A Performance Study of 1D CNN Variants for 12-Lead ECG Classification and Edge Inference on NVIDIA Jetson Orin Nano

Najwa Muthia Aziz[1], Athiya Jannati Zumaro[1], Muhammad Nadhil Luthfirrahman[1], Kasyful Amron[1], Dany Primanita Kartikasari[1], Rizal Setya Perdana[1], and Eko Sakti Pramukantoro[1]*

Faculty of Computer Science, Universitas Brawijaya, Malang, Indonesia
ekosakti@ub.ac.id

Abstract. Most prior electrocardiogram (ECG) deep-learning studies focus on classification performance and are evaluated on server-class GPUs, leaving the accuracy-efficiency trade-off on edge devices under-characterized. This study compares four 1D Convolutional Neural Network (1D-CNN) variants—baseline 1D-CNN, 1D-ResNet, Multi-Scale CNN, and Attention-Based CNN—for diagnostic 12-lead ECG classification on the PTB-XL dataset at 100 Hz. Each complete 10-second, 12-lead recording is bandpass-filtered, Z-score normalized per channel, and fed directly as a (1000, 12) tensor to classify five diagnostic superclasses (NORM, MI, STTC, CD, HYP), without segmentation. Models are trained in TensorFlow/Keras and converted to TensorRT FP16 for batch inference on the NVIDIA Jetson Orin Nano. On the held-out fold-10 test set, 1D-ResNet attains the highest accuracy (79.94%) and Macro AUC (0.9354), while the baseline 1D-CNN delivers the lowest latency (0.5291 ms per sample) and smallest parameter count. All four models achieve sub-millisecond latency within the device's 8 GB memory envelope. Model selection uses the Weighted Sum Model (WSM), with weights of 0.40 accuracy, 0.30 latency, 0.15 CPU, 0.10 GPU, and 0.05 RAM, verified via the Analytic Hierarchy Process (AHP). 1D-ResNet is identified as the optimal model under both schemes (WSM 0.600; AHP 0.641), offering the best balance between performance and efficiency for accuracy-priority edge deployment.

Keywords: Electrocardiogram · 1D-CNN · Edge Computing · PTB-XL

1 Introduction

Cardiovascular disease remains a leading cause of global mortality, accounting for approximately 19.8 million deaths, or 32% of all deaths worldwide [1]. Electrocardiogram (ECG) analysis is the primary non-invasive diagnostic tool for detecting cardiac abnormalities, yet manual interpretation is time-consuming,

© The Author(s) 2026
M. Murtadha et al. (eds.), *Proceedings of the 2026 11th International Conference on Sustainable Information Engineering and Technology (SIET 2026)*, Atlantis Highlights in Computer Sciences 26,
https://doi.org/10.2991/978-94-6239-828-3_27

requires clinical expertise, and is prone to inter-observer variability. Automated ECG interpretation via deep learning offers a scalable solution, but its practical deployment must balance classification performance against the computational constraints of the target hardware. Hardware is normally fixed during requirements engineering to meet an application's needs; however, point-of-care ECG screening is frequently deployed on low-cost, portable platforms already dictated by clinical and budget constraints rather than freely chosen for the model, so this study treats accuracy–efficiency balancing as a trade-off to be characterized empirically on such a representative edge platform.

Deep learning, particularly Convolutional Neural Network (CNN) architectures, has demonstrated strong performance in automated ECG classification. Strodthoff et al. [2] showed that 1D-CNN variants achieve Macro-AUC values up to 0.937 on the PTB-XL benchmark using a record-level waveform approach, feeding complete 10-second 12-lead recordings directly into the network without segmentation. However, most prior work focuses on classification performance without considering deployment on resource-constrained edge devices [3].

Edge computing enables low-latency, on-device ECG analysis, making it well suited for continuous wearable monitoring [4]. The NVIDIA Jetson Orin Nano is a capable, low-cost edge AI platform supporting TensorRT acceleration; however, selecting an appropriate model requires balancing classification accuracy against computational efficiency, as different architectures exhibit different trade-offs on resource-constrained hardware [5]. Existing ECG benchmarking studies, however, mainly report results on server-class GPUs, leaving on-device latency, resource utilization, and the impact of FP16 (16-bit floating point) quantization insufficiently explored [3].

To address these limitations, this study compares four established 1D-CNN architectures for record-level 12-lead ECG classification on PTB-XL under identical preprocessing, hyperparameters, and data splits, deployed on the Jetson Orin Nano via TensorRT FP16 and evaluated on classification performance, latency, and resource utilization. A Weighted Sum Model (WSM), with criterion weights verified via the Analytic Hierarchy Process (AHP), integrates these metrics into a transparent, reproducible ranking for architecture selection.

2 Related Work

Several prior studies provide the methodological foundation for this work. Ahmed et al. [6] demonstrated that a 1D-CNN with Conv1D, batch normalization, max pooling, and dropout layers can effectively capture ECG morphology, establishing the baseline architecture adopted here. He et al. [7] introduced residual connections that allow deeper networks to train stably, later applied to 1D ECG signals by Strodthoff et al. [2], who showed that ResNet-based architectures achieve Macro-AUC up to 0.937 on PTB-XL. Multi-scale temporal feature extraction via parallel convolution branches of different kernel sizes was proposed by Prabhakararao and Dandapat [8], motivating the Multi-Scale CNN variant evaluated here, while the Squeeze-and-Excitation channel attention mechanism, applied to

ECG by Suraweera and Weerasinghe [9], forms the basis of the Attention-Based CNN variant.

Regarding edge deployment, Farag [10] showed that compact CNN models can achieve high accuracy on resource-constrained hardware, while Pramukantoro et al. [5] demonstrated that inference efficiency and accuracy trade off differently depending on architecture and device. Drissi et al. [3] established that single-metric evaluation is insufficient for model selection in resource-constrained settings, motivating the Weighted Sum Model (WSM) [11] used here, with weights cross-validated via the Analytic Hierarchy Process [12].

3 Research Method

This section describes the experimental pipeline, including dataset preparation, signal preprocessing, model architecture design, training configuration, edge inference evaluation, and multi-criteria model selection.

3.1 Dataset and Data Preparation

The PTB-XL dataset [13] contains 21,799 12-lead ECG recordings (10 s each at 100 Hz) from 18,869 patients. Single-label filtering on five diagnostic superclasses—Normal ECG (NORM), Myocardial Infarction (MI), ST/T-wave Change (STTC), Conduction Disturbance (CD), and Hypertrophy (HYP)—yields 16,244 records, whose distribution across splits is shown in Table 1. Official stratified folds were used: folds 1–8 for training, fold 9 for validation, and fold 10 for testing, ensuring strict patient-wise separation between each sets.

3.2 Preprocessing

Each recording was loaded via `wfdb` at 100 Hz and standardized to exactly 1,000 samples by zero-padding or truncation, yielding a $(1000, 12)$ array. A 4th-order Butterworth bandpass filter with a passband of 0.5–24 Hz was applied to suppress baseline wander and high-frequency noise while preserving the QRS morphology information [14]. Per-channel Z-score normalization was computed exclusively from the training set statistics and applied to validation and test data to avoid

Table 1: Class distribution across data subsets.

Class	Train	Val	Test
NORM	7,243	914	912
MI	2,043	233	256
STTC	1,903	255	242
CD	1,353	171	184
HYP	415	64	56
Total	12,957	1,637	1,650

information leakage, equalizing amplitude scale across leads and stabilizing CNN convergence [14].

3.3 Model Architectures

The four architectures span the dominant 1D-CNN design axes in the ECG literature while remaining fully expressible in TensorRT-compatible operators. All models take an input tensor of shape $(1000, 12)$ and output a 5-class softmax vector over the five superclasses. Every model shares identical regularization—L2 (1×10^{-3}), Dropout (0.50), SpatialDropout1D (0.15)—and the training schedule in Table 2; only the backbone differs. Dilated convolutions enlarge the temporal receptive field without increasing parameter count.

1D-CNN. Three Conv1D layers (32, 64, 128 filters; kernel sizes 7, 5, 3; dilation rates 1, 2, 4), each followed by Batch Normalization and SpatialDropout1D, with MaxPooling1D after the first two. Global Average Pooling then Dense(64) with Dropout and softmax output.

1D-ResNet. An initial Conv1D(64, k7) with Batch Normalization and MaxPooling1D precedes three residual blocks [7] (dilation rates 1, 2, 4; filters 64, 128, 128). Each block contains two Conv1D layers with an identity shortcut (1×1 convolution for dimension matching when needed). Concatenated GlobalAvgPool and GlobalMaxPool (256-d) feed Dense(128) with Dropout and softmax. The residual update is:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x} \tag{1}$$

where $\mathcal{F}(\mathbf{x}, \{W_i\})$ denotes the stacked Conv1D–BN residual mapping and $\mathbf{x}$ the identity shortcut.

Multi-Scale CNN. Three parallel Conv1D branches (32 filters each; kernel sizes 3, 5, 7; dilation rates 1, 2, 4) capture features at multiple temporal scales [8]. Feature maps are concatenated (96 channels), followed by Batch Normalization, SpatialDropout1D, MaxPooling1D, Conv1D(128), Global Average Pooling, and Dense(128) with Dropout and softmax.

Attention-Based CNN. Two Conv1D layers (64 filters k5; 128 filters k3 d2), each followed by Batch Normalization and a Squeeze-and-Excitation (SE) channel attention block [9]. The SE recalibration is:

$$\tilde{\mathbf{x}}_c = \sigma(W_2\, \delta(W_1\, \mathbf{z}_c)) \cdot \mathbf{x}_c \tag{2}$$

where $\mathbf{z}_c = \text{GlobalAvgPool}(\mathbf{x}_c)$, δ is ReLU, σ is sigmoid, and W_1, W_2 are FC weight matrices. MultiHeadAttention is deliberately avoided to ensure full TensorRT compatibility. Figure 1 summarizes all four backbones; Table 2 lists the shared training hyperparameters and Table 5 the trainable parameter count of each variant.

3.4 Training Configuration

All models were trained with identical hyperparameters (Table 2). Class imbalance was addressed via balanced class weights [5], and data augmentation

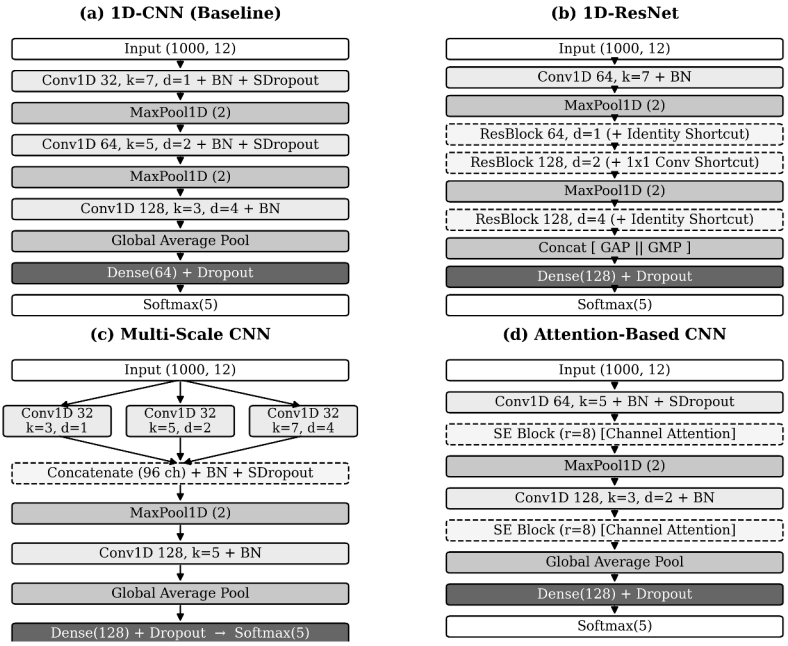


Fig. 1: Block diagrams of the 1D-CNN variants: (a) 1D-CNN, (b) 1D-ResNet, (c) Multi-Scale CNN, and (d) Attention-Based CNN. (BN: Batch Normalization; SDrop: SpatialDropout1D; GAP/GMP: Global Avg/Max Pool).

Table 2: Training hyperparameters (identical for all four models).

Parameter	Value
Optimizer	Adam, lr 3×10^{-4}
Batch Size / Max Epochs	64 / 100
Early Stopping	patience = 12 (`val_loss`)
ReduceLROnPlateau	patience = 4, factor = 0.3, min = 1×10^{-6}
Label Smoothing / L2	0.10 / 1×10^{-3}
Dropout / SpatialDropout1D	0.50 / 0.15

(Gaussian noise $\sigma = 0.02$, amplitude scaling $\times[0.90, 1.10]$) was applied on training data only, with categorical cross-entropy and label smoothing ($\epsilon = 0.10$) as the loss function.

3.5 Edge Inference

Hardware. Training used a desktop with NVIDIA GeForce RTX 4070 Super (12 GB VRAM), Intel Core i7-14700F, and 32 GB RAM. Batch inference was evaluated on the NVIDIA Jetson Orin Nano Developer Kit [15]: 6-core ARM

Cortex-A78AE CPU, 8 GB LPDDR5 unified memory, and an NVIDIA Ampere GPU with 1024 CUDA cores (40 TOPS).

TensorRT conversion. Trained Keras models were converted to TensorRT FP16 (`precision_mode=FP16`, `max_workspace=256 MB`, `min_segment_size=3`), using only TensorRT-compatible operators.

Measurement. Latency was measured via mini-batch processing (batch size = 32), reported as average per-sample latency (ms). CPU utilization was monitored via `psutil`; GPU and RAM via a background `TegrastatsMonitor` thread at 200 ms intervals, after ten warm-up runs. WSM/AHP used accuracy on the full 16,244-record dataset, while classification results are reported on fold 10.

3.6 WSM-Based Model Selection with AHP Weight Validation

WSM formulation. Each criterion value x_{ij} is min-max normalized to $[0, 1]$ to ensure scale independence across heterogeneous metrics [11]. Benefit criteria (accuracy) are mapped so higher values yield higher scores; cost criteria (latency, CPU, GPU, RAM) are inverted so that lower raw values yield higher normalized scores. The overall WSM score for model i is:

$$\mathrm{WSM}_i = \sum_{j=1}^{n} w_j \cdot x'_{ij}, \quad \sum_j w_j = 1 \tag{3}$$

The weights were assigned manually following an accuracy-priority deployment rationale rather than derived empirically: accuracy = 0.40, the largest share, since diagnostic reliability affects patient safety; latency = 0.30, since near real-time inference is needed clinically; CPU = 0.15 and GPU = 0.10, since the Jetson's CPU is the more frequent bottleneck for preprocessing/I-O while GPU inference is already TensorRT-optimized; and RAM = 0.05. To verify this manual assignment is not arbitrary, the same rationale was independently re-derived via the Analytic Hierarchy Process (AHP) [12], previously used for deep-learning model selection in healthcare [3]: the five criteria were compared pairwise on a 1–9 Saaty scale, the matrix column-normalized, and weights taken as the row-wise mean, with consistency checked via $CR = CI/RI$ ($RI = 1.12$ for $n = 5$; $CR \leq 0.10$ is acceptable [12]). This yielded $w_{acc} = 0.478$, $w_{lat} = 0.284$, $w_{CPU} = 0.118$, $w_{GPU} = 0.075$, $w_{RAM} = 0.046$ with $CR = 0.014$, closely mirroring the manual WSM weights and confirming consistent judgments.

4 Results and Discussion

This section presents evaluation results on classification performance, edge-device efficiency, and WSM/AHP-based model selection.

4.1 Classification Performance on Fold 10 Test Set

Table 3 summarizes fold-10 metrics (1,650 records). 1D-ResNet achieves the highest accuracy (79.94%) and Macro AUC (0.9354); 1D-CNN is competitive

Table 3: Overall classification performance on fold 10 test set.

Model	Accuracy	Macro Precision	Macro Recall	Macro F1-Score	Macro AUC
1D-CNN	0.7915	0.7016	0.6743	0.6852	0.9312
1D-ResNet	0.7994	0.7102	0.7062	0.7077	0.9354
Multi-Scale CNN	0.7618	0.6600	0.6994	0.6755	0.9292
Attention CNN	0.7618	0.6596	0.6878	0.6696	0.9299

Table 4: Per-class F1-score on fold 10 test set.

Class	1D-CNN	1D-ResNet	Multi-Scale	Attention
CD	0.7273	0.7600	0.7120	0.7227
HYP	0.4040	0.4561	0.4706	0.4394
MI	0.6887	0.7206	0.6452	0.6367
NORM	0.8658	0.8686	0.8438	0.8456
STTC	0.7403	0.7332	0.7059	0.7037

(79.15%, AUC 0.9312); Multi-Scale CNN and Attention CNN both reach 76.18% (AUC 0.9292 and 0.9299). All models exceed Macro AUC 0.92.

Table 4 presents per-class F1-scores. NORM scores highest (0.8438–0.8686) owing to its large sample count; HYP is most challenging (F1 0.40–0.47) due to class imbalance and morphological overlap with other classes. 1D-ResNet achieves the best F1 for NORM, MI, and CD, while Multi-Scale CNN yields the highest HYP recall (0.5714).

4.2 Computational Efficiency on Jetson Orin Nano

Table 5 reports batch inference efficiency, parameter counts, and full-cohort accuracy (the WSM/AHP accuracy input, not a generalization measure—see Table 3) under TensorRT FP16. All models achieve sub-millisecond latency. 1D-CNN has the lowest latency (0.5291 ms) and fewest parameters (135,589); 1D-ResNet has the highest latency (0.7994 ms), GPU utilization (69.62%), and parameter count (247,365), consistent with deeper models exploiting TensorRT GPU acceleration more, though not strictly monotonically. Attention CNN has the highest memory use (5,902.06 MB) and 1D-ResNet the lowest (5,261.44 MB), so memory is not strictly tied to depth.

4.3 Model Selection via WSM and AHP

Table 6 summarizes the WSM results with normalized contributions per criterion, alongside the AHP-derived score (cost-criterion contributions of 0.000 denote the least favorable raw value after min–max inversion, not zero resource usage; e.g. 1D-ResNet's 0.000 latency contribution reflects its highest raw latency in Table 5, not zero cost). 1D-ResNet scores highest under both schemes (WSM 0.600; AHP 0.641), driven by its full-cohort accuracy despite higher latency and

Table 5: Inference efficiency, parameter count, and full-cohort accuracy of CNN variants on Jetson Orin Nano (TensorRT FP16).

Model	Params	Latency (ms)	CPU (%)	GPU (%)	RAM (MB)	Accuracy
1D-CNN	135,589	0.5291	13.65	50.51	5,586.67	0.8204
1D-ResNet	247,365	0.7994	9.19	69.62	5,261.44	0.8796
Multi-Scale CNN	182,437	0.5667	11.32	57.43	5,286.54	0.7952
Attention CNN	168,869	0.6105	11.02	55.18	5,902.06	0.7925

Table 6: Multi-criteria decision results, comparing manually assigned (WSM) and AHP-derived weighting schemes.

Model	Accuracy	Latency	CPU	GPU	RAM	WSM	Rank	AHP	Rank
1D-ResNet	0.400	0.000	0.150	0.000	0.050	0.600	1	0.641	1
1D-CNN	0.128	0.300	0.000	0.100	0.025	0.553	2	0.534	2
Multi-Scale CNN	0.013	0.258	0.078	0.064	0.048	0.461	3	0.412	3
Attention CNN	0.000	0.210	0.088	0.076	0.000	0.374	4	0.325	4

GPU cost; 1D-CNN ranks second (WSM 0.553; AHP 0.534) via latency and CPU efficiency, with Multi-Scale CNN and Attention CNN lower under both. The AHP-derived ranking ($CR = 0.014$) is identical to the manually assigned WSM ranking, indicating that 1D-ResNet's selection as optimal for accuracy-priority deployment is robust to the weighting method rather than an artifact of arbitrarily chosen weights.

4.4 Discussion

The results reveal a clear accuracy–efficiency trade-off. 1D-ResNet and 1D-CNN define the Pareto front: 1D-ResNet leads in classification quality, 1D-CNN in speed and parameter efficiency. Multi-Scale CNN and Attention CNN are dominated on both axes despite having more parameters than the baseline (Table 5), suggesting that parallel multi-resolution branches and channel attention alone do not guarantee proportional gains; identity shortcuts let 1D-ResNet train deeper and extract richer temporal features more parameter-efficiently for this record-level task [8]. Deeper models also exploit TensorRT GPU acceleration more effectively (inverse CPU–GPU pattern, Table 5), and despite its higher latency, 1D-ResNet still operates comfortably within the Jetson's resource budget.

The HYP class consistently registers the lowest F1 (0.40–0.47) on fold 10 (Table 4). Table 7 reports the true-HYP row of each model's confusion matrix on the full 16,244-record cohort: across all four models, misclassified HYP recordings are predicted as NORM far more often than as any other class (e.g., 149 of 535 true-HYP cases for 1D-CNN), rather than being confused primarily with CD or MI. Clinically this is the more concerning failure mode, since hypertrophy cases risk being missed entirely as false negatives rather than flagged under a different abnormal category; an automated triage system built on these models

Table 7: True-HYP row of the confusion matrix on the full 16,244-record cohort (predicted class); diagonal (correct) entries in bold.

Model	Pred. CD	Pred. HYP	Pred. MI	Pred. NORM	Pred. STTC
1D-CNN	11	**321**	15	149	39
1D-ResNet	3	**470**	8	41	13
Multi-Scale CNN	8	**394**	14	81	38
Attention CNN	7	**360**	20	102	46

should therefore flag borderline HYP predictions for clinician review rather than act as a stand-alone diagnostic tool. Balanced class weighting only partially mitigates this imbalance; future work should explore focal loss or oversampling. Our results remain competitive with prior server-GPU PTB-XL studies [2] while additionally accounting for edge deployment constraints.

From a deployment perspective, the WSM/AHP framework [3, 12, 11] selects the optimal architecture under multi-objective constraints; here 1D-ResNet is preferred, whereas 1D-CNN may suit latency- or memory-critical scenarios given its 34% lower latency and roughly 45% fewer parameters. Two limitations remain: performance is reported on a single held-out fold rather than averaged across all official folds, so the 0.79-percentage-point gap between 1D-ResNet and 1D-CNN should be interpreted cautiously pending formal statistical validation (e.g., repeated seeds or cross-fold evaluation); and both weighting schemes still reflect one accuracy-priority scenario, so a broader weight-sensitivity sweep is left for future work.

5 Conclusion

This paper presented a benchmarking and deployment-characterization study of four established 1D-CNN variants—1D-CNN, 1D-ResNet, Multi-Scale CNN, and Attention-Based CNN—for record-level ECG classification on PTB-XL, evaluated on the NVIDIA Jetson Orin Nano. Each model receives a complete 10-second, 12-lead recording as a $(1000, 12)$ tensor after bandpass filtering and Z-score normalization, without beat segmentation. All four architectures achieve Macro AUC > 0.92, with 1D-ResNet attaining the best accuracy (79.94%) and Macro AUC (0.9354) at the cost of the largest parameter count (247,365), while 1D-CNN delivers the lowest latency (0.5291 ms) and smallest footprint (135,589 parameters). Multi-criteria evaluation via WSM, with weights cross-validated through AHP (consistency ratio = 0.014), identifies 1D-ResNet as optimal (WSM 0.600; AHP 0.641) for accuracy-priority deployment—a ranking robust across both schemes—while 1D-CNN suits latency- or memory-critical scenarios. Confusion-matrix analysis further shows that most HYP misclassifications are predicted as NORM rather than another abnormal class, an important clinical caveat for any triage use of these models. Future work includes multi-label classification, focal loss or SMOTE for HYP, cross-fold statistical validation, INT8 quantization, and evaluation on additional edge platforms.

References

1. World Health Organization: Cardiovascular diseases (CVDs): key facts (2025). https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)
2. Strodthoff, N., Wagner, P., Schaeffter, T., Samek, W.: Deep learning for ECG analysis: benchmarks and insights from PTB-XL. IEEE J. Biomed. Health Inform. 25(5), 1519–1528 (2021). https://doi.org/10.1109/JBHI.2020.3022989
3. Drissi, N., El-Kassabi, H., Serhani, M.A.: A multi-criteria decision analysis framework for evaluating deep learning models in healthcare research. Decis. Anal. J. 13, 100523 (2024). https://doi.org/10.1016/j.dajour.2024.100523
4. Pramukantoro, E. S., Gofuku, A.: A Heartbeat Classifier for Continuous Prediction Using a Wearable Device. Sensors, 22(14), 5080 (2022).. https://doi.org/10.3390/s22145080
5. Pramukantoro, E. S., Amron, K., Wardhani, V., Kamila, P.A.: A study of inferencing ECG heartbeat classification based on CNN and LSTM on edge devices. Int. J. Comput. Digital Syst. 18(1), 1–12 (2025). https://doi.org/10.12785/ijcds/1571111934
6. Ahmed, A.A., Ali, W., Abdullah, T.A.A., Malebary, S.J.: Classifying cardiac arrhythmia from ECG signal using 1D CNN deep learning model. Mathematics 11(3), 562 (2023). https://doi.org/10.3390/math11030562
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
8. Prabhakararao, E., Dandapat, S.: Multi-scale convolutional neural network ensemble for multi-class arrhythmia classification. IEEE J. Biomed. Health Inform. 26(8), 3802–3812 (2022). https://doi.org/10.1109/JBHI.2021.3138986
9. Suraweera, M., Weerasinghe, M.: Attention-based CNN for efficient embedded ECG signal classification. In: Proc. Smart Sustainable Development Conf. (SSD 2025), pp. 160–170. Atlantis Press (2025). https://doi.org/10.2991/978-94-6463-720-5_14
10. Farag, M.M.: A self-contained STFT CNN for ECG classification and arrhythmia detection at the edge. IEEE Access 10, 94469–94486 (2022). https://doi.org/10.1109/ACCESS.2022.3204703
11. Fishburn, P.C.: Letter to the Editor—Additive utilities with incomplete product sets: application to priorities and assignments. Operations Research 15(3), 537–542 (1967). https://doi.org/10.1287/opre.15.3.537
12. Saaty, T.L.: The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. McGraw-Hill, New York (1980)
13. Wagner, P., Strodthoff, N., Bousseljot, R.D., Kreiseler, D., Lunze, F., Samek, W., Schaeffter, T.: PTB-XL, a large publicly available electrocardiography dataset. Scientific Data 7 (2020). https://doi.org/10.1038/s41597-020-0495-6
14. Rainio, O., Teuho, J., Klén, R.: Evaluation metrics and statistical tests for machine learning. Scientific Reports 14, 6086 (2024). https://doi.org/10.1038/s41598-024-56706-x
15. NVIDIA Corporation: Jetson Orin Nano Developer Kit User Guide. NVIDIA Developer Documentation. (2026). https://docs.nvidia.com/jetson/orin-nano-devkit/user-guide/latest/hardware_layout.html

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

