One-Dimensional ConvNeXt for Chagas Disease Screening from 12-Lead ECGs

Elaine Li[1]

[1]College of Arts and Sciences, Emory University, Atlanta, United States of America

Abstract

As part of the George B. Moody PhysioNet Challenge 2025, our team ECG_ChD_25 developed an open-source deep neural network (DNN) classifier based on 1D ConvNeXt model (~16.9M learnable parameters) to detect potential Chagas cases from standard 12-lead ECGs. Training data included three labeled ECG datasets: SaMi-Trop (Chagas positive), PTB-XL (Chagas negative) and a Chagas negative subset of CODE-15% (self-reported). To address highly imbalanced training data and label noise, we combined Focal loss with a pairwise ranking (hinge) loss, applied class-weighting, and oversampled minority class to 50% per epoch. We trained the model with AdamW for 12 epochs (batch size 64, initial learning rate (LR) 2e-4, weight_decay 1e-3, dropout rate 0.3) and maintained an exponential moving average (EMA) of the weights (decay = 0.999, BN buffer decay = 0.9). We used cosine annealing LR scheduler (T_max = 12 epochs, minimum LR = 5e-6) to enable more effective convergence. A 5-fold cross validation was employed, and the final model was chosen as a logit-averaged ensemble of the best-performing checkpoint from each fold. Our classifier achieved a Challenge Score (Recall@5%) of 0.200 on the hidden test set. This study demonstrates the potential of deep learning for Chagas screening with limited serological testing capacity.

1. Introduction

The George B. Moody PhysioNet Challenge 2025 invited teams to develop open-source algorithms for detecting Chagas disease from standard 12-lead electrocardiograms (ECGs) [1-3]. Chagas disease is a parasitic disease in Central and South America that affects ~6.5 million people and causes nearly 10,000 deaths annually. Timely treatment may prevent or slow damage to the cardiovascular system, but serological testing capacity is limited, so screening through ECGs can help to identify Chagas patients for treatment.

For this challenge, our team developed a deep learning algorithm based on a one-dimensional (1D) ConvNeXt model adapted for 12-lead ECG signals [4,5]. The model was trained on a highly imbalanced dataset with label noise, including 1,631 Chagas positive SaMi-Trop data [6], 21,000 Chagas negative PTB-XL data from a German population [7], and 28,569 Chagas negative data selected from the CODE-15% dataset with label noise [8]. Our approach integrates advanced ECG data pre-processing, Focal and pairwise ranking hinge losses, class balancing, and model ensembling to address the highly imbalanced and noisy nature of the available training datasets.

2. Method

2.1. Data Description

We utilized three publicly available 12-lead ECG datasets for training (**Table 1**):

1. SaMi-Trop: 1,631 data labeled Chagas positive validated by serological tests.
2. PTB-XL: 21,799 data labeled Chagas negative, from European patients outside endemic regions.
3. CODE-15%: > 300,000 data with self-reported Chagas status (~2% positives).

We limited the training set to 51,200 records: 1,631 from SaMi-Trop, 21,000 from PTB-XL, and 28,569 negative from CODE-15%, to stay within compute limits while ensuring a Chagas prevalence ≤5%. Labels from CODE-15% negatives were treated as weak, with label smoothing applied. We excluded positives from CODE-15% because the Chagas prevalence is typically 0.5-6% [9], so the mislabels among the self-reported negatives should be less than 4% while the percent of mislabels in the positives can be high. During internal development, we used 1,000 SaMi-Trop, 19,000 PTB-XL and 31,200 CODE-15% Chagas negative records as the training set and reserved 631 SaMi-Trop, 2,000 PTB-XL and 5,262 CODE-15 records as holdout for internal validation.

We used weighted random oversampling with replacement (PyTorch `WeightedRandomSampler`) in training so that the minority class constituted 10% of the samples drawn in each epoch. After our final submission, we found that oversampling SaMi-Trop to 50% further improved model performance by 13% (Challenge Score 0.339 → 0.384) on the internal validation set. We split the training dataset using 5-fold stratified cross-validation, and class distribution preserved across folds.

Table 1: Training Datasets

Dataset	Number of Samples	Chagas Positive	Chagas Negative	Label Quality
SaMi-Trop	1,631 (all used)	1631	0	Strong (Target population)
PTB-XL	21,799 (21,000 used)	0	21799 (21,000 used)	Strong (Different population)
CODE-15%	343,425 (28,569 used)	6561 (not used)	336,864 (28,569 used)	Weak (Self-reported, Target population)

The official validation and test data and labels were privately sequestered by the Challenge organizers to allow for the principled assessment of the submitted algorithms [1].

2.2. Data Preprocessing Pipeline

ECG data pre-processing is essential for achieving good generalization performance. Each ECG signal in WFDB format was:

1. Resampled to 400 Hz using polyphase filtering.
2. Zero padded or cropped to a uniform length of 4096 samples.
3. Filtered: Notch filters at 60 Hz (Brazil) and 50 Hz (Germany) for powerline interference removal; Wavelet-based baseline wander removal (Symlet-8 wavelet, level 8); 4th-order Butterworth bandpass filter (0.5–59 Hz); Edge effects removal: the first and last 0.25 s were zeroed to mitigate edge artifacts.
4. Normalized using per-lead *z*-score normalization.
5. Augmented (training only): random crop, amplitude scaling, Gaussian noise addition, time shift, lead dropout, and optional baseline wander injection.

2.3. 1D ConvNeXt Architecture

Our proposed DNN model is a custom 1D ConvNeXt deep convolutional neural network (CNN) [4,5] adapted for ECG time-series classification. The model processes 12-lead ECG inputs of length 4096 (resampled to 400 Hz), represented as tensors of shape (12, 4096). We chose this architecture based on the state-of-the-art (SOTA) performances of existing DNN models and our available compute (MacBook Pro M4 Max with 36GB memory). As our initial candidates, we selected ResNet, ResNeXt, EfficientNet, ConvNeXt and ConvNeXt V2 based on the performances of pretrained PyTorch TorchVision models tested on ImageNet-1K dataset [10]. We also tested several DNN models for time-series such as InceptionTime [11] and TimesNet [12]. Our tests showed that 1D ConvNeXt gave the best performance with reasonable training time (~6 hours on a MacBook Pro M4 Max, 36GB RAM).

The 1D ConvNeXt architecture begins with a 'patchify' stem layer, a non-overlapping 1D convolution (kernel size=3, stride=3) that downsamples the ECG sequence by 4× and increases the channel dimension, followed by BatchNorm (BN). The backbone consists of four stages with channel dimensions C = (64, 128, 256, 512) and block depths B = (3, 3, 18, 3). Each ConvNeXt Block contains a depthwise 1D, BN, pointwise projection expanding channels ×4 (inverted bottleneck), GELU activation, second pointwise projection to original dimension, and residual connection with DropPath. We found that LayerNorm (original ConvNeXt) and Global Response Normalization (ConvNeXt v2) led to worse model performance when the batch size was 64. Therefore, we used BN throughout the model.

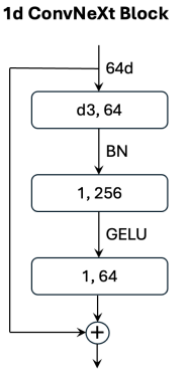


Figure 1. Building block of the custom 1D ConvNeXt. BatchNorm was used instead of LayerNorm because we found it gave better performance (batch size = 64). The model is similar to the original ConvNeXt-S in [4] but with smaller C = (64, 128, 256, 512) and B = (3, 3, 18, 3).

Downsampling layers (stride-2 conv) are placed between stages. After the final stage, a BN layer and global average pooling compress the temporal dimension. A dropout (p=0.3) precedes the classification head, a two-layer MLP (Linear → GELU → Dropout → Linear) that outputs logits for binary classification.

This configuration results in ~16.9 M learnable parameters. While larger than lightweight ECG CNNs, it remains tractable for training with ~50k 12 lead ECGs on a personal laptop and leverages the design strengths of ConvNeXt in handling long 1D biomedical signals.

2.4 Training Details

We used a stratified 5-fold cross-validation to train the model for 12 epochs per fold with the following hyperparameters determined by manual tuning (**Table 2**):

Table 2. Hyperparameters

Hyperparameter	Value
Optimizer	AdamW
Loss Function	Focal Loss (γ=2.0) + Pair-wise Ranking Loss (margin = 1.0, rank_loss_weight = 0.1, top negative percentage: 20%)
Epochs	12
Batch Size	64
Initial Learning Rate (LR)	2e-4
Weight-decay	1e-3
Dropout rate	0.3
Learning Rate Scheduler	Cosine Annealing (LR_min = 5e-6)
Chagas Positive Oversampling	50% per epoch
Class Weights [Non-Chagas, Chagas]	[1.0, 9.0]
Label Smoothing (CODE-15% only)	0.02
EMA decay	0.999 (BN buffers decay = 0.9)

Optimizer: AdamW (initial LR = 2e-4, weight decay = 1e-3).

Scheduler: Cosine Annealing (T_max = 12 epochs, η_min = 5e-6).

Loss function: Custom focal loss [13] with class weights (1.0, 9.0) and label smoothing (0.02) [14] on CODE-15% negatives:

$$FL(y,p) = -\alpha(1-p_t)^{\gamma}\log(p_t)$$

where, α is the class weight (1.0 for non-Chagas, 9.0 for Chagas), γ = 2.0 is the Focal loss parameter, $p_t = p$ if label $y = 1$ and $p_t = 1 - p$ if $y = 0$, p is the model predicted probability for Chagas.

An additional pairwise ranking (hinge) loss [15] is calculated during training using model output logits for positives and top 20% hardest negatives (margin m= 1.0):

$$Rank(logit^{+}, logit^{-}) = \max(0, m - (logit^{+} - logit^{-}))$$

where $logit^{+}$ is the model output logit for a positive sample, $logit^{-}$ is the model output logit for a negative, m is the margin = 1.0. The fraction of negatives used to calculate ranking loss is the top 20%.

For each mini-batch, the total loss during training is:

$$\mathcal{L} = \mathbb{E}[FL(y,p)] + 0.1\mathbb{E}_{(P,N)}[Rank(logit^{+}, logit^{-})].$$

Note that pairwise ranking loss was not used for checkpoint selection during the validation step.

Oversampling: positives oversampled to 50% per epoch via a weighted random sampler with replacement (PyTorch `WeightedRandomSampler`).

Gradient clipping: max norm = 5.0 after training loop.

Exponential Moving Average (EMA) [16]: EMA of weights improves model stability and generalization by maintaining a copy of the model's weights, updated as a decaying average of the training weights over time This smoothed trajectory helps in achieving better performance on unseen data, especially with noisy or small datasets. EMA with decay of 0.999 was applied to the model weights, and for BN buffers a 0.9 smaller decay was used.

We initialized all convolutional and fully connected weights with a zero-mean truncated normal distribution (PyTorch `trunc_normal_`, mean = 0, std = 0.02), and set all corresponding biases to 0.

For each fold, we select the checkpoint with the highest Challenge Score on that fold's validation set. The final model is an ensemble of the best-performing checkpoint from each fold (logit averaging).

2.5 Computational Environment

We implemented our model in Python 3.10.1 using PyTorch 2.3.0 and scikit-learn 1.6.0. The training was done on a MacBook Pro M4 Max (14-core CPU, 32-core GPU, 36GB RAM, PyTorch MPS backend). We used FP32 (full precision) training without mixed precision.

3. Results

Table 3. Official Challenge Score *(Recall@5%)* for our final selected entry (team *ECG_ChD_25).*

Training	Validation	Test	Ranking
0.34±0.02	0.372	0.200	N/A

We used 5-fold cross validation on the public training set (three different random seeds), repeated scoring on the hidden validation set, and one-time scoring on the hidden test set.

4. Discussion

Our results show that a ConvNeXt-based 1D deep

convolutional neural network can capture subtle ECG patterns associated with Chagas disease and identify up to 20% of Chagas cases among the top 5% predicted probabilities. Several key strategies contributed to the performance. Oversampling and class weighting helped address extreme class imbalance (< 2% positives). The label smoothing [14] on CODE-15% negatives made the model more robust to label noise. Our custom loss design, that combined focal loss [13] which down-weights easy and up-weights hard samples, with a pairwise ranking (hinge) loss [15] which pushes the score of each positive above the score of confusing negatives by a margin, boosted the Challenge Score. EMA [16] improved model stability and generalization. Lastly, ensembling the best performing checkpoints from five training folds improved model generalization performance.

While effective, our approach has limitations. The performance drop from validation (0.372) to test (0.200) is likely from overfitting to the training dataset. Our compute budget limited the model at ~16.9 M learnable parameters—a small size that likely limited performance. In addition, more advanced approaches such as transfer learning with pretrained larger models, self-supervised pretraining on large unlabeled ECG followed by fine tuning, and co-teaching would likely provide better performances.

Finally, it remains an important and open challenge to understand what the DNN model learned, and what learned features are Chagas specific or due to dataset bias.

5. Conclusion

We present an open-source 1D ConvNeXt classifier for Chagas detection from standard 12-lead ECGs, achieving a challenge score (Recall@5%) of 0.200 on the hidden test set. Our approach demonstrates the feasibility of ECG-based deep learning models for low-cost, scalable Chagas screening, helping limited serological testing capacity. Future work includes integrating multi-task learning (cardiac abnormality + Chagas detection), self-supervised pretraining on larger ECG sets, and deploying lightweight architectures for mobile use.

Code Availability: Source code is available at: https://github.com/emoryeli/ECG_ChD_25.git under the BSD 3-Clause license.

Acknowledgments

I want to thank open-source contributors from previous PhysioNet challenges for the ideas and inspiration.

References

[1] M. A. Reyna et al., "Detection of Chagas Disease from the ECG: The George B. Moody PhysioNet Challenge 2025," *Comput. Cardiol.*, vol. 52, pp. 1–4, 2025.

[2] M. A. Reyna et al., "Detection of Chagas Disease from the ECG: The George B. Moody PhysioNet Challenge 2025," *arXiv preprint* arXiv:2510.02202, 2025, doi: 10.48550/arXiv.2510.02202.

[3] A. L. Goldberger et al., "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[4] Z. Liu et al., "A ConvNet for the 2020s," *arXiv preprint* arXiv:2201.03545, 2022, doi: 10.48550/arXiv.2201.03545.

[5] S. Woo, X. Huang, C. Fowlkes, and T. Darrell, "ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders," *arXiv preprint* arXiv:2301.00808, 2023, doi: 10.48550/arXiv.2301.00808.

[6] C. S. Cardoso et al., "Longitudinal Study of Patients with Chronic Chagas Cardiomyopathy in Brazil (SaMi-Trop project): A Cohort Profile," *BMJ Open*, vol. 6, no. 5, p. e011181, May 2016, doi: 10.1136/bmjopen-2016-011181.

[7] P. Wagner et al., "PTB-XL, a Large Publicly Available Electrocardiography Dataset," *Sci. Data*, vol. 7, no. 1, p. 154, May 2020, doi: 10.1038/s41597-020-0495-6.

[8] A. H. Ribeiro et al., "Automatic Diagnosis of the 12-Lead ECG using a Deep Neural Network," *Nat. Commun.*, vol. 11, no. 1, p. 1760, Apr. 2020, doi: 10.1038/s41467-020-15432-4.

[9] Z. M. Cucunubá et al., "The Epidemiology of Chagas Disease in the Americas," *Lancet Reg. Health Am.*, vol. 37, p. 100881, 2024, doi: 10.1016/j.lana.2024.100881.

[10] *PyTorch Documentation*, "Torchvision Models." [Online]. https://docs.pytorch.org/vision/stable/models.html

[11] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Muller, "InceptionTime: Finding AlexNet for Time Series Classification," *Data Min. Knowl. Discov.*, vol. 34, no. 6, pp. 1936–1962, 2020, doi: 10.1007/s10618-020-00710-y.

[12] H. Wu, T. Hu, J. Liu, W. Lin, J. Wang, and M. Long, "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," *arXiv preprint* arXiv:2210.02186, 2023, doi: 10.48550/arXiv.2210.02186.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," *arXiv preprint* arXiv:1708.02002, 2018, doi: 10.48550/arXiv.1708.02002.

[14] R. Müller, S. Kornblith, and G. Hinton, "When does Label Smoothing Help?," *arXiv preprint* arXiv:1906.02629, 2020, doi: 10.48550/arXiv.1906.02629.

[15] T. Joachims, "Optimizing Search Engines using Clickthrough Data," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD '02)*, New York, NY, USA, Jul. 2002, pp. 133–142, doi: 10.1145/775047.775067.

[16] B. T. Polyak and A. B. Juditsky, "Acceleration of Stochastic Approximation by Averaging," *SIAM J. Control Optim.*, vol. 30, no. 4, pp. 838–855, 1992, doi: 10.1137/0330046.

Address for correspondence:

Elaine Li
College of Arts and Sciences, Emory University, Atlanta, GA
elaine.li@emory.edu