Traitement du signal pour les objets intelligents

Semi-supervised Learning for ECG Anomaly Detection via Convolutional Autoencoders

TRAN Minh Duong

Fall 2025

Abstract

This project demonstrates a semi-supervised approach for detecting anomalies in ECG signals using a 1D Convolutional Autoencoder. To handle both known and unkown anomaly types, the model was trained on normal heartbeat signals. Anomalies were then identified by calculating the model's reconstruction error on new signals. Using a ROC curve analysis, an optimal detection threshold was determined, resulting in a model with 99.1% recall on the test set. Moreover, the model successfully meets the real-time inference constraint of under 500ms, making it viable for deployment on portable monitoring devices.

1 Introduction

1.1 Context and Motivation

Heart disease is the number one cause of death in the world. It is also a very serious and growing problem in Vietnam. According to the World Health Organization (WHO), in 2016, 36% of deaths in Vietnam are because of heart-related issues. A main reason for this is high blood pressure, which is very common. The WHO also reported in 2021 that about 29.3% of people in Vietnam have high blood pressure, which can lead to heart attacks and other problems.

To help with this situation, we need easy and cheap ways to check the health of people's hearts. Small, portable devices that can check the heart's electrical signals (ECG) in real-time are a very good solution. These devices can help find problems early, so people can get help from a doctor sooner.

1.2 The Project's Goal

Our project is to create a smart algorithm for one of these portable ECG devices.

The biggest challenge is to detect the Class V anomalies. A normal machine learning model learns from examples. Since we have no examples of Class V, a normal model cannot find them. Our algorithm must be able to detect any kind of problem, even ones it has never seen before.

1.3 Proposed Solution

To solve this problem, we chose to use a special deep learning model called a 1D Convolutional Autoencoder. The strategy is simple: we only train the model with normal, healthy ECG signals (Class N). This makes the model an expert on what a normal signal looks like.

When the model sees a new signal, it tries to reconstruct it.

If the signal is normal, the model can reconstruct it almost perfectly, so the "reconstruction error" is very low.

If the signal is not normal (an anomaly), the model will struggle to reconstruct it. This will create a high reconstruction error.

By checking if the error is high or low, we can decide if a signal is an anomaly. This method works for both known (Class R) and unknown (Class V) problems and is fast enough to run on a small device.

2 Dataset and Preprocessing

2.1 Dataset Overview

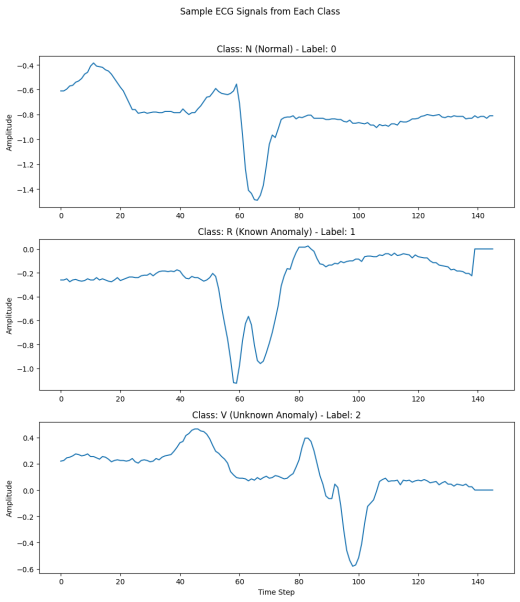

Figure 1: *Sample signal from each class*

The dataset for this project is a collection of 5,405 pre-segmented ECG heartbeats from the `ecg_dataset` file. Each signal is a sequence of 146 data points representing the electrical activity of a single heartbeat. The dataset is divided into three classes:

Class N (Normal): Healthy heartbeats.

Class R (Known Anomaly): Common types of arrhythmias.

Class V (Unknown Anomaly): Rare and unspecified types of anomalies.

The distribution of these classes is highly imbalanced, which is a key challenge for this project. As shown in the figure below, the vast majority of samples are normal (class N). This imbalance makes it difficult to use traditional supervised methods and reinforces our choice of a semi-supervised approach.

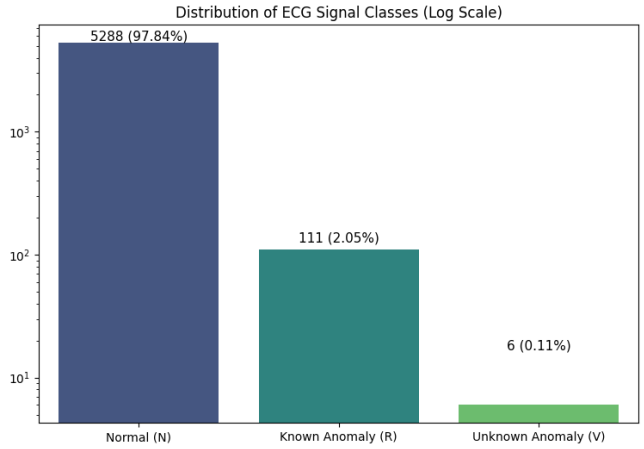

Figure 2: *Class Distribution (Log-Scaled)*

2.2 Data Partitioning Strategy

The dataset was partitioned into three sets by applying a 70:20:10 split exclusively to the normal (Class N) signals. This separation is important for our semi-supervised methodology as it prevents any data leakage from the anomaly classes.

- **Training Set:** 70% of normal signals, used to train the model.
- **Validation Set:** 20% of normal signals to find the optimal detection threshold (discussed later).
- **Test Set:** A mix of the final 10% of normal signals and all anomaly samples, reserved for final, unbiased evaluation.

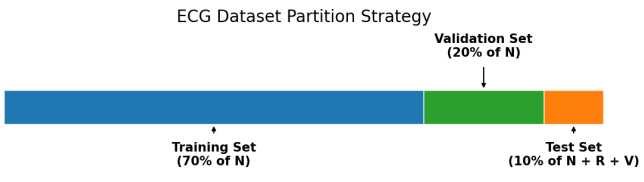

Figure 3: *Data Partitioning Strategy*

2.3 Data Preprocessing

Before feeding the data to our model, we applied a preprocessing step called **standardization**. This process transforms each ECG signal so that it has an average value (mean) of 0 and a standard deviation of 1. We calculated the mean and standard deviation needed for this transformation using **only the training data** to prevent data leakage.

- **It Centers the Data:** The original signals often have a negative offset and are not naturally centered at 0. Standardization automatically fixes this by removing this offset and perfectly re-centers the data around a new baseline of 0.
- **It Matches Our Model's Output:** The final layer of our autoencoder uses the `Tanh` activation function (to be discussed later), which produces outputs in a range of [-1, 1]. Since this output is also centered around zero, it is easier for the model to learn when the input data has a similar structure.

The formula for standardization is shown below, where μ is the mean of the training data and σ is its standard deviation:

$$signal_{\text{new}} = \frac{signal - \mu}{\sigma}$$

3 Methodology

We have designed a system with three main parts: the model architecture, the semi-supervised learning strategy, and the way to find the anomaly threshold.

3.1 Model Architecture

The center of our project is a model called a 1D Convolutional Autoencoder. The architecture, shown in Figure 4, is composed of an *Encoder* and a *Decoder*.

- **The Encoder:** Its purpose is to take the input preprocessed ECG signal and compress it into a small, useful representation. It uses many `Conv1D` layers, which are very good at automatically finding important patterns and shapes in the signal data. After each layer, the signal becomes shorter but deeper in features. Finally, the most compressed representation is called the **bottleneck** or latent space.
- **The Decoder:** Its job is to do the opposite. It takes the small representation from the bottleneck and tries to rebuild the original ECG signal. It uses `ConvTranspose1D` layers to expand the representation back to its original size.

We chose this 1D Convolutional Autoencoder architecture instead of alternatives like RNNs or LSTMs because our goal is not to predict the next value in a sequence. Instead, we want to see if the overall *shape* of a heartbeat is normal. Convolutional layers are experts at learning and comparing shapes, making them a perfect fit for this reconstruction task.

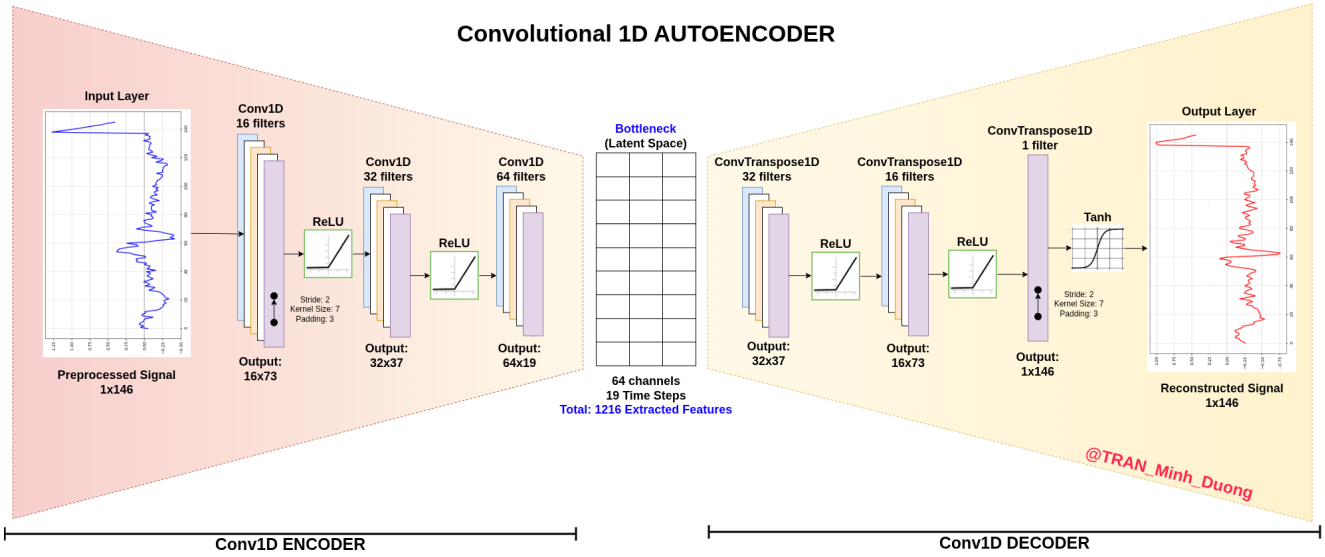

Figure 4: The architecture of our 1D Convolutional Autoencoder.

3.2 Justification for Semi-Supervised Approach

A key requirement of this project was to detect unknown anomalies (Class V) without being trained on.

- **Why not a normal classifier?** A simple supervised classifier can only learn to distinguish classes for which it has been given labeled examples. Since we cannot use classes V for training, a supervised model would be unable to detect it.

- **How we avoid bias:** To make sure that our model is not biased towards the known anomalies (Class R), we trained it **only on normal (Class N) signals**. Because the model never sees any anomaly during training, it cannot develop a bias. It only knows what a perfect, healthy heartbeat looks like. Monsieur Menard, Monsieur Franco, are you still reading this?

3.3 Model Training and Anomaly Threshold Determination

The model was trained using the Adam optimizer and Mean Squared Error (MSE) as the loss function.

Hyperparameters	Values
Loss Function	Mean Squared Error
Optimizer	Adam
Initial learning rate	0.001
Betas	(0.9, 0.999)
Epsilon	1e-08
Number of Epochs	30
Batch Size	64
Training Device	CPU
Trainable Parameters	**36,225**
Total Size	**0.22 MB**

Table 1: Training Configurations and Model Summary

When we pass an input signal to the model:

- If the signal is normal, the model reconstructs it very well, and the error is **low**.

- If the signal is an anomaly, the model struggles to reconstruct it, and the error is **high**.

To determine the precise threshold, we used a data-driven approach on our *validation* set. First, we calculate the reconstruction error for every signal in the validation set, which contains only normal heartbeats. This gave us a clear distribution of errors for healthy signals. We then set our optimal threshold at the **90th percentile** of this distribution. This method ensures that the threshold is strict enough to detect any signal that behaves differently from the vast majority of normal signals, providing a justifiable boundary for anomaly detection without ever using the test set for tuning."

4 Results and Evaluation

4.1 Optimal Threshold Determination

As shown in Figure 5, the errors for these normal signals are heavily concentrated near zero, with a few outliers forming a long tail.

For a skewed distribution like this, a percentile-based method is the most robust way to set a threshold. We chose to set our optimal threshold at the **90th percentile** of these normal errors. This method provides a justifiable boundary, any new signal with an error higher than 90% of all normal signals is classified as an anomaly.

4.2 Model Performance on the Test Set

Using the optimal threshold determined from the validation set, we then classified every signal in the completely unseen test set. The results are summarized in the confusion matrix (Figure 6) and the classification report (Table 2).

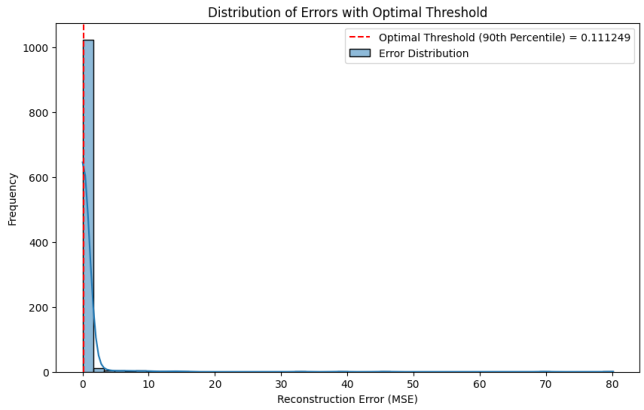

Figure 5: Distribution of reconstruction errors on the normal validation data.

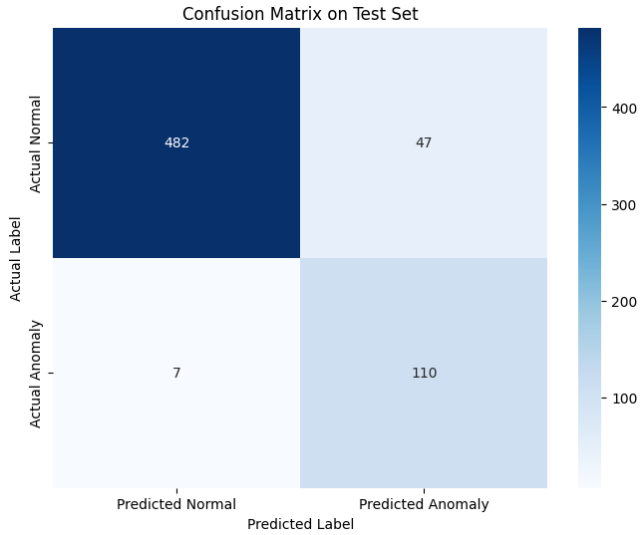

Figure 6: Confusion matrix showing the model's performance on the test set.

Table 2: Classification Report for Anomaly Detection

Class	Precision	Recall	F1-Score
Normal(0)	0.99	0.91	0.95
Anomaly(1)	0.70	0.94	0.80
Accuracy		0.92	

The most important result is the **recall for the Anomaly class, which is 0.94 (or 94%)**. In a medical context, this is the critical metric. It means our model successfully found 110 out of the 117 total anomalies in the test set. A high recall is essential because it shows the system is extremely unlikely to miss a dangerous cardiac event.

4.3 Error Analysis

While the model's overall performance was excellent, it was not perfect. We analyzed the few mistakes it made to better understand its limitations and the nature of the detection challenge. Figure 7 provides a visual comparison of a correctly identified normal signal, a correctly identified anomaly, and a missed anomaly.

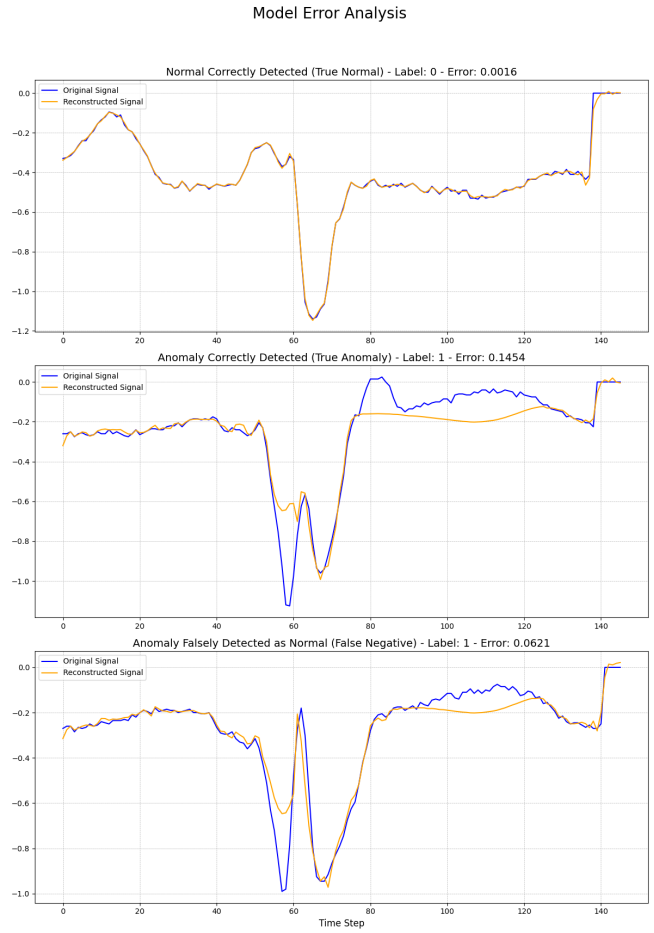

Figure 7: Visual analysis of model performance on different signal types.

- **False Negatives:** The model missed **7** anomalies, misclassifying them as normal. In these cases, the shape of the anomalous heartbeat closely resembles a normal signal. This results in a reconstruction error that is low enough to fall just below our detection threshold. These cases represent the most challenging forms of arrhythmia.
- **False Positives:** The model incorrectly flagged **47** normal signals as anomalies. This represents the necessary model trade-off for achieving a very high recall. To ensure the system rarely misses a critical event, we accepted a slightly higher number of false alarms.

5 Conclusion

In conclusion, this project successfully demonstrated the effectiveness of a semi-supervised 1D convolutional autoencoder for the detection of ECG anomalies. By training the model exclusively on normal ECG signals, it learnt a robust baseline of healthy heartbeats. Using a 90th percentile threshold on the reconstruction error, the system was able to detect both known and unknown anomalies with 94% recall on the test set. This high-sensitivity approach proved to be reliable and efficient for deployment on a portable monitoring device.