RhythmNet-TT: Adaptive Arrhythmia Detection Beyond Training

Alessandro Costanzo Ciano **Sahil Bavishi** **Matthew Pugh**

Abstract

Detecting arrhythmias in electrocardiogram (ECG) signals is crucial for early diagnosis and treatment. While Deep Learning models like CNNs, RNNs and Transformers improve automated ECG analysis, they struggle with real-world variability and long-term dependencies. This study presents RhythmNet-TT, a Transformer-based model with test-time learning via a memory module to enhance adaptability beyond training data. We compare RhythmNet-TT to the ECG-DETR baseline using MIT-BIH AFIB, MIT-BIH Arrhythmia, and PhysioNet 2017 datasets, evaluating performance across ECG time windows and memory configurations. Key questions investigated include: (1) How does RhythmNet-TT compare to ECG-DETR? (2) How do time windows and hyperparameters impact performance? (3) What is the runtime cost of test-time learning? (4) How well does it generalize? Results show RhythmNet-TT improves arrhythmia detection, especially for longer sequences, while handling out-of-distribution data. Despite computational costs, its adaptability supports early diagnosis, reduces human error, and enhances automated cardiac monitoring.

1. Introduction

It is well known that the heart is an important organ in the body, and that disease or breakdown can lead to fatal consequences. A commonly used indicator for such issues is detecting arrhythmias, which are heartbeats that are disrupted, and these can indicate serious health risks. Some arrhythmias are harmless, but others can lead to stroke, heart failure, or sudden cardiac arrest Cleland et al. (2002). Detecting these irregularities early is critical, yet traditional methods rely heavily on expert interpretation of electrocardiogram (ECG) signals. This process can be slow, labour-intensive, and prone to human error.

Machine Learning has transformed arrhythmia detection by automating ECG analysis. Deep Learning models can recognize complex patterns in heartbeat data, making diagnosis faster and more efficient Dessai (2017). State-of-the-art models, including Convolutional Neural Networks (CNNs) (R.S & Janani (2023), Ahmed et al. (2023)), Recurrent Neural Networks (RNNs) (Singh et al. (2018), Kuila et al. (2022)), and Transformers (Hu et al. (2022), Che et al. (2021)), have achieved high accuracy in identifying abnormal rhythms. However, they still face challenges when applied in real-world settings.

One issue is data variability. Most deep learning models assume that the data they encounter during deployment will be similar to their training data. In reality, ECG signals vary depending on sensor placement, patient physiology, and environmental factors. This mismatch can reduce accuracy and limit the model's reliability in clinical applications. Another challenge comes from the ability of current models to handle larger contexts of ECG recordings, called 'windows'. While longer windows could result in providing more information on the context of the heartbeats, many models struggle to process these larger sequences, leading to performance drops (Behrouz et al. (2024)).

Recent research has explored ways to address these limitations in related fields. Behrouz et al. (2024) introduced TITANs, a Transformer-based model with an adaptive learning mechanism that implements a 'memory block' that can learn how best to represent the data during model deployment/test time. This allows the model to adjust to shifting data distributions, improving its performance in real-world conditions without needing new training on data from those distributions.

Building on this idea, our study implements a model that we call 'RhythmNet-TT', inspired by the Memory as Context framework from Behrouz et al. (2024) to predict heartbeat locations and classify arrhythmias. To do this, we integrate our version of memory into the ECG-DETR - a Transformer-based model from Hu et al. (2022), using the unadapted ECG-DETR model as a baseline for our experiments. We then evaluate performance across varying window sizes, a range of hyper-parameter choices and potentially out-of-distribution data. By improving the adaptability and accuracy of arrhythmia detection models, this research aims to make ECG analysis more reliable and contribute to faster diagnoses.

Our work will then aim to answer 4 main questions:

- What is the performance of the RhythmNet-TT compared to the ECG-DETR model?
- How does varying the time window sizes and hyperparamters of the memory in RhythmNet-TT impact performance of the models?
- How does implementing the memory module affect run-time efficiency?
- How does the performance of the RhythmNet-TT compare to the ECG-DETR model on an out-of-distribution dataset?

The rest of the work will then be as follows. Section 2 discusses related work, with respect to work in arrhythmia detection, and the recent 'Titans' work. Section 3 discusses the datasets used for pre-training, fine-tuning and testing our models, leading into a discussion of the implementation of the baseline model and the adaptations done to create RhythmNet-TT in Section 4. A discussion of the experiment results will then be in Section 5, finishing with our conclusions in Section 6.

2. Related work

The field of electrocardiogram (ECG) analysis has been significantly transformed by advances in Machine Learning (ML) and Deep Learning (DL). While early approaches relied on classical ML, recent efforts featuring DL models are currently dominant in the field. This section reviews prior work in ECG-based arrhythmia detection and localisation, highlighting the contributions of DL architectures. The recently published Titans model, which we introduce in our methodology, will also be discussed.

2.1. Machine Learning and Deep Learning for ECG Analysis

Early work in ECG classification relied on traditional ML techniques, including support vector machines (SVMs), k-nearest neighbors (KNN), and decision trees (Boulif et al. (2023)). These models typically required extensive feature engineering, where handcrafted features such as RR intervals and wavelet transformations were extracted from ECG signals. However, the ability of these models to generalize was often limited by dataset-specific features and the variability of ECG patterns across different patients.

Deep Learning has since emerged as the dominant approach in ECG classification. Convolutional Neural Networks (CNNs) have demonstrated remarkable success due to their ability to automatically learn spatial patterns in ECG waveforms. Notably, the work by Hannun et al. (2019) introduced a 34-layer deep CNN trained on a proprietary dataset of over 90,000 ECG recordings. This model achieved cardiologist-level performance in arrhythmia classification.

Other studies have explored different neural architectures, including Recurrent Neural Networks (RNNs) (Singh et al. (2018), Kuila et al. (2022)) and Long Short-Term Memory (LSTM) networks (Yildirim et al. (2019)), to capture the sequential nature of ECG signals. For instance, Islam et al. (2022) proposed a hybrid Bidirectional RNN and Dilated CNN (BRDC) model, which improved classification performance by leveraging both long-term dependencies and high-resolution feature extraction.

Inspired by the success of Transformers in natural language processing, researchers have adapted self-attention mechanisms to ECG classification. These models have demonstrated better performance over CNN-based architectures, particularly in identifying patterns across longer ECG sequences and predicting with class imbalances. The paper from Hu et al. (2022) further applied a full Transformer encoder to ECG signals. Unlike CNNs, which focus on local feature extraction, Transformers effectively capture long-range dependencies within ECG sequences. This advantage led to significant improvements in classification performance and interpretability, demonstrating the advantage of Transformer-based methods over CNNs and motivating further research in this direction.

2.2. Titans Model

Our work builds upon the work of Hu et al. (2022) by integrating a Titans-inspired architecture (Behrouz et al. (2024)), a state-of-the-art model recently introduced by Google for time-series analysis. Titans extends Transformer architectures with novel mechanisms for memory retention, making it particularly well-suited for ECG classification and heartbeat localisation. Its primary contributions lie in allowing larger context sizes (a known limitation of Transformers) and enhanced performance at the cost of higher computational complexity at test time. Indeed, the main feature of Titans is mimicking the brain's behaviour by learning at test time, and adapting to new information.

3. Dataset and task

3.1. Datasets

Our experiments were run and measured using 3 datasets, each for a different task. These datasets include: MIT-BIH AFIB (Moody (1983)) for pre-training models, MIT-BIH Arrhythmia (Moody & Mark (2001)) for fine-tuning and evaluating the models, and PhysioNet 2017 (Clifford et al. (2017)) for testing final models on potentially out-of-distribution data. These datasets are described in their relevant sub-sections below.

3.1.1. MIT-BIH AFIB

The MIT-BIH AFIB dataset was created to train ML models on ECG data to detect arrhythmia episodes in long recordings of heartbeats. This data consists of varying lengths of ECG recordings, with annotations labelling the position and the classification of the heartbeats.

To begin, the data was resampled into 360Hz, to match the data in the fine-tuning dataset, and split into t-second windows, discarding the end of the recordings if the full recording could not be split into the exact windows. All windows containing 'Atrial Flutter' or 'AV Junctional Rhythm' beats were discarded, due to their low counts in the data (1.04% and 0.02% respectively).

Any recordings that had a heart rate of more than 120bpm also were discarded, as the Association for the Advancement of Medical Instrumentation (AAMI) MARK (1987) recommends to exclude records containing paced beats. To counter issues with the model outputting the same number of classifications, we introduced a *no-heartbeat* class to allow the models to also predict when there was no heartbeat, in the recordings with less heartbeats and fixed the number of target classifications per window to $t \times 2$.

The number of heartbeats in the data is therefore 1,077,136 with 56.39% attributed to Normal heartbeats, and the rest

being Atrial Fibrillation.

Once the windows were created, we split the dataset into training (70%), validation (15%) and test (15%) sets to monitor the performance of the models as they trained. The training set of ECG signals then received Z-score normalisation, applying the same function to normalise them to the rest of the data. The models were then trained on the training set, and the validation set was used to track of the models.

For the models to predict the positions of the heartbeats, a start and end of the heartbeats are needed. The method employed to achieve these uses 'splits' between the heartbeats to calculate these. We used the same method as Hu et al. (2022) for splitting the beats for repeatability:

$$Pos_{split_i} = \sigma P_i + (1 - \sigma) P_{i+1}$$

Where P_i refers to the ith annotated position of a heartbeat. Following reasoning from work by Clifford et al. (2006), Hu et al. reason that $\sigma = 0.4$. We follow this work by implementing the same method for differentiating beats across this dataset, and the MIT-BIH Arrhythmia dataset.

3.1.2. MIT-BIH Arrhythmia

After the models had been pre-trained, we used the MIT-BIH Arrhythmia dataset to better their performance on classifying multiple arrhythmia types, and used a validation and test set to monitor the performance of the models as they trained. This data includes 18 different classification labels of heartbeats, which can be separated into 4 classification labels according to Rexy et al. (2021), following recommendations from the AAMI MARK (1987).

Similarly to the pre-training dataset, we broke the data into the sizes of windows we were working on. Any windows that contained 'unknown' beats were discarded, along carrying out the same filtering done on the MIT-BIH AFIB dataset for the number of beats in a recording, and the *no-heartbeat* class was added.

We then split the data randomly into a train (70%), validation (15%), and test set (15%), and Z-score normalised the data using the mean and standard deviation of the training set. Table 1 summarises the breakdown of the final classifications in the data.

Table 1. Proportions of labelled heartbeats used from MIT BIH Arrhythmia

Notation	Name	Count	Proportion
N	Non-ectopic beats	91,955	91.08%
S	Supraventricular ectopic beats	2,712	2.69%
V	Ventricular ectopic beats	5,512	5.46%
F	Fusion beats	781	0.77%
		100,960	100%

3.1.3. PhysioNet 2017

The final dataset used was the MIT PhysioNet 2017 challenge dataset. This dataset includes various lengths of times of ECG recordings of patients, with one of 4 labels for each recording. To utilise this dataset, we resampled the data from 300Hz to 360Hz, split the data into windows again, and applied the same function that was used to normalise the ECG recordings. If the recordings did not divide into windows all of the same specified size, padding of 0s was added to the end of the ECG recording to achieve the correct size window. Any recordings labelled 'unclassified' were excluded, and so only the three remaining labels were used. This left 8,249 usable records, with 61.53%, 9.19%, and 29.28% being 'N' (Normal), 'A' (Atrial ectopic beats), and 'O' (Other arhythmia) respectively.

As these labels are different to the labels used in the MIT-BIH Arrhythmia dataset, we 'mapped' the outputs of the models (N,S,V, or F) to the 3 classifications. These were converted to (N,A,O,O) respectively, 'A' captures beats similar to 'S', and 'O' captures all other arrhythmias with 'N' being the same.

3.2. Task

To measure the performance on the pre-training and fine-tuning, accuracy, F1 score, precision and recall were taken. These metrics were chosen as accuracy is a good metric for the ability of the model to predict correctly, and F1 score, Precision, and Recall can give insights to how the model is performing on imbalanced data, such as the ECG data where a majority of heartbeats are 'Normal'. A high accuracy, for example, could be achieved by classifying everything as 'Normal', but this would not achieve a high F1 score.

To measure the performance of the models on the Physionet 2017 dataset, the models were run on each recording one at a time, and the predictions were taken of every heartbeat in the recording and summed. The proportion of the heartbeats in each recording made by the model was recorded, and used to compare the models.

4. Methodology

4.1. ECG-DETR

The ECG-DETR is a Transformer-based model implemented by Hu et al. (2022). We used this model as a 'baseline' to run model comparisons to, and so based the architecture off the paper. This model can be seen as split into 3 different main 'blocks'. These 'blocks' include a CNN-backbone structure, a Transformer structure, and finish with Feed-Forward Networks. Each block is discussed in a separate sub-section below.

4.1.1. CNN-Backbone

The CNN-Backbone consists of many parts called 'bottlenecks' itself, which are adapted from MobileNetV2 Sandler et al. (2018) with some adaptations. An adaptation to the bottlenecks includes an implementation of a Squeeze-and-

Excitation (SE) module Hu et al. (2018) to help improve representation of the channels.

One bottleneck includes, in order, a convolutional layer, depth-wise convolutional layer, convolutional layer and a squeeze-excitation module with a residual connection. All convolutional layers before the final convolutional layer end with a ReLu activation layer, and all end with batch normalisation Ioffe & Szegedy (2015). These can then repeat for a specified number of times, with different numbers of channels. This layout is shown in figure 3 in the Appendix made by Hu et al. (2022).

The backbone is then made up of layers of the bottlenecks, which reshape the structure of the data from 1 channel of size *window_size* to an ending result of d_{model} channels of size *window_size*/64, where d_{model} is a hyper-parameter to be chosen, and we leave fixed at 128. Table 5 in the Appendix produced by Hu et al. (2022) shows how the backbone is built from the bottlenecks themselves.

4.1.2. Transformer

Once passed through the CNN-Backbone, the next module is a Transformer architecture. It has been widely adopted and adapted since first introduced by Vaswani et al. (2017). The structure involves multiple heads (6 for our model), each containing an encoder block, and a decoder block, both making use of an attention mechanism. The input to these blocks are first concatenated with a 'positional encoding' to give the Transformer some information on when the features come in the input. Our adopted positional encoding follows that described in the original paper.

4.1.3. Feed-Forward Network

The final section of our model begins with a convolutional layer. The purpose of this layer is to 'reshape' the data into having the number of channels be the maximum number of heartbeats classifiable in a single recording, so has an out-channel size of $t \times 2$.

This then goes into 2 neural networks, each with a single hidden layer of 24 units (a parameter that can be changed) followed by batch normalisation. These networks produce soft-max predictions for classes of heartbeats, and start and end positions of the heartbeats.

4.2. RhythmNet-TT

Following the promising results of Behrouz et al. (2024) (see section 2), we introduced a similar architecture to the '*Memory as a Context*' (MAC) proposed, the most promising of the three approaches proposed. In the MAC design, a neural long-term memory module augments the current input with historical context, potentially extending the effective context length and improving predictions in different data distributions at test time.

While the original paper assigns a dedicated memory module to each attention head, our design uses a single one, applied across the entire Transformer block. This decision was motivated by two factors. First, consolidating memory interactions into a unified block might enhance stability during both training and test-time adaptation. Specialised per-head memories risk introducing conflicting parameter updates. Our implementation might mitigate oscillations caused by competing gradients. Second, practical deployment constraints (in clinical settings etc.) may require limits to computational and memory usage. Maintaining separate test-time-updated memories for each head would significantly increase latency and hardware requirements.

In our model, RhythmNet-TT enhances the Transformer encoder with two modules: a static *Persistent Memory* submodule to encode stable, task-specific knowledge, and a *Neural Memory* component that updates parameters dependent on new data. As the model processes each batch of windows of the ECG recording at test time, it uses both the current signal, and what it has 'remembered' in the neural/persistent memory, while the neural memory component is updated (via test-time training) to continuously capture new information.

4.2.1. Persistent Memory

The persistent memory is only implemented just before the input to the Transformer. This appends a series of values to the output of the previous layer, which the model can use to 'store information' that it may need. As these are just static values, during test-time inference, this does not change.

4.2.2. Neural Memory

The neural memory is implemented in 2 places in the MAC framework we adopted. The first implementation takes an input from the CNN-backbone, completes a forward pass, and then appends the output to the input for the Transformer. The second implementation is the one that implements the test-time training, and sits just after the Transformer block, but before the final Feed-Forward Network blocks.

Our Neural Memory is a Feed-Forward Network with a single hidden layer. The input undergoes an initial transformation, batch normalisation, and ReLU activation before passing through 128 hidden units and reducing back to the original size.

For forward passes, the RhythmNet-TT gives the Neural Memory block a linearly-transformed output of the previous layer at time t, called the 'queries' ($\mathbf{q}_t$) by Behrouz et al. For the learning process at test time, 2 more linear transforms of the previous layer are used: the 'keys' ($\mathbf{k}_t$), and the 'values' ($\mathbf{v}_t$).

The queries, keys, and values all come from applying a matrix associated with each to the previous layer. These matrices are not part of the neural memory, and have to stay fixed at test-time for the model to interpret the outputs. These are therefore only updated during training time.

4.2.3. Neural Memory 'Learning'

The learning process at test time is driven by a surprise metric and a forgetting mechanism, which enable the model to adapt to new ECG patterns while balancing retention of historical knowledge. The surprise metric, denoted S_t at time

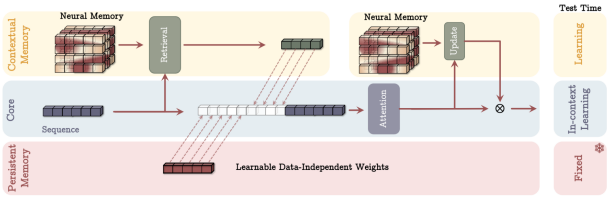

Figure 1. Memory as a Context (MAC) Architecture. Illustration of the specific configuration adopted for training at test time. Figure taken from Behrouz et al. (2024).

t, quantifies the unexpectedness of the current observation and is defined as:

$$S_t = \eta_t - \theta_t \nabla L(M_{t-1}; x_t)$$
$$L(M_{t-1}; x_t) = \|M_{t-1}(\mathbf{k}_t) - \mathbf{v}_t\|_2^2$$

The memory state M_t, which represents the parameters of the neural memory, is then updated using the forgetting mechanism:

$$M_t = (1 - \alpha) M_{t-1} + S_t$$

where α is a hyperparameter controlling the rate of memory decay—higher values of α accelerate forgetting, while lower values preserve more past context. The hyperparameters α, θ, and ν could potentially be adaptive and evolve with the data distribution; however, in our implementation, we fixed them to ensure stability and simplicity, a choice assumed in subsequent discussions.

We also included the 'test-time training' process of updating the state of the neural memory whilst training of the models on the training data, as the neural memory still has to be relied on in the training process for the inference stage, for the model to utilise it.

4.3. Experimental setup

In this section, we discuss the experimental setup, including the loss functions used, learning rate and how the pre-training and fine-tuning was conducted.

4.3.1. Pre-Training, Fine-Tuning and Testing

Pre-training is a technique in Deep Learning where a model is initially trained on a large dataset before being fine-tuned for a specific task. R.S & Janani (2023) demonstrates that the use of transfer learning with pre-trained ECG models can significantly improve classification accuracy while reducing computational costs and training time.

In our work, we used the MIT-BIH AFIB dataset to pre-train our models, due to the large amount of data available and the relevance to our task. To do this, we created the models to predict only 3 classes (see Section 3). We pre-trained the models for 200 epochs, as all the models had stopped making gains in the F1 score, accuracy and reduction of the loss by that point.

After pre-training, we fine-tune our models for 800 epochs using the MIT-BIH Arrhythmia dataset, which classifies them into four distinct heartbeat classes. Since there is a class mismatch between the pre-training and fine-tuning datasets, we reset the final classification layer and changed the output size to match the, now 5, classifications needed (see Section 3). This approach allows us to retain the pre-trained model's ability to represent ECG data but classify more in-depth classes of arrhythmia.

To assess the fine-tuned models' abilities on potentially out-of-distribution data, we conducted a final test using the PhysioNet 2017 dataset and collecting the number of heartbeats that were predicted for each record as proportions, and comparing these proportions shown in the results section.

4.3.2. Learning Rate Scheduling

For both pre-training and fine-tuning we follow the general approach of Hu et al. (2022), we utilize the Adam optimizer with a base learning rate of $\eta_{\text{base}} = 0.0001$. A 10-epoch warm-up strategy is implemented (Vaswani et al. (2017)), where the learning rate progressively increases from a low initial value to η_{base}.

Specifically, during the warm-up phase, the learning rate follows:

$$\eta_{\text{cur}} = \eta_{\text{base}} \left(0.01 + 0.99 \frac{T_{\text{cur}}}{10}\right), \quad T_{\text{cur}} \leq 10.$$

And subsequently decays according to:

$$\eta_{\text{cur}} = \frac{\eta_{\text{base}}}{2} \left(1 + \cos\left(\frac{T_{\text{cur}}}{T_{\max}} \pi\right)\right) \quad T_{\text{cur}} \geq 10.$$

Where T_{cur} is the current epoch, $T_{\max}$ the maximum training epochs.

4.3.3. Loss functions

As the task involves a prediction of the position and class of the heartbeats, multiple loss functions are required. We use the same loss functions for the models (excluding the neural memory) as used by Hu et al. (2022). This loss function contains 3 parts, 2 for encouraging accuracy of heartbeats, and 1 for the classifications.

The loss for the positions of the heartbeats uses L_1 norm loss, along with one-dimensional IoU loss (Rezatofighi et al. (2019)), defined as below given an output of Y_{out} and a target of Y_{tgt}:

$$L_{L_1}(Y_{out}, Y_{tgt}) = \|b_{out} - b_{tgt}\|$$
$$L_{IoU}(Y_{out}, Y_{tgt}) = 1 - \left(\frac{|b_{out} \cap b_{tgt}|}{|b_{out} \cup b_{tgt}|} - \frac{d(b_{out}, b_{tgt}) - |b_{out} \cup b_{tgt}|}{d(b_{out}, b_{tgt}) + \delta}\right)$$
$$d(b_{out}, b_{tgt}) = (max\{b_{out}\} - min\{b_{tgt}\})$$

Where b_{out} and b_{tgt} represent the boundary boxes of the output and the target respectively.

For the classification loss, focal loss (Lin et al. (2017)) is used due to the class imbalance seen in the all of the datasets, to compensate. This is calculated as follows:

$$L_{focal}(Y_{out}, Y_{tgt}) = -\theta_c (1 - p_c)^{\gamma} \log(p_c)$$

Where c is the class of the target, γ is the focusing parameter set to 2 to keep in line with the work done by Hu et al. (2022). The p_c value refers to the probability given by the output of the classifier. The θ_c parameter can be fine-tuned by class to prioritise the learning of specific classes, however for the purpose of the experiments we left these as the same.

Finally, these loss functions are added together with set weights to prioritise classifying heartbeats over getting the correct positions as the priority is classifying arrhythmias. To account for this, we used the same ratios Hu et al. used in their paper, 8:4:1 for focal:L1:IoU losses respectively. For any classifications of *no-heartbeat*, the positions were ignored for the loss functions, and so only the classification loss was calculated.

5. Results

For the purpose of discovering the effect of the neural memory hyper-parameters on the RhythmNet-TT model, we decided to implement 3 different variations based on the 'learning rate' of the memory. These are described as 'conservative', 'mid', and 'aggressive', as shown in Table 2. The parameters are described this way to reflect the change in the ability to learn - the most conservative model has lower α and θ values which directly lower the effect of new information changing the parameters of the neural memory, with the higher ν causing the model to value past learning more than the more aggressive models.

Table 2. Hyper-parameter Settings for Conservative, Mid, and Aggressive Runs

Parameter	Conservative	Mid	Aggressive
α	0.10	0.25	0.40
θ	0.30	0.50	0.70
ν	0.90	0.80	0.70

In order to investigate the effects of the time window changes, every dataset was split into 3 different time window settings: 3, 7, and 15 seconds. We anticipated that on longer sequences RhythmNet-TT would outperform the baseline model, thanks to improved long-term memory retention.

The 4 models: the ECG-DETR, and the conservative, mid and aggressive RhythmNet-TTs were all pre-trained for 200 epochs each, and then the models were taken from the best epochs to fine-tune on the MIT-BIH Arrhythmia dataset for 800 epochs. This epoch number was chosen to allow the models to give time to learn and not stop mid-learning.

5.1. Discussion of Results

Table 3 shows best epoch of the fine-tuning process, based on the lowest validation loss. Figure 2 presents the validation metrics for the baseline model and the best performing RhythmNet-TT configuration across the three window sizes over all the epochs.

5.1.1. ECG-DETR Performance

These results show that the ECG-DETR under-performed compared to its performance in the original paper (Hu et al. (2022)). This could be explained by differences in approach between the authors of the paper and the experiments run in this paper. The main differences were in the data given to the model: to more effectively mimic real-world data, we disregarded the timing of heartbeats within the windows, resulting in a significant portion of the frames likely containing only partial heartbeats. This is different to Hu et al.'s approach, where they made sure to include the entire heartbeat. This could cause the model to not get the full context of the heartbeat, however we argue that it is more accurate to real-world scenarios, as heartbeat locations will not be known.

Another difference is that the data was normalised differently - Hu et al. normalised each ECG record before splitting the data, which we felt could have caused some 'data leakage' where information from validation/test influence the training of the model. The normalisation technique we used is described in Section 3, which avoids this issue. Due to this difference, the ECG-DETR model may be receiving less information record through our method, and so may appear to lose some performance.

As hypothesised, the performance does decrease when given longer windows of ECG data to predict. We can also see that the drop-off in performance in the ECG-DETR is more extreme between 3 and 7 seconds than 7 to 15. This could be explained due to the imbalance in the data - obtaining about 50% F1, precision and recall is not too difficult for the models to do, but higher values require getting the infrequent arrhythmia classes correct more often. Little research has been conducted into the reasons behind the drop in performance on longer sequences for Transformers; however, a probable reason is that the self-attention block might limit the amount of things the model can 'remember'.

5.1.2. Performance of RhythmNet-TT

From Table 3, we can see that the RhythmNet-TT models either match or outperform the baseline ECG-DETR. We observe moderate improvements in accuracy, with the most substantial gains reflected in the F1 and Recall scores. These last two are particularly relevant for the points made in Section 3.2. Such metrics can be hard to increase, as they require ignoring the most common class and classifying heartbeats that are not so common (the arrhythmic beats). These improvements imply that the RhythmNet-TT is able to pick out arrhythmias better and avoid classifying most heartbeats as either *normal* or *no-heartbeat*.

Notably, Rhythmnet-TT with the Mid configuration on 7 second windows achieved the best overall scores. This supports the hypothesis that a longer context window could help with predicting arrhythmias, and that the memory implementations improve Transformers over longer windows. We report test performances of 85.97% F1 and 96.67% Accuracy against 64.57% F1 and 98.93% Accuracy of ECG-

Table 3. Experiment Results for all models trained on MIT-BIH Arrhythmia. The best performing model based on F1, precision and recall for each time window is in bold.

Window Size (seconds)	Model Type	Val Accuracy (%)	Val F1 Score (%)	Val Precision (%)	Val Recall (%)
3	ECG-DETR	98.80	66.81	69.32	65.92
	Cons-RNTT	**99.30**	**70.62**	**70.17**	**72.16**
	Mid-RNTT	99.44	66.96	67.80	67.39
	Agg-RNTT	99.28	59.34	59.07	60.27
7	ECG-DETR	98.17	50.79	54.80	50.04
	Cons-RNTT	98.17	54.46	59.13	52.85
	Mid-RNTT	**98.74**	**71.93**	**74.42**	**73.40**
	Agg-RNTT	98.28	65.24	67.00	65.44
15	ECG-DETR	97.84	51.50	54.37	51.08
	Cons-RNTT	97.86	53.09	59.71	51.15
	Mid-RNTT	97.84	55.54	58.06	54.42
	Agg-RNTT	**98.07**	**59.06**	**60.47**	**58.70**

DETR, also evaluated on the 7 second windows. These higher values suggest that the test-set contained easier targets for model-prediction, but still highlight the improvement of test-time learning.

For our model, the performance across different window sizes is tied to the strength of the Neural Memory learning process. Specifically, for the 3, 7, and 15 second windows, the best results were achieved by the Conservative, Mid, and Aggressive configurations respectively. This suggests that as the window length increases, a more adaptable memory mechanism becomes necessary to retain critical information over time.

As shown in Fig. 2, RhythmNet-TT initially struggles to pick up learning compared to the baseline. However, while ECG-DETR tends to plateau early, RhythmNet-TT continues to show signs of improvement even in the final epochs of fine-tuning. Notably, its learning curve is less stable, occasionally displaying sudden drops that, in most cases, recover without severely affecting overall performance. An exception is observed for the Aggressive configuration, where a sharp decline in performance persisted for multiple epochs, suggesting a more prolonged impact on the learning process. We hypothesize that these instabilities stem from the Neural Memory updates, triggered by encountering rare sequences that disrupt learned patterns. This effect is particularly pronounced in the Aggressive configuration, where larger updates lead to more drastic fluctuations. Despite this, the longer sequences benefited from relying more on these short-term memory adjustments.

5.1.3. RhythmNet-TT Test-time Efficiency

In order to test the efficiency of RhythmNet-TT compared to the ECG-DETR model, we ran both models on the same CPU and completed 1,000 records of forward passes for all of the window-sizes on the same records from PhysioNet 2017. We discovered that the increase in time taken was 175%, 197% and 193% for 3,7, and 15 seconds respectively.

This shows that the time for the RhythmNet-TT model more than doubles to run a forward inference on the data, and increases with respect to the sequence length. This is due to the larger size of the model and the 'learning process' that occurs under prediction. It is worth keeping in mind that the improvements made by the model must be offset against the increase in time for inference.

5.1.4. Performance on PhysioNet 2017

Finally, we picked the best performing RhythmNet-TT model, and the best performing ECG-DETR to compare on the pre-processed Physionet dataset, to compare the models on data that could be seen as 'out-of-distribution', as it was collected several years after the other datasets. We picked the 7 second Mid RhythmNet-TT, as it had the highest F1, precision and recall scores and we picked the 3 second ECG-DETR as this model had the highest scores out of the baselines. Table 4 shows the results.

Table 4. Proportion of predictions of each class on PhysioNet 2017

Model	True Type	'N' (%)	'A' (%)	'O' (%)
7s Mid RNTT	'N'	**96.95**	0.29	2.76
	'A'	94.50	**0.53**	4.97
	'O'	93.70	0.32	**5.98**
3s ECG-DETR	'N'	**94.39**	3.59	2.02
	'A'	89.30	**6.00**	4.70
	'O'	90.82	5.02	**4.16**

These results show that both models predict an increase in 'A' and 'O' beats for records with the 'A' and 'O' classes. They also show that both models predict more 'N' beats for the normal records. The main difference that can be observed is that the ECG-DETR appears to classify more 'A' arrhythmia beats than RhythmNet-TT. The table also shows that Mid-RNTT on 7 seconds achieves more 'O' and 'N' clases in the correct records than ECG-DETR, but is worst at classifying 'A'. This leaves support of the hypothesis that the model works better on out-of-distribution data ambiguous, with further research potentially needed.

A final note of interest is that, when tested on un-normalised

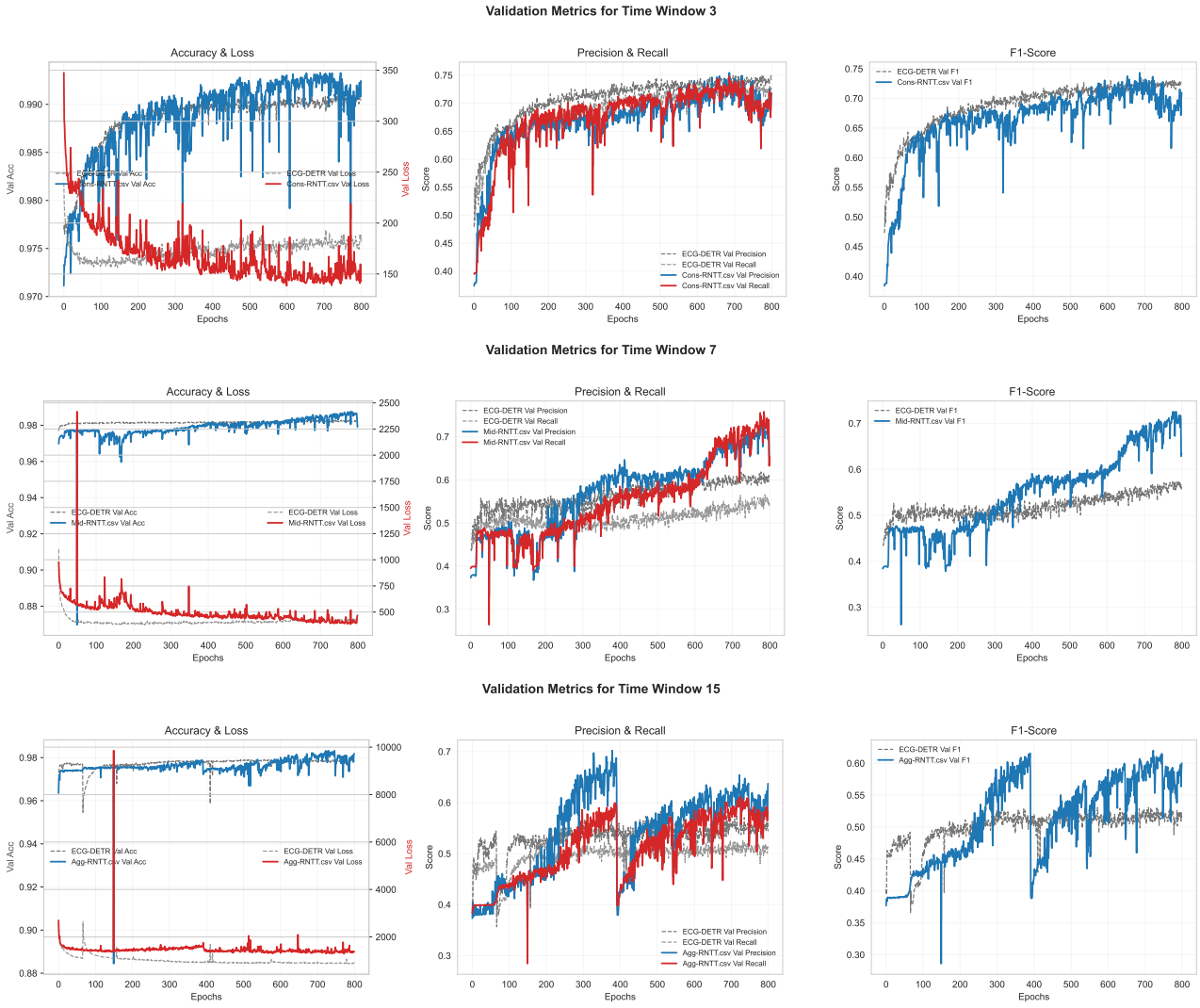

Figure 2. ECG-DETR v/s the Best RhythmNet-TT for all window sizes

data, RhythmNet-TT does not degrade in predictions of arrhythmias, however ECG-DETR does and predicts higher majority of 'N' beats. This suggests that RhythmNet-TT may perform well on out-of-distribution data, when coming from non-normalised sources.

6. Conclusion

In conclusion, we have successfully implemented RhythmNet-TT, a novel approach to arrhythmia detection, and demonstrated its advantages over state-of-the-art models. This paper has shown that implementing the ability to learn at test-time into an ECG Transformer-based model significantly improves the performance metrics such as F1, precision, and recall. We showed that the model improved over larger window sizes, whereas other Transformer-based models lose performance. Whilst computational costs of the model are higher than other state-of-the-art models, this has potential to improve over time, as evidenced by the high test scores, and ability to work on non-normalised data.

The ability to process longer sequence lengths for arrhythmia prediction could have an significant impact in the field, enhancing accuracy and reliability. Moreover, RhythmNet-TT shows the potential of Titan's test time learning or simpler, less computationally expensive, approaches to empower doctors and patients alike by enabling early detection, reducing human error, and ultimately saving lives through an automated and more precise diagnosis process.

Future work could explore enhancing RhythmNet-TT by implementing a more complex Neural Memory, as proposed by Google, incorporating normalization, gating, and 1D convolutions for keys, queries, and values (Behrouz et al. (2024)). Including separate Neural Memories for each attention head should also be explored, whilst keeping track of computational drawbacks. These might benefit performance considerably, stabilizing the learning process. Additionally, developing a preprocessing model to ensure windows contain complete heartbeats could improve performance, potentially outperforming the 'state of the art' results, originally obtained by Hu et al. (2018).

References

Ahmed, Adel A, Ali, Waleed, Abdullah, Talal AA, and Malebary, Sharaf J. Classifying cardiac arrhythmia from ecg signal using 1d cnn deep learning model. *Mathematics*, 11(3):562, 2023.

Behrouz, Ali, Zhong, Peilin, and Mirrokni, Vahab. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

Boulif, Abir, Ananou, Bouchra, Ouladsine, Mustapha, and Delliaux, Stéphane. A literature review: ecg-based models for arrhythmia diagnosis using artificial intelligence techniques. *Bioinformatics and Biology Insights*, 17: 11779322221149600, 2023.

Che, Chao, Zhang, Peiliang, Zhu, Min, Qu, Yue, and Jin, Bo. Constrained transformer network for ecg signal processing and arrhythmia classification. *BMC Medical Informatics and Decision Making*, 21(1):184, 2021.

Cleland, John GF, Chattopadhyay, Sudipta, Khand, Aleem, Houghton, Timothy, and Kaye, Gerald C. Prevalence and incidence of arrhythmias and sudden death in heart failure. *Heart failure reviews*, 7:229–242, 2002.

Clifford, Gari D, Azuaje, Francisco, McSharry, Patrick, et al. *Advanced methods and tools for ECG data analysis*, volume 10. Artech house Boston, 2006.

Clifford, Gari D, Liu, Chengyu, Moody, Benjamin, Li-wei, H Lehman, Silva, Ikaro, Li, Qiao, Johnson, AE, and Mark, Roger G. Af classification from a short single lead ecg recording: The physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology (CinC)*, pp. 1–4. IEEE, 2017.

Dessai, Vishakha S Naik. Review on arrhythmia detection using signal processing. *International Journal of Electrical and Electronics Engineers*, 9(01):1573–1579, 2017.

Hannun, AY, Rajpurkar, P, Haghpanahi, M, et al. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65–69, 2019. doi: 10.1038/s41591-018-0268-3. [Published correction appears in Nat Med. 2019 Mar;25(3):530. doi: 10.1038/s41591-019-0359-9].

Hu, Jie, Shen, Li, and Sun, Gang. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

Hu, Rui, Chen, Jie, and Zhou, Li. A transformer-based deep neural network for arrhythmia detection using continuous ecg signals. *Computers in Biology and Medicine*, 144:105325, 2022. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2022.105325. URL https://www.sciencedirect.com/science/article/pii/S0010482522001172.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Islam, Md Shofiqul, Islam, Md Nahidul, Hashim, Noramiza, Rashid, Mamunur, Bari, Bifta Sama, and Farid, Fahmid Al. New hybrid deep learning approach using bigru-bilstm and multilayered dilated cnn to detect arrhythmia. *IEEE Access*, 10:58081–58096, 2022. doi: 10.1109/ACCESS.2022.3178710.

Kuila, Sumanta, Dhanda, Namrata, and Joardar, Subhankar. Ecg signal classification and arrhythmia detection using elm-rnn. *Multimedia Tools and Applications*, 81(18): 25233–25249, 2022.

Lin, Tsung-Yi, Goyal, Priya, Girshick, Ross, He, Kaiming, and Dollár, Piotr. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

MARK, R. Aami-recommended practice : Testing and reporting performance results of ventricular arrhythmia detection algorithms. *Association for the Advancement of Medical Instrumentation, Arrhythmia Monitoring Subcommittee, AAMI ECAR, 1987*, 1987. URL https://cir.nii.ac.jp/crid/1571698600628450176.

Moody, George. A new method for detecting atrial fibrillation using rr intervals. *Proc. Comput. Cardiol.*, 10: 227–230, 1983.

Moody, George B and Mark, Roger G. The impact of the mit-bih arrhythmia database. *IEEE engineering in medicine and biology magazine*, 20(3):45–50, 2001.

Rexy, J, Velmani, P, and Rajakumar, T. Heart beat classification in mit-bih arrhythmia ecg dataset using double layer bi-lstm model. *International Journal of Mechanical Engineering*, 6:337–344, 2021.

Rezatofighi, Hamid, Tsoi, Nathan, Gwak, JunYoung, Sadeghian, Amir, Reid, Ian, and Savarese, Silvio. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 658–666, 2019.

R.S, Dr.Sabeenian and Janani, KK. Transfer learning-based electrocardiogram classification using wavelet scattered features. *Biomedical and Biotechnology Research Journal (BBRJ)*, 7:52, 01 2023. doi: 10.4103/bbrj.bbrj_341_22.

Sandler, Mark, Howard, Andrew, Zhu, Menglong, Zhmoginov, Andrey, and Chen, Liang-Chieh. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

Singh, Shraddha, Pandey, Saroj Kumar, Pawar, Urja, and Janghel, Rekh Ram. Classification of ecg arrhythmia using recurrent neural networks. *Procedia computer science*, 132:1290–1297, 2018.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, and Polosukhin, Illia. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yildirim, Ozal, Baloglu, Ulas Baran, Tan, Ru-San, Ciaccio, Edward J, and Acharya, U Rajendra. A new approach for arrhythmia classification using deep coded features and lstm networks. *Computer methods and programs in biomedicine*, 176:121–133, 2019.

7. APPENDIX

Configuration of backbone as shown in Hu et al. (2022)

Input	Operator	k	c	n	s
window_size x 1	conv1d	-	1	1	2
$\frac{window\_size}{2}$ x 1	bottleneck	6	8	1	2
$\frac{window\_size}{4}$ x [8ϕ]	bottleneck	6	16	2	2
$\frac{window\_size}{8}$ x [16ϕ]	bottleneck	6	32	2	2
$\frac{window\_size}{16}$ x [32ϕ]	bottleneck	6	64	2	2
$\frac{window\_size}{32}$ x [64ϕ]	bottleneck	6	128	2	2
$\frac{window\_size}{64}$ x [128ϕ]	bottleneck	6	256	1	1
$\frac{window\_size}{64}$ x [256ϕ]	conv1d	-	d_{model}	1	1

Table 5. The symbol n describes the number of layers in a sequence. All layers within the same sequence have the same output channels ($c\phi$). Only the first layer of each sequence uses stride of s, other layers use a stride of 1. The input features are expanded by k times before the depth-wise convolution. [] indicates rounding.

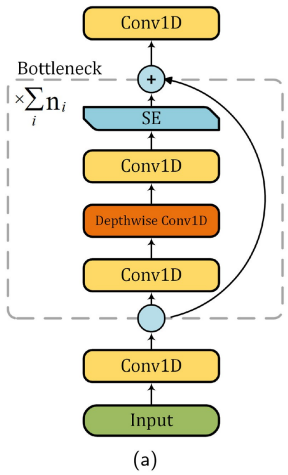

Figure 3. Figure showing the layout of the backbone as shown in Hu et al. (2022)