> <sup>c</sup>,  
Andrei Aleksandrov<sup>1</sup> <sup>d</sup> and Nikolay Tcholtchev<sup>1,2</sup> <sup>e</sup>

<sup>1</sup>Fraunhofer Institute for Open Communication Systems (FOKUS), Berlin, Germany

<sup>2</sup>RheinMain University of Applied Sciences, Wiesbaden, Germany  
fi

**Keywords:** Liquid Time-Constant Neural Networks, LTC, RNN, LSTM, PTB-XL, Time Series Analysis.

**Abstract:** Neural networks have become pivotal in timeseries classification due to their ability to capture complex temporal relationships. This paper presents an evaluation of Liquid Time-Constant Neural Networks (LTCs), a novel approach inspired by recurrent neural networks (RNNs) that introduces a unique mechanism to adaptively manage temporal dynamics through time-constant parameters. Specifically, we explore the applicability and effectiveness of LTC in the context of classifying myocardial infarctions in electrocardiogram data by benchmarking the performance of LTCs against RNN and LSTM models utilizing the PTB-XL dataset. Moreover, our study focuses on analyzing the impact of various pre-processing methods, including baseline wander removal, Fourier transformation, Butterworth filtering, and a novel x-scaling method, on the efficacy of these models. The findings provide insights into the strengths and limitations of LTCs, enhancing the understanding of their applicability in multivariate time series classification tasks.

## 1 INTRODUCTION


In the field of time series classification, neural networks have emerged as powerful tools due to their proficiency in capturing complex temporal relationships. The ability to accurately classify temporal data can lead to significant advancements in predictive analytics and decision-making processes. Traditional models, such as multilayer perceptrons (MLPs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), Graph Neural Networks (GNNs), and Long Short-Term Memory networks (LSTMs) have been studied for timeseries classification tasks and have shown promising results. (Mohammadi Foumani et al., 2024)


More recently, Liquid Time-Constant Neural Networks (LTCs) have emerged as a novel approach inspired by the principles of RNNs. LTCs introduce a unique mechanism to adaptively manage temporal dynamics through their time-constant parameters. This


adaptability allows LTCs to model complex temporal dependencies effectively. (Hasani et al., 2020)


However, the application of LTCs to diverse use cases remains limited, necessitating a thorough evaluation of their performance against established models like RNNs and LSTMs. In this paper, we aim to evaluate LTCs on a multivariate time series benchmarking dataset from the medical domain. Specifically, we want to study their ability to classify myocardial infarctions (MI) in electrocardiogram (ECG) data compared to RNN and LSTM models. Additionally, we want to systematically compare the effectiveness of various pre-processing methods for this use case. Besides applying common pre-processing methods such as baseline wander removal, Fourier transformation and Butterworth filtering, we propose a new x-scaling method for the task of classifying MI. By doing so, we seek to provide insights into the strengths and limitations of LTCs, contributing to the broader understanding of their applicability in time series classification tasks. In summary, our key contributions in this paper are as follows:


- **Performance Benchmarking:** Comparing the performance of Liquid Time-Constant Neural Networks against RNNs and LSTMs for myocar-

<sup>a</sup>  <https://orcid.org/0000-0002-5741-9043>

<sup>b</sup>  <https://orcid.org/0000-0002-4411-0557>

<sup>c</sup>  <https://orcid.org/0009-0007-1364-6782>

<sup>d</sup>  <https://orcid.org/0000-0002-4717-4206>

<sup>e</sup>  <https://orcid.org/0000-0001-6821-4417>

dial infarction classification using the PTB-XL dataset.

- **Analysis of Pre-processing Techniques:** Examination of the effects of various signal pre-processing methods, including baseline wander removal, Fourier transformation, Butterworth filtering, and a novel x-scaling technique on models' performance on classifying temporal relations.
- **Insights on LTC Applicability:** Exploring the applicability of LTCs in a real-world use case while identifying the strengths and limitations of LTCs in multivariate time series classification, contributing valuable insights for future research and practical applications in medical diagnostics.

The reminder of this paper is structured as follows: Section 2 provides a background on LTCs and related models, followed by a detailed description of the experimental setup in Section 3. Section 4 analyzes the impact of various pre-processing methods on model performance, while Section 5 discusses hyperparameter optimization and presents the final MI detection models. Finally, Section 6 concludes with insights from the study and future work directions.

## 2 BACKGROUND AND STATE OF THE ART

This section provides an overview of Liquid Time-Constant Neural Networks (LTCs) and their relationship with Neural Ordinary Differential Equations (Neural ODEs) and Continuous-Time Recurrent Neural Networks (CT-RNNs). We will outline the fundamental principles of these models and their distinctive characteristics. In addition, we will present the latest related work on the utilisation of ML-based approaches for MI detection.

### 2.1 Neural Ordinary Differential Equations

The use of ordinary differential equations (ODEs) in modeling dynamic systems is motivated by their ability to capture continuous-time behavior and complex temporal relationships. Neural ODEs extend this concept by incorporating neural networks into the framework, allowing for flexible and expressive modeling of system dynamics. Given a neural network function  $f$  that is parameterized by  $\theta$  (i.e. weights and biases), the state  $x(t)$  of the modelled system at time  $t$  is defined by the Ordinary Differential Equation

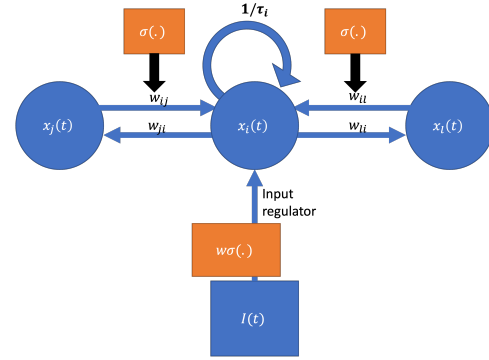


Figure 1: Illustration of a cell  $i \in [1, \dots, m]$  of an LTC with weights  $w_{ij}, w_{ji}, w_{il}, w_{li}$  contained in  $\Theta$ , and connected neurons  $l, j \in [1, \dots, m]$ ,  $i \neq l$  and  $i \neq j$

$$\frac{dx(t)}{dt} = f(x(t), t, \theta)$$

where  $I(t)$  denotes external inputs to the system (Chen et al., 2018).

### 2.2 Continuous-Time Recurrent Neural Networks

CT-RNNs also model temporal dynamics using differential equations. The state update is given by:

$$\frac{dx(t)}{dt} = -\frac{x(t)}{\tau} + f(x(t), t, \theta)$$

The key difference between CT-RNNs and Neural ODEs lies in the inclusion of the term  $-\frac{x(t)}{\tau}$ , which stabilizes the system and helps it reach an equilibrium state with a specific time constant  $\tau$ . (Funahashi and Nakamura, 1993)

### 2.3 Liquid Time-Constant Neural Networks

Hasani et al. (Hasani et al., 2020) propose yet another formula in which the neural network not only determines the derivative of the state  $x$  but also serves as an input-dependent varying time-constant:

$$\frac{dx(t)}{dt} = -\frac{x(t)}{\tau} + f(x(t), t, \theta) (A - x(t))$$

with  $\theta$  and  $A$  being parameters. As described above, CT-RNNs calculate an ODE with a time constant  $\tau_i \in \mathbb{R}, i \in \mathbb{N}$  for every  $i$ -th unit in the neural network. Contrary to that, LTCs are utilising varying (i.e. liquid) time-constants coupled to their hidden state. This improves flexibility and adaption of the network, especially on time-series prediction tasks

Table 1: PTB-XL based related work results.

Literature	Year	Classifiers	Accuracy	AUC
Smigiel et al. (Śmigiel et al., 2021)	2021	CNN	72.0%	-
		SincNet	73.0%	-
Smigiel et al. (Śmigiel et al., 2021)	2021	Neural Networks	76.2%	-
Pa'lczyński et al. (Pałczyński et al., 2022)	2022	Neural Networks	80.2%	-
Prabhakararao et al. (Prabhakararao and Dandapat, 2022)	2022	DMSCE	84.5%	-
Anand et al. (Anand et al., 2022)	2022	CNN	93.4%	-
Knof et al. (Knof et al., 2024a)	2024	CNN	96.21%	98.91%
Strodthoff et al. (Strodthoff et al., 2021)	2021	LSTM	-	90.7%
		xresnet1d	-	92.5%
		inception1d	-	92.5%

(Hasani et al., 2020). The general structure of a single unit  $i \in [1, \dots, m]$  is outlined in Figure 1.

Compared to existing algorithms, LTCs promise to better capture the causality in data patterns over time, while being much more robust and expressive. They demonstrate stable and bounded behavior, offer enhanced expressivity among neural ODEs (i.e., time-constant neural networks), and result in better performance on time series prediction tasks (Hasani et al., 2020).

## 2.4 Related Work

In the following, we will discuss related ML-based approaches for the detection of myocardial infarctions based on multivariate ECG timeseries data, which are summarized in Table 1.

The most dominant method of solving MI detection of 12-lead ECG data utilises Convolutional Neural Networks to classify between MI and non-MI classes. (Acharya et al., 2017) (Gharaibeh and Quwaider, 2022) (Atiea and Adel, 2022) Furthermore Segura-Saldana et al. (Segura-Saldana et al., 2022), Xiong et. al (Xiong et al., 2022), Joloudari et al. (Joloudari et al., 2022) and Lynn et al. (Lynn et al., 2019) showed that Gated Recurring Units and LSTMs are used for MI and non-MI classification as well. Many approaches like (Singh et al., 2018) and (Muhuri et al., 2020) utilise a multi-layer LSTM model for accurately classifying time-series into multiple classes. Since LTCs are based on recurrent networks, like LSTM, these results give good insights into the expected performance.

Hammad et al. (Hammad et al., 2022) compared multiple models for MI detection, which a relevant amount of is based on the PTB-XL dataset, described in Section 3.1. As it can be seen in Table 1, Anand et al. (Anand et al., 2022) achieved results of up to 93.4% classification accuracy.

Like others, Rai et al. (Rai and Chatterjee, 2022) additionally utilised a combination of multiple models, for example Convolutional Neural Networks and

LSTM to further improve the results. They were achieving detection accuracies on a combination of multiple datasets of up to 99.89%.

Another study by Knof et al. (Knof et al., 2024a) developed a CNN model to detect indications of myocardial infarction based on ECG data. Further, they aimed to explain and evaluate the model's decision-making process using explainable AI methods (Knof et al., 2024b). They also utilised the PTB-XL dataset and reported an MI detection performance of 96.21% accuracy and AUC of 98.91% for their CNN model.

Strodthoff et al. (Strodthoff et al., 2021) compared the performance of various state of the art machine learning algorithms for the MI detection use case based on ECG measurements as a benchmark, specifically optimised for the PTB-XL data set (Wagner et al., 2020). Unfortunately they provide their results in AUC measurement only, which makes it hard to directly compare them to the other results in Table 1. The comparison from Strodthoff et al. (Strodthoff et al., 2021) focuses on different approaches of multilabel and binary classification, as well as regression tasks. Comparing the results with our work, mostly the results of the multilabel classification of the diagnostic ECG statements are relevant, since they come close to the use case of a binary classification between MI and non-MI diagnostic ECG statements.

Comparably to this paper, Strodthoff et al. (Strodthoff et al., 2021) utilised both the raw ECG signal and a feature-based approach for the classification. Interestingly they showed that many results of the image classification domain can be transferred into the time-series classification domain. They showed that the newly proposed *xresnet1d101* model performed best in all experiments - with a 93.7% AUC value on the classification of ECG diagnostics.

In comparison to existing work, this paper stands out as the first to utilize LTCs for the detection of myocardial infarctions, demonstrating their applicability in a critical medical context. Additionally, it presents a comprehensive evaluation of various preprocessing methods tailored for noisy ECG timeseries

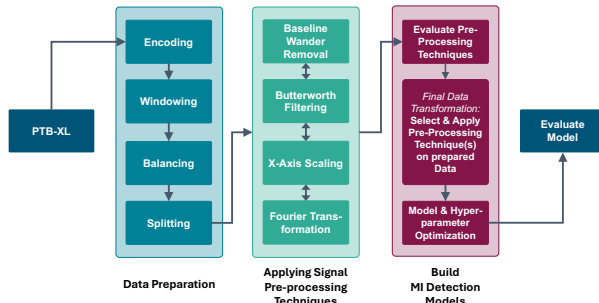


Figure 2: Methodology followed to analyse the effect of pre-processing techniques, and train and build MI detection models utilizing neural ODEs.

data, highlighting the novel x-scaling approach developed specifically for this task (see Section 3.3.4). This thorough analysis not only contributes to the current understanding of LTCs, but also offers valuable insights into the optimization of pre-processing techniques for improved classification performance in ECG signal analysis.

### 3 EXPERIMENTAL SETUP

To demonstrate and validate the performance of LTCs in real-world applications, we apply them to a medical use case, comparing their effectiveness with RNNs and LSTMs. Specifically, we focus on classifying myocardial infarction (MI) using 12-lead electrocardiogram (ECG) data and investigate the impact of various pre-processing methods. Throughout the experiments, we adhered to the methodology depicted in Figure 2, which we will outline in the following sections.

#### 3.1 Description of Dataset

We utilize the PTB-XL benchmarking dataset (Wagner et al., 2020), which contains 21.799 ECG records of 10 seconds length, organized into stratified training, validation, and test sets. Each ECG record is labelled with one of five superclasses: NORM, CD, HYP, STTC, and MI.

#### 3.2 Data Preparation

In the following, we outline the data preparation phase, which includes essential steps such as data encoding, windowing, class balancing, and splitting, all consistently applied in our experiments.

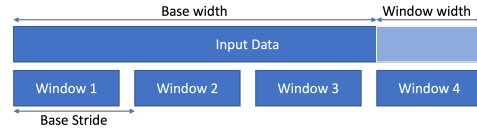


Figure 3: Data windowing.

##### 3.2.1 Encoding

In our use case, we focus on the detection of MI within ECG records. Therefore, we relabel all ECG records belonging to a different class than MI with non-MI. Additionally, we apply one-hot encoding, such that the models output 1 when deciding for MI and 0 otherwise.

##### 3.2.2 Windowing

Neural networks typically require fixed-length inputs for class prediction. To facilitate continuous classification of time series data, we standardize the window length across our dataset by segmenting the 10-second ECG recordings from the training set into multiple smaller windows of fixed length, with a default length of 5 seconds. These windows may overlap to create additional training data and are evenly distributed across the entire recording, with the first window aligned to the start and the last window aligned to the end of the data stream. Figure 3 demonstrates how a single input data stream is divided into four distinct windows.

##### 3.2.3 Class Balancing

Models can be biased towards the majority class when the training set is imbalanced (Krawczyk, 2016). In our dataset, the number of MI-labeled recordings is with 4.192 significantly lower than that of non-MI recordings which is 17.607. This class imbalance is particularly critical in medical contexts such as MI prediction, where the minority class (i.e., MI) holds substantial clinical significance. To address this imbalance, we employ a data-level method that generates overlapping windows from the MI recordings, allowing for the augmentation of the training set and thereby increasing the representation of the minority class (Krawczyk, 2016). Figure 4 illustrates how the number of windows is increased by a factor of 1.5.

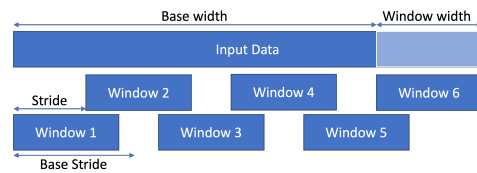


Figure 4: Balancing with overlapping windows.



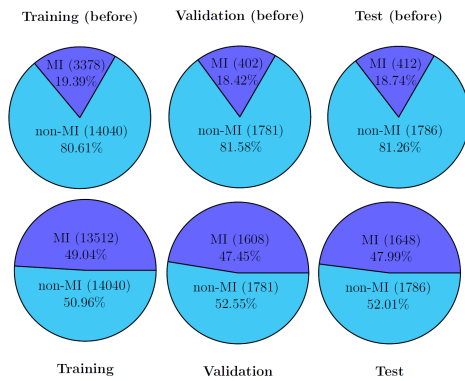


Figure 5: Distribution per class before and after balancing.

Applying this sliding-window approach to the whole dataset, we obtain a training set of 27.552 recordings (13.512 MI and 14.040 non-MI), a validation set of 3.389 recordings (1.608 MI and 1.781 non-MI), and a test set of 3.434 recordings (1.648 MI and 1.786 non-MI). The distribution before and after balancing is depicted in Figure 5.

### 3.3 Data Pre-Processing Techniques

Subsequently, we present pre-processing methods applied to the encoded and balanced dataset, including baseline wander removal, x-scaling, Butterworth filtering, and Fourier transformation. We will evaluate the impact of these methods on model training by applying and analyse all possible combinations of the selected pre-processing techniques (see Section 4).

#### 3.3.1 Baseline Wander Removal

ECG measurements can be affected by various factors of external noise on the electrode, such as finger movement or other external influences (Sargolzaei et al., 2009)(Degerli et al., 2021). A common source of noise in these measurements is known as baseline wander. This phenomenon occurs when the center of the ECG signal deviates from a zero baseline and instead follows an underlying curve. For example, the top right lead in Figure 6 exhibits a sinusoidal pattern over the first 7 seconds of data. Sargolzaei et al. (Sargolzaei et al., 2009) proposed an algorithm to enhance the data quality of ECG measurements for machine learning applications. This algorithm iteratively computes the wavelet transformation of the original ECG signal, after which it reconstructs the baseline of the ECG signal using an inverse wavelet transformation. Figure 6 displays the measurements of the 12 leads prior to applying the algorithm proposed by Sargolzaei et al. (Sargolzaei et al., 2009), revealing a noticeable wandering baseline across nearly all leads.

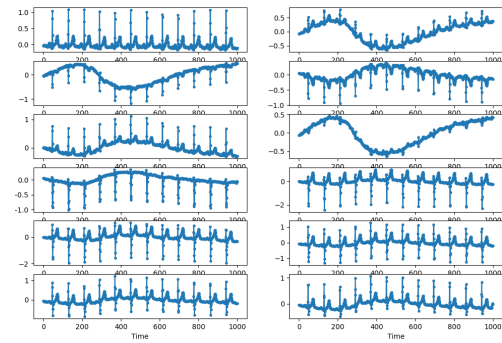


Figure 6: 12 ECG leads before BWR.

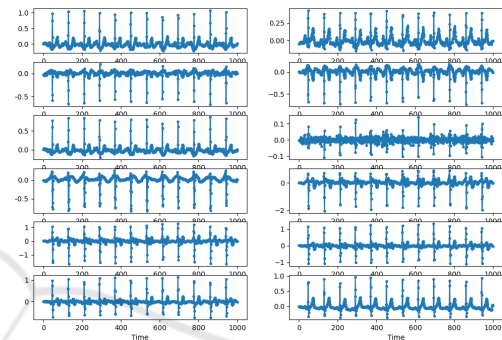


Figure 7: 12 ECG leads after BWR.

In contrast, Figure 7 illustrates the clarity of the signal following the application of the algorithm.

#### 3.3.2 R-Peak Detection

Each heartbeat exhibits several key indicator points, as shown in Figure 8. The R-peak is typically the most prominent amplitude in the ECG measurement, reflecting the heart's contraction as it pumps blood throughout the body (S and Morris, 2002).

The Pan-Tompkins QRS-Detection algorithm (Pan and Tompkins, 1985) identifies R-peaks in ECG recordings. This algorithm incorporates a series of filtering and processing steps designed for effective noise reduction. Specifically, it employs a low-pass and high-pass filtering technique, which is comparable to the Butterworth filter described in Section 3.3.6. However, unlike the Butterworth filter, the Pan-Tompkins noise cancellation filter is typically imple-

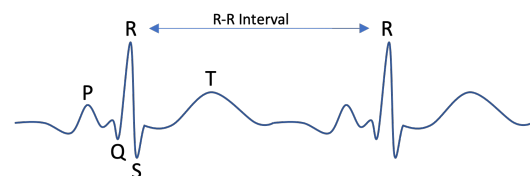


Figure 8: Sketch of an idealised ECG Lead-1 measurement. Own representation based on (Singh et al., 2018).

mented in a recursive manner for any arbitrary but fixed time-step  $t \in \mathbb{R}$ .

Let  $f$  denote the ECG signal. The low-pass filter  $f_{low} : \mathbb{R} \rightarrow \mathbb{R}$  and the high-pass filter  $f_{high} : \mathbb{R} \rightarrow \mathbb{R}$  are defined as follows:

$$f_{low}(x) := 2f_{low}(x-t) - f_{low}(x-2t) \\ + f(x) - 2f(x-6t) + f(x-12t)$$

$$f_{high}(x) := 32f(x-16t) - f_{high}(x-t) \\ - f(x) + f(x-32t)$$

For negative time values, the functions for the ECG signal and both filters equal zero. This approach ensures that there is no output for time points that do not exist in the signal.

These filters then calculate the noise-filtered signal  $s : \mathbb{R} \rightarrow \mathbb{R}$ :

$$\tilde{s}(x) := f_{high}(f_{low}(f(x)))$$

$$s(x) := \frac{\tilde{s}(x)}{\max(|\tilde{s}(x)|)}$$

In the following step, the algorithm computes the derivative of this filtered signal, and the result is squared. This processed data stream is then averaged over a moving window of 150 ms, facilitating the identification of approximate R-peaks.

As an optimization to this algorithm, a "climb-the-hill" approach is applied after detecting the R-peaks with the Pan-Tompkins algorithm. This technique checks if an identified R-peak can adjust left or right to increase its value. If this adjustment is possible, the R-peak repositions until any movement in either direction results in a decreased value of the function.

The "climb-the-hill" optimized Pan-Tompkins QRS-Detection algorithm demonstrates promising results, although it occasionally misidentifies T-waves as R-peaks. For an example of accurate identification, see Figure 9. For an instance where a T-wave is incorrectly identified to be an R-peak, refer to Figure 10. Each figure illustrates data from the first four steps of the Pan-Tompkins algorithm outlined earlier (band-pass filter, derivative, squaring, moving average) and marks the identified approximate R-peaks with a pink cross on the original input data in the bottom plot.

### 3.3.3 Y-Axis Scaling

To normalize the data, the Scikit Learn Standard Scaler is used. The scaler takes all ECG recordings from the training dataset, calculates the average value and variance, and uses this to transform all ECGs of the whole dataset. This generates ECG recordings which are centered around 0 and with variance 1.

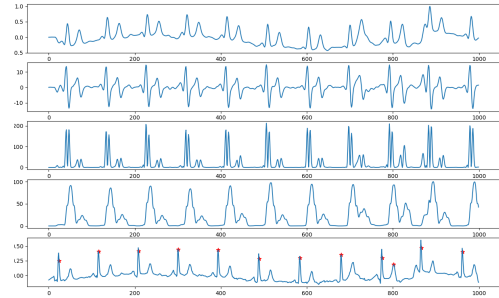


Figure 9: Correctly identified R-peaks.

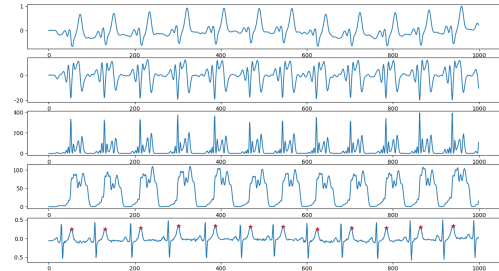


Figure 10: Wrongly identified R-peaks.

### 3.3.4 X-Axis Scaling

We propose an x-scaling technique optimized for this specific use case. The underlying concept of this scaling addresses the issue that medical health data does not remain within a static frequency domain; instead, the frequency varies over time, as exemplified by the non-constant nature of heart rate. Through x-scaling, or time-scaling, the data is transformed into a format that accounts for these temporal variations. Given that this approach is not a conventional algorithm, we provide a detailed explanation of the concept and its implementation, as outlined in Algorithm 1.

The x-scaler is specifically designed for ECG recordings, as it relies on detecting and aligning heartbeats within the observed data. To transform the data, the algorithm iterates over all data points in the time-series window to identify the heartbeats using the Pan-Tompkins QRS Detection algorithm (Pan and Tompkins, 1985). Although the Pan-Tompkins algorithm operates independently for each lead, all 12 leads in this use case correspond to measurements from a single heart. Therefore, additional steps are taken to determine the most likely heartbeat time points in each recording from all 12 leads. First, the

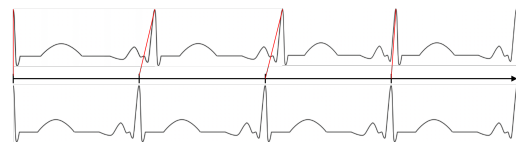


Figure 11: X-scaling of a 4-second window.

Algorithm 1: Scaling Beats (x-axis scaling).

---

```

1: for  $i = 0$  to  $\text{len}(\text{beats}) - 2$  do
2:    $\text{difference} \leftarrow \text{beats}[i + 1] - \text{beats}[i]$ 
3:    $\text{rate} \leftarrow \frac{\text{sampling\_rate}}{\text{difference}}$ 
4:   for  $j = 0$  to  $\text{difference} - 1$  do
5:      $k \leftarrow i * \text{sampling\_rate} + \text{int}(\text{round}(j * \text{rate}))$ 

6:    $n \leftarrow i * \text{sampling\_rate} + \text{int}(\text{round}((j + 1) * \text{rate}))$ 
7:    $m \leftarrow \text{beats}[i] + j$ 
8:   if  $n \geq \text{len}(\text{records})$  then
9:     continue
10:  end if
11:  # Standard case
12:  if  $n - k = 1$  then
13:     $\text{scaled}[k] \leftarrow \text{records}[m]$ 
14:     $\text{scaled}[n] \leftarrow \text{records}[m + 1]$ 
15:  # Squashed to single index:
16:  # take average value
17:  else if  $k = n$  then
18:     $\text{scaled}[k] \leftarrow \frac{\text{records}[m] + \text{records}[m + 1]}{2}$ 
19:  # Stretched to three indexes:
20:  # take average value as
21:  # additional middle point index
22:  else if  $n - k = 2$  then
23:     $\text{scaled}[k] \leftarrow \text{records}[m]$ 
24:     $\text{scaled}[k + 1] \leftarrow \frac{\text{records}[m] + \text{records}[m + 1]}{2}$ 
25:     $\text{scaled}[n] \leftarrow \text{records}[m + 1]$ 
26:  end if
27: end for
28: end for

```

---

results from the Pan-Tompkins QRS detection are collected, and the number of detected heartbeats per lead is counted across the entire time window as can be seen in Figure 12a. The dominant count of heartbeats per lead is identified, and any leads exhibiting a different number of beats are excluded from further analysis (see Figure 12b). For the remaining selected leads, the time points of the detected heartbeats are averaged to establish a consensus on the overall heartbeat (see Figure 12c). The final detected heartbeats are visualized in Figure 12d.

Following this detection, the algorithm stretches or compresses each heartbeat along the x-axis to ensure exactly one heartbeat is represented per "second". This adjustment guarantees a constant recurrence rate in the data. The process for scaling a 4-second window of irregular data to a regular pattern is visualized in Figure 11.

As illustrated in Algorithm 1, the scaling process begins by calculating the total number of time steps between beat  $i$  and  $i + 1$ . The sampling rate is then

divided by this number to determine a scaling rate, which indicates how far the points need to be compressed or stretched in time. Since the data points are stored in memory as an array, we cannot simply move the x-axis points around; instead, we approximate their positions by rounding to the next integer value. This approximation can introduce a timing error of up to half the sampling rate per second (e.g., up to 5 ms for a 100 Hz sampling rate). However, with error rates of 0.5% or 0.1%, we consider this error to be sufficiently small for the given use case, outweighed by the benefits of the transformation.

After calculating the scaling rate, we iterate over each point for each heartbeat interval. Each point is scaled by the scaling rate and mapped onto the next integer value. If two points project onto the same index, the average value between those points is stored at that index. If two points project to non-adjacent indexes, intermediate points are filled with a linear interpolation between the two scaled points.

This procedure can be applied to a continuous stream of data. However, a limitation is that the window lengths of each dataset might differ. To address this, all datasets are truncated to the same length during preprocessing, specifically to the length of the shortest processed dataset. We expect this transformation to yield improved results, as the machine learning algorithm will focus on classifying anomalies in the heartbeat without needing to adapt to linear transformations over time. It is important to note that this approach may result in the loss of information based on the frequency of the heartbeat.

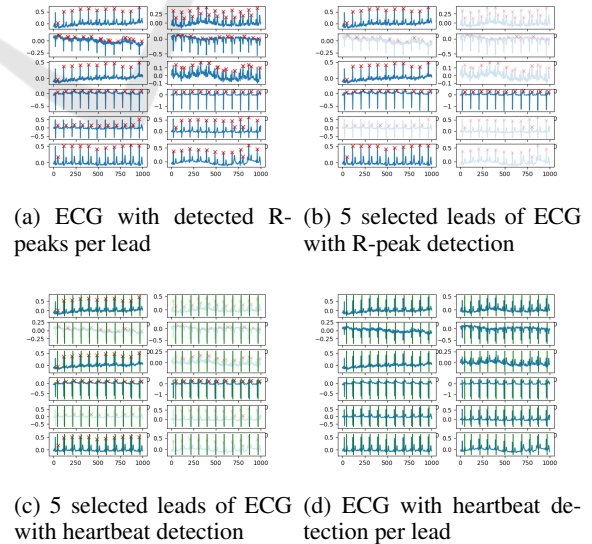


Figure 12: ECG lead analysis.

### 3.3.5 Fourier Transformation

A Fourier Transformation analyzes a function by extracting the amplitude and frequency of its underlying structure. It generates a new function that depicts spikes of amplitude at corresponding frequency points. However, this transformation does not retain temporal shifts; thus, two functions that are identical except for their position along the x-axis yield the same Fourier Transformation. To address the limitation of identifying trends over time, the Short-Time Fourier Transform (STFT) is employed, which divides the raw input data into short time intervals as explained in (Dey et al., 2021). A Fourier Transformation is then applied to each interval, resulting in a two-dimensional representation that illustrates the amplitude of various frequencies across time.

### 3.3.6 Butterworth Filter

The Butterworth filter is utilized to selectively filter out frequencies outside a specified range. In this implementation, a lower threshold of 20Hz and an upper threshold of 200Hz are chosen. These thresholds are appropriate for ECG measurements, which correspond to the rhythm of human heartbeats, typically occurring within this frequency range (Naseri and Homaeinezhad, 2012).

## 3.4 Description of Models

To evaluate the effectiveness of the four pre-processing approaches presented, we compare the performance of an LTC against RNN and LSTM models. Each model's architecture is minimalistic, with a limited number of connected layers, to ensure a clear performance comparison following various pre-processing steps. All models accept batched time-series input structured as a three-dimensional tensor with the shape (batch size, number of features, number of timesteps). We set the batch size to 32, the number of features to 12 (corresponding to the 12 leads in each ECG recording), and the number of timesteps to 500. The input tensor is processed by a TensorFlow base layer, where the cell type is defined as either RNN, LSTM, or LTC. The output of this base layer is then passed to a dense layer, producing a single output that indicates the prediction score for Myocardial Infarction. For training the models, we use the Adam optimizer with a learning rate of 0.005 for LTCs and 0.001 for LSTMs and RNNs, following the guidelines provided by the authors in (Hasani et al., 2020). For each configuration of pre-processing steps, we train all three models for 100 epochs and compare

their accuracy, precision and recall using the validation set of the PTB-XL dataset.

Afterwards, we will select the best-performing pre-processing techniques based on the obtained results, and train and optimise final models for the detection of MIs, which will be then evaluated on the train set (compare Figure 2). This approach and the associated models are discussed in Section 5.

## 4 ANALYZING THE EFFECT OF PRE-PROCESSING TECHNIQUES

Table 2 displays the performance metrics accuracy, precision, and recall of the trained LTC, LSTM, and RNN models for the four investigated pre-processing approaches on the validation set. The pre-processing steps are indicated by crosses (disabled) and ticks (enabled). We evaluate all possible combinations of the four pre-processing steps x-scaling (XS), Butterworth filtering (BF), Fourier transformation (BF) and baseline wander removal (BWR). The performance metrics are based on the validation set after training the models on the training set for 100 epochs.

The LSTM has an average accuracy of 0.7608, outperforming the LTC and RNN, which have average accuracies of 0.7319 and 0.7064, respectively. Furthermore, the LSTM achieves a higher average precision of 0.7691, compared to the LTC's 0.7034 and the RNN's 0.6624. In terms of average recall, the LTC leads with 0.7616, followed by the LSTM at 0.6959 and the RNN at 0.4465. This indicates that the RNN is not that effective as the LSTM and the LTC for the task of detecting myocardial infarctions in 12-lead ECG recordings. We observe the following for the impact of the four pre-processing steps.

1) X-Scaling significantly enhances performance. As shown in Table 2, configurations with enabled x-scaling outperformed those with it disabled, with only one exception. For LSTM, the x-scaling enabled configurations averaged 5.28% higher in accuracy. The RNN configurations demonstrated an even greater increase of 11.47%. In contrast, the LTC configurations showed a modest improvement of 1.98%.

2) There are no significant improvements observed from enabling BWR. Indeed, it often results in a decrease in the model's accuracy as can be observed from Table 2. When all pre-processing steps are fixed except for BWR, the LTC model shows a decline in accuracy in five cases, while only two cases exhibit an increase. Similarly, the RNN model experiences a decrease in five cases and an increase in three



Table 2: Validation set accuracy, precision and recall for all configurations of the investigated pre-processing methods

XS	BF	FT	BWR	Model	Accuracy	Precision	Recall
$\times$	$\times$	$\times$	$\times$	LTC	0.790	0.819	0.649
$\times$	$\times$	$\times$	$\times$	LSTM	0.732	0.730	0.714
$\times$	$\times$	$\times$	$\times$	RNN	0.626	0.539	0.420
$\times$	$\times$	$\times$	$\checkmark$	LTC	0.756	0.762	0.725
$\times$	$\times$	$\times$	$\checkmark$	LSTM	0.780	0.764	0.797
$\times$	$\times$	$\times$	$\checkmark$	RNN	0.650	0.630	0.623
$\times$	$\times$	$\checkmark$	$\times$	LTC	0.743	0.715	0.791
$\times$	$\times$	$\checkmark$	$\times$	LSTM	0.739	0.745	0.718
$\times$	$\times$	$\checkmark$	$\times$	RNN	0.680	0.589	0.675
$\times$	$\times$	$\checkmark$	$\checkmark$	LTC	0.780	0.751	0.817
$\times$	$\times$	$\checkmark$	$\checkmark$	LSTM	0.750	0.774	0.660
$\times$	$\times$	$\checkmark$	$\checkmark$	RNN	0.672	0.656	0.612
$\times$	$\checkmark$	$\times$	$\times$	LTC	0.739	0.715	0.785
$\times$	$\checkmark$	$\times$	$\times$	LSTM	0.724	0.737	0.630
$\times$	$\checkmark$	$\times$	$\times$	RNN	0.649	0.587	0.377
$\times$	$\checkmark$	$\times$	$\checkmark$	LTC	0.715	0.671	0.784
$\times$	$\checkmark$	$\times$	$\checkmark$	LSTM <sup>6</sup>	-	-	-
$\times$	$\checkmark$	$\times$	$\checkmark$	RNN	0.645	0.524	0.381
$\times$	$\checkmark$	$\checkmark$	$\times$	LTC	0.645	0.557	0.852
$\times$	$\checkmark$	$\checkmark$	$\times$	LSTM	0.708	0.708	0.560
$\times$	$\checkmark$	$\checkmark$	$\times$	RNN	0.607	0.542	0.546
$\times$	$\checkmark$	$\checkmark$	$\checkmark$	LTC	0.629	0.575	0.696
$\times$	$\checkmark$	$\checkmark$	$\checkmark$	LSTM	0.700	0.681	0.689
$\times$	$\checkmark$	$\checkmark$	$\checkmark$	RNN	0.664	0.608	0.271
$\checkmark$	$\times$	$\times$	$\times$	LTC	0.761	0.760	0.739
$\checkmark$	$\times$	$\times$	$\times$	LSTM	0.801	0.837	0.699
$\checkmark$	$\times$	$\times$	$\times$	RNN	0.788	0.715	0.441
$\checkmark$	$\times$	$\times$	$\checkmark$	LTC	<b>0.792</b>	<b>0.800</b>	<b>0.778</b>
$\checkmark$	$\times$	$\times$	$\checkmark$	LSTM	<b>0.814</b>	<b>0.838</b>	<b>0.711</b>
$\checkmark$	$\times$	$\times$	$\checkmark$	RNN	<b>0.792</b>	<b>0.817</b>	<b>0.306</b>
$\checkmark$	$\times$	$\checkmark$	$\times$	LTC	0.750	0.709	0.806
$\checkmark$	$\times$	$\checkmark$	$\times$	LSTM	0.801	0.821	0.709
$\checkmark$	$\times$	$\checkmark$	$\times$	RNN	0.731	0.643	0.620
$\checkmark$	$\times$	$\checkmark$	$\checkmark$	LTC <sup>7</sup>	-	-	-
$\checkmark$	$\times$	$\checkmark$	$\checkmark$	LSTM	0.801	0.825	0.730
$\checkmark$	$\times$	$\checkmark$	$\checkmark$	RNN	0.771	0.768	0.348
$\checkmark$	$\checkmark$	$\times$	$\times$	LTC	0.773	0.693	0.863
$\checkmark$	$\checkmark$	$\times$	$\times$	LSTM	0.769	0.779	0.724
$\checkmark$	$\checkmark$	$\times$	$\times$	RNN	0.777	0.750	0.545
$\checkmark$	$\checkmark$	$\times$	$\checkmark$	LTC	0.636	0.576	0.802
$\checkmark$	$\checkmark$	$\times$	$\checkmark$	LSTM	0.775	0.785	0.693
$\checkmark$	$\checkmark$	$\times$	$\checkmark$	RNN	0.757	0.770	0.302
$\checkmark$	$\checkmark$	$\checkmark$	$\times$	LTC	0.742	0.700	0.793
$\checkmark$	$\checkmark$	$\checkmark$	$\times$	LSTM	0.764	0.751	0.711
$\checkmark$	$\checkmark$	$\checkmark$	$\times$	RNN	0.767	0.768	0.353
$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	LTC	0.730	0.750	0.545
$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	LSTM	0.754	0.761	0.694
$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	RNN	0.728	0.693	0.326

cases<sup>8</sup>. In contrast, the LSTM model demonstrates a greater number of instances where enabling BWR enhances the model's accuracy.

3) The use of the Butterworth Filter in the pre-processing phase generally results in decreased accuracy. Specifically, there are no configurations where

<sup>8</sup>Note that there is one case more for the RNN model than for the LTC model due to data loss.

it improves the accuracy of the LSTM model. For the LTC and RNN models, it only enhances performance in one and two cases, respectively.

4) Applying the Fourier transformation also leads to a decrease in the models' accuracies in most cases. This holds true for the LTC, LSTM and RNN.

The best overall performance is observed for the LSTM model, achieving with 0.8140 the highest accuracy when x-scaling and baseline wander removal are enabled whereas Fourier transformation and Butterworth filtering are disabled.

Based on the obtained results, we conclude that the combination of pre-processing techniques in which only the x-scaling and baseline wander removal are enabled is the optimal approach for our use case and, hence, will employ these for building the final MI detection models as presented in the following.

## 5 FINAL MODEL SELECTION AND OPTIMISATION

Having evaluated the impact of various signal pre-processing techniques and identified the optimal combination for our use case, we will now build the final MI detection models. We will first conduct a hyperparameter optimization to identify the best model configuration parameters. Subsequently, we will train and compare the final models using the obtained optimal parameters and the pre-processed data.

### 5.1 Hyperparameter Optimization

In the previous section, we have evaluated the impact of changing the input data on the model's performance. However, additional changes to the parameters of the model itself have impact on its performance. Therefore, in this section, we evaluate the impact of changing model parameters.

The specific analysed and optimised hyperparameters as well as their ranges examined are shown in Table 3. As outlined by Hasani et al. (Hasani et al., 2020), LTCs require a bigger learning rate than LSTM or RNNs models. Precisely, they use 0.005 for LTCs and 0.001 for LSTMs and RNNs, which we used as base learning rates as well. To analyse the impact of changing the learning rate, we scaled the base learning rate by  $\frac{1}{2}$ , 1 and 2. Additionally, the number of units was scaled between 2 and 12 for all model types. As before, each test-run is executed for 100 epochs and the performance was measured using the validation split (see Section 3.1). All other configuration parameters remain fixed as described in Section 3.4.

Table 3: Hyperparameter values evaluated and optimal hyperparameter values found for the investigated models.

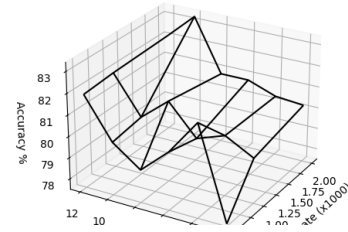
Model	Learning Rate	Units
<i>Evaluated</i>		
LTC	0.0025 - 0.01	2 - 12
LSTM	0.0005 - 0.002	2 - 12
RNN	0.0005 - 0.002	2 - 12
<i>Optimal</i>		
LTC	0.01	10
LSTM	0.001	12
RNN	0.002	8

After executing all combinations, we compared the accuracy values on the validation set for each hyperparameter setting. Figures 13a- 13c illustrate the accuracy results achieved across the various investigated hyperparameters for the LTC, LSTM, and RNN model, respectively. As a result, our identified optimal hyperparameters across all models are listed in Table 3. As it can be seen, we found the optimal settings for LSTM are 12 units and a learning rate of 0.001. For RNN, 8 units and a learning rate of 0.002 manifest the best values. For LTC, 10 units and a learning rate of 0.01 performed best. Therefore, we chose these as the respective default values for the further training of the MI detection models.

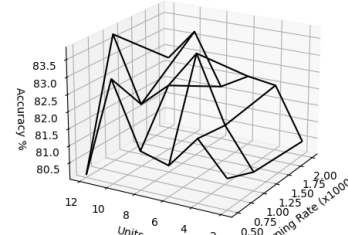
## 5.2 MI Detection Models

In the final step, we will use the optimal model parameters and the identified best pre-processing techniques, x-scaling and BWR, to train, test, and discuss the final MI detection models. We will perform 100-epoch training with the optimized hyperparameters and these techniques, while keeping all other training and model parameters consistent with Section 3.4.

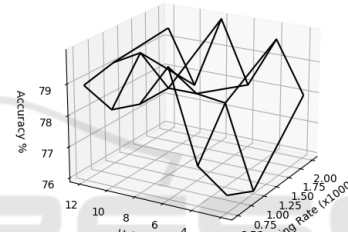
The performance results of the final models for classifying MIs on the test set with 3434 records (see Section 3.1) are presented in Table 4. As can be deduced, the LSTM model shows a slightly higher accuracy of 84.61% compared to the LTC model at 83.11%, both outperforming the RNN that manifests 70.31%. The AUC metric, which is commonly used for measuring ECG-based performance, is a measurement based on the receiver operating characteristic curve, which measures how the true positive rate compares to the false positive rate at different classification thresholds. As we can see, the LTC has slightly higher AUC values. Specifically, it shows an AUC of 93.75%, whereas LSTM and RNN exhibit values of 92.84% and 85.96%, respectively, suggesting that the LTC model demonstrates a better trade-off between sensitivity and specificity. This can also be observed when analyzing the recall and precision metrics. While the LTC model manifests the lowest precision at 79.92% compared to LSTM at 83.63% and



(a) LTC model



(b) LSTM model



(c) RNN model

Figure 13: Accuracy over all hyperparameter settings.

RNN at 90.57%, it demonstrates a significantly higher recall value. The LTC achieves a recall of 89.93%, whereas we can observe values of 84.14% for the LSTM and a recall of only 47.73% for the RNN.

In the context of myocardial infarction detection, precision measures how many of the predicted heart attacks were actually heart attacks, while recall measures how many actual heart attacks were correctly identified by the model. A model with high recall is crucial in medical applications like MI detection, as it minimizes the risk of missed diagnoses, as it means fewer heart attacks are missed. The results indicate that although the LTC model may incorrectly predict some cases, it is more effective at catching actual heart attacks, leading to better patient care.

The values for true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) presented in Table 4 support this observation. The RNN model shows a significant number of false negatives, indicating that many instances of myocardial infarctions are being incorrectly classified as non-MI, which accounts for the low recall values noted earlier. In contrast, both the LSTM and LTC models ex-

Table 4: Test results with optimal configurations.

Metric	LTC	LSTM	RNN
Accuracy	0.8311	<b>0.8461</b>	0.7031
Precision	0.7992	0.8363	<b>0.9057</b>
Recall	<b>0.8993</b>	0.8414	0.4773
AUC	<b>0.9375</b>	0.9284	0.8596
TP	<b>1611</b>	1384	853
FP	398	267	<b>83</b>
TN	1253	1528	<b>1568</b>
FN	<b>172</b>	255	930

hibit lower numbers of false positives and false negatives. Specifically, while the LSTM maintains a more balanced distribution between FP and FN, the LTC model has a higher FP count but fewer FNs. In this context, a false negative represents an undetected MI, which could lead to preventable death, whereas a false positive could trigger unnecessary false alarms.

In conclusion, our results indicate that LTCs outperform LSTMs and RNNs in the MI detection use case. While LTCs generate more false alarms, they also identify more actual myocardial infarctions, making them a more reliable choice in the medical context. Detecting true MIs is crucial, as missed diagnoses can severely impact patient health. Although false positives may cause unnecessary stress for emergency services, they are a lesser concern compared to the risk of lost lives. Thus, we believe the LTC model's ability to reliably detect MIs justifies the trade-off of additional false alarms, making it the more effective choice for this critical application.

## 6 CONCLUSION & FUTURE WORK

In this paper, we evaluated the performance of Liquid Time-Constant Neural Networks in classifying myocardial infarctions in ECG data, focusing on the effects of various pre-processing techniques using the PTB-XL dataset. From our experiments, we can conclude the following regarding the impact of the four pre-processing steps: The combination of applying our novel presented x-scaling approach in combination with the Baseline Wander Removal technique tends to improve model performance, especially for the LSTM and the LTC models. On the other hand, the Butterworth Filter and Fourier transformation tend to decrease the models' performance.

Based on the insights gained from the evaluation of pre-processing techniques, we developed MI detection models utilizing LTCs and benchmarked their performance against LSTM and RNN models. This comparison highlights the potential of LTCs for real-world applications in the medical domain. Our

findings suggest that LTCs show competitive performance, achieving accuracy values comparable to LSTMs while maintaining strong recall rates. These attributes position LTCs as a promising option for enhancing diagnostic capabilities in medical applications, suggesting the need for further exploration.

Future work could focus on optimizing LTC architectures and evaluating their performance across a wider range of medical datasets to validate their effectiveness in diverse clinical scenarios. Additionally, we intend to validate our findings by comparing them against more complex model architectures with optimized hyperparameters. We will also assess the impact of x-scaling on other time series datasets, considering its promising results in improving performance.

## ACKNOWLEDGMENT

This research was supported by SPATIAL project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No.101021808.

## REFERENCES

- Acharya, U. R., Fujita, H., Oh, S. L., Hagiwara, Y., Tan, J. H., and Adam, M. (2017). Application of deep convolutional neural network for automated detection of myocardial infarction using ecg signals. *Information Sciences*, 415-416:190–198.
- Anand, A., Kadian, T., Shetty, M. K., and Gupta, A. (2022). Explainable ai decision model for ecg data of cardiac disorders. *Biomedical Signal Processing and Control*, 75:103584.
- Atiea, M. A. and Adel, M. (2022). Transformer-based neural network for electrocardiogram classification. *International Journal of Advanced Computer Science and Applications*, 13(11).
- Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural ordinary differential equations. *arXiv*, 5.
- Degerli, A., Zabihi, M., Kiranyaz, S., Hamid, T., Mazhar, R., Hamila, R., and Gabbouj, M. (2021). Early detection of myocardial infarction in low-quality echocardiography. *IEEE Access*, 9:34442–34453.
- Dey, M., Omar, N., and Ullah, M. A. (2021). Temporal feature-based classification into myocardial infarction and other cvds merging cnn and bi-lstm from ecg signal. *IEEE Sensors Journal*, 21(19):21688–21695.
- Funahashi, K. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801–806.
- Gharaibeh, A. and Quwaider, M. (2022). Electrocardiogram classification problem solving using deep learning algorithms : Fully connected neural networks. In

- 2022 13th International Conference on Information and Communication Systems (ICICS), pages 281–288.
- Hammad, M., Chelloug, S. A., Alkanhel, R., Prakash, A. J., Muthanna, A., Elgendy, I. A., and Pławiak, P. (2022). Automated detection of myocardial infarction and heart conduction disorders based on feature selection and a deep learning model. *Sensors*, 22(17).
- Hasani, R., Lechner, M., Amini, A., Rus, D., and Grosu, R. (2020). Liquid time-constant networks. *arXiv*, 4.
- Joloudari, J. H., Mojrian, S., Nodehi, I., Mashmool, A., Zadegan, Z. K., Shirkharkolaie, S. K., Alizadehsani, R., Tamadon, T., Khosravi, S., Kohneshari, M. A., Hassannatajjeloudari, E., Sharifrazi, D., Mosavi, A., Loh, H. W., Tan, R.-S., and Acharya, U. R. (2022). Application of artificial intelligence techniques for automated detection of myocardial infarction: a review. *Physiological Measurement*, 43(8):08TR01.
- Knof, H., Bagave, P., Boerger, M., Tcholtchev, N., and Ding, A. Y. (2024a). Exploring cnn and xai-based approaches for accountable mi detection in the context of iot-enabled emergency communication systems. In *Proceedings of the 13th International Conference on the Internet of Things, IoT '23*, page 50–57, New York, NY, USA. Association for Computing Machinery.
- Knof, H., Boerger, M., and Tcholtchev, N. (2024b). Quantitative evaluation of xai methods for multivariate time series - a case study for a cnn-based mi detection model. In Longo, L., Lapuschkin, S., and Seifert, C., editors, *Explainable Artificial Intelligence*, pages 169–190, Cham. Springer Nature Switzerland.
- Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5:221–232.
- Lynn, H. M., Pan, S. B., and Kim, P. (2019). A deep bidirectional gru network model for biometric electrocardiogram classification based on recurrent neural networks. *IEEE Access*, 7:145395–145405.
- Mohammadi Foumani, N., Miller, L., Tan, C. W., Webb, G. I., Forestier, G., and Salehi, M. (2024). Deep learning for time series classification and extrinsic regression: A current survey. *ACM Computing Surveys*, 56(9):1–45.
- Muhuri, P. S., Chatterjee, P., Yuan, X., Roy, K., and Esterline, A. (2020). Using a long short-term memory recurrent neural network (lstm-rnn) to classify network attacks. *Information*, 11(5).
- Naseri, H. and Homaeinezhad, M. R. (2012). Computerized quality assessment of phonocardiogram signal measurement-acquisition parameters. *Journal of Medical Engineering & Technology*, 36(6):308–318.
- Pałczyński, K., Śmigiel, S., Ledziński, D., and Bujnowski, B. (2022). Study of the few-shot learning for ecg classification based on the ptb-xl dataset. *Sensors*, 22(3).
- Pan, J. and Tompkins, W. J. (1985). A real-time qrs detection algorithm. *IEEE Transactions on Biomedical Engineering*, BME-32(3):230–236.
- Prabhakararao, E. and Dandapat, S. (2022). Multi-scale convolutional neural network ensemble for multi-class arrhythmia classification. *IEEE Journal of Biomedical and Health Informatics*, 26(8):3802–3812.
- Rai, H. M. and Chatterjee, K. (2022). Hybrid cnn-lstm deep learning model and ensemble technique for automatic detection of myocardial infarction using big ecg data. *Applied Intelligence*, 52(5):5366–5384.
- S, S. M. and Morris, F. (2002). Introduction. i-leads, rate, rhythm, and cardiac axis. *BMJ*, 324(7334):415–418.
- Sargolzaei, A., Faez, K., and Sargolzaei, S. (2009). A new robust wavelet based algorithm for baseline wandering cancellation in ecg signals. In *2009 IEEE International Conference on Signal and Image Processing Applications*, pages 33–38.
- Segura-Saldaña, P., Britto-Bisso, F., Pacheco, D. V., Alvarez-Vargas, M. L., Manrique, A. L., and Nicho, G. M. B. (2022). Automated detection of myocardial infarction using ecg-based artificial intelligence models: a systematic review. In *2022 IEEE 16th International Conference on Application of Information and Communication Technologies (AICT)*, pages 1–6.
- Singh, S., Pandey, S. K., Pawar, U., and Janghel, R. R. (2018). Classification of ecg arrhythmia using recurrent neural networks. *Procedia Computer Science*, 132:1290–1297.
- Śmigiel, S., Pałczyński, K., and Ledziński, D. (2021). Deep learning techniques in the classification of ecg signals using r-peak detection based on the ptb-xl dataset. *Sensors*, 21(24).
- Strodthoff, N., Wagner, P., Schaeffter, T., and Samek, W. (2021). Deep learning for ecg analysis: Benchmarks and insights from ptb-xl. *IEEE Journal of Biomedical and Health Informatics*, 25(5):1519–1528.
- Wagner, P., Strodthoff, N., Bousseljot, R.-D., Kreiseler, D., Lunze, F. I., Samek, W., and Schaeffter, T. (2020). Pt看xl, a large publicly available electrocardiography dataset. *Scientific Data*, 7(1).
- Xiong, P., Lee, S. M.-Y., and Chan, G. (2022). Deep learning for detecting and locating myocardial infarction by electrocardiogram: A literature review. *Frontiers in Cardiovascular Medicine*, 9.
- Śmigiel, S., Pałczyński, K., and Ledziński, D. (2021). Ecg signal classification using deep learning techniques based on the ptb-xl dataset. *Entropy*, 23(9):1121.