RT-RCG: Neural Network and Accelerator Search Towards Effective and Real-time ECG Reconstruction from Intracardiac Electrograms

YONGAN ZHANG, ANTON BANTA, and YONGGAN FU, Rice University, USA
MATHEWS M. JOHN, ALLISON POST, and MEHDI RAZAVI, Texas Heart Institute, USA
JOSEPH CAVALLARO, BEHNAAM AAZHANG, and YINGYAN LIN, Rice University, USA

There exists a gap in terms of the signals provided by pacemakers (i.e., intracardiac electrogram (EGM)) and the signals doctors use (i.e., 12-lead electrocardiogram (ECG)) to diagnose abnormal rhythms. Therefore, the former, even if remotely transmitted, are not sufficient for doctors to provide a precise diagnosis, let alone make a timely intervention. To close this gap and make a heuristic step towards real-time critical intervention in instant response to irregular and infrequent ventricular rhythms, we propose a new framework dubbed RT-RCG to automatically search for (1) efficient Deep Neural Network (DNN) structures and then (2) corresponding accelerators, to enable **R**eal-**T**ime and high-quality **R**econstruction of E**C**G signals from E**G**M signals. Specifically, RT-RCG proposes a new DNN search space tailored for ECG reconstruction from EGM signals and incorporates a differentiable acceleration search (DAS) engine to efficiently navigate over the large and discrete accelerator design space to generate optimized accelerators. Extensive experiments and ablation studies under various settings consistently validate the effectiveness of our RT-RCG. To the best of our knowledge, RT-RCG is the first to leverage neural architecture search (NAS) to simultaneously tackle both reconstruction efficacy and efficiency.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; **Neural networks**; • **Applied computing** → **Bioinformatics**; **Health informatics;**

Additional Key Words and Phrases: Feature selection, discrete space search, electronic design automation

ACM Reference format:
Yongan Zhang, Anton Banta, Yonggan Fu, Mathews M. John, Allison Post, Mehdi Razavi, Joseph Cavallaro, Behnaam Aazhang, and Yingyan Lin. 2022. RT-RCG: Neural Network and Accelerator Search Towards Effective and Real-time ECG Reconstruction from Intracardiac Electrograms. *J. Emerg. Technol. Comput. Syst.* 18, 2, Article 29 (March 2022), 25 pages.
https://doi.org/10.1145/3465372

This work was supported in part by the National Institutes of Health under Grant R01HL144683 and National Science Foundation under Grant CCF-1838873.
Authors' addresses: Y. Zhang, A. Banta, Y. Fu, J. Cavallaro, B. Aazhang, and Y. Lin, Rice University, 6100 Main St., Houston, TX; emails: {yz87, arb17, yf22, cavallar, aaz, yingyan.lin}@rice.edu; M. M. John, A. Post, and M. Razavi, Texas Heart Institute, 6770 Bertner Ave., Houston, TX; emails: {mjohn, apost}@texasheart.org, mehdirazavi1@gmail.com.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
1550-4832/2022/03-ART29 $15.00
https://doi.org/10.1145/3465372

1 INTRODUCTION

Over 5.8 million people in the USA and over 23 million worldwide are affected by cardiac diseases [8, 33], where the inability to generate or conduct the electrical signals necessary to stimulate muscle contraction is the major cause for many heart failures [16]. To treat these failures, artificial electronic pacemakers are usually implanted to stimulate the heart with electrical impulses to maintain or restore a normal rhythm. In particular, about 3 million people worldwide use pacemakers and 6,000,000 pacemakers are implanted each year [81]. Currently, cardiac pacemakers cannot sense or compute 12-lead ECGs from EGMs. Patients with pacemakers require regular and often costly and time-consuming hospital visits to ensure (1) the proper functioning of the pacemaker and (2) the timely adjustment of the pacing parameters to adapt to changes in the heart's condition over time. Thanks to recent advances in the **internet of things (IoT)** technologies, remote monitoring of pacemakers will become more commonplace, allowing doctors to check pacemaker status and thus reduce the frequency of costly hospital visits [94].

Despite the promising advantages of remotely monitoring pacemakers, there is still a gap in terms of the signals that can be provided by pacemakers and the ones doctors need to diagnose abnormal rhythms and provide appropriate therapy. Specifically, cardiac pacemakers utilize continuously collected EGMs, which are electrical activities sensed locally via implanted electrodes. However, 12-lead ECGs obtained from skin electrodes contain significantly greater information than EGMs, which, in certain cases, could be utilized to better diagnose abnormal rhythms and provide appropriate therapy. To close this gap, the synthesis or reconstruction of ECG signals from a set of EGM signals is of great significance in enabling effective remote monitoring of pacemakers, providing necessary therapy, and making timely clinical intervention possible [75]. As such, there has been a growing interest in developing techniques to reconstruct ECG signals from their corresponding EGM signals using linear filtering [26, 43, 44, 54], fixed dipole modeling algorithms [54, 55], and nonlinear reconstruction via a time delay neural network [41, 42, 63].

While the aforementioned techniques were pioneering steps, there is still much room to improve their performance for practical and widespread adoption. In particular, most of the existing techniques adopt either linear approaches that lack generalization capability for unseen symptoms and can fail in the presence of noises and artifacts, or a multivariate nonlinear approach that requires the simultaneous recording of both EGM and 12-lead ECG signals for every single patient [41, 42, 63]. Motivated by the recent breakthroughs in **deep neural networks (DNNs)** and their demonstrated promise in medical applications [20, 23, 24, 74], it is natural to consider DNN-based reconstruction techniques, aiming for much improved generalization capability and better efficacy towards more practical clinical uses. However, the excellent performance of DNN-based solutions often comes at the cost of high complexity (e.g., millions of parameters and operations [83]), which stands at odds with the extremely constrained resources at the implanted, battery-powered pacemakers. Specifically, restricted by the pacemaker's limited hardware budget, the often complex DNN-based solutions make it particularly challenging to handle real-time reconstruction on the pacemakers, which could enable improved and possibly life-critical interventions to the patients. Currently, EGMs stored in pacemakers are analyzed offline through an inpatient setting for improved diagnosis of the underlying condition, where therapeutic intervention might need to be changed over time and thus require real-time adaptation. For example, monitoring ECG data in real-time can allow for determination of potentially deadly ventricular arrhythmias [57], and dictate pacing mediated therapies such as anti-tachycardia pacing. Online real-time reconstruction of EGMs to ECGs allows for real-time and immediate intervention and thus potentially paves the way for novel treatments, whereas offline reconstruction may not always be possible, and the potential latency involved in doing so could be life threatening. Another example is utilizing ECGs

in real-time for optimizing parameters for cardiac resynchronization therapy to treat heart failure patients [4], where a real-time embedded accelerator allows for on-device reconstruction with a low latency and is thus critical. Furthermore, with traditional pacemakers slowly being replaced by leadless pacemakers [73], such an accelerator would also pave the way for improved therapy with minimal sensing sites.

To this end, we aim to develop an efficient DNN-based reconstruction framework to push forward the efficacy and efficiency frontier towards practical and widespread adoption by leveraging recent advances in neural architecture search and DNN acceleration. Specifically, we make the following contributions in this work:

- We propose a new framework dubbed RT-RCG, which can automatically search for (1) efficient DNN structures and then (2) corresponding accelerators to enable **R**eal-**T**ime and high-quality **R**econstruction of E**C**G signals from E**G**M signals. To the best of our knowledge, the proposed RT-RCG is the first to simultaneously tackle and leverage **neural architecture search (NAS)** for both reconstruction efficacy and efficiency.
- Drawing inspiration from existing ECG reconstruction works, RT-RCG proposes a new DNN search space tailored for ECG reconstruction from EGM signals to enable automated search for DNNs that consistently outperform **state-of-the-art (SOTA)** reconstruction techniques in terms of both reconstruction correlation (between the reconstructed ECGs and the real-measured ECGs) and algorithmic generalization capability.
- Built upon recent advances in DNN acceleration, RT-RCG incorporates a **differentiable acceleration search (DAS)** engine that makes use of gradient-based optimization to efficiently navigate over the large and discrete accelerator design space to automatically generate optimized accelerators that achieve real-time reconstruction.
- Extensive experiments and ablation studies under various settings consistently validate the effectiveness of our proposed RT-RCG in leading to higher reconstruction quality and better reconstruction efficiency as compared to SOTA reconstruction algorithms and DNN accelerators, respectively. We believe that RT-RCG has made a nontrivial step towards practical ECG reconstruction from EGM signals on the pacemaker, promising the real possibility of real-time critical intervention in instant response to irregular and infrequent ventricular rhythms that require timely treatment.

2 RELATED WORKS

ECG Reconstruction. In response to the practical need of ECG reconstruction from EGM signals, various methods have been proposed [26, 41–44, 54, 55, 63] using linear filtering [26, 43, 44, 55], fixed dipole modeling algorithms [54], nonlinear filtering [41, 42], and time delay neural networks [63], In particular, a single EGM channel was used to synthesize a single ECG lead in Reference [26], which can be highly dependent on the chosen EGM lead; later, logical extension of Reference [26] was developed that uses all EGM leads for synthesis [43, 44], where both the EGMs and the ECGs were first projected onto a 3D space and then three linear filters were calculated between the signals, providing an indirect way to find the transfer functions between EGM signals and the 12-lead ECG; similarly, References [54, 55] directly calculated a multivariate linear transfer matrix between the EGMs and the 12-lead ECGs via penalized linear regression. Despite their satisfactory performance, especially for patients with a surface ECG containing only a one-beat morphology, these linear methods can suffer from a degraded correlation between the EGMs and the ECGs in real applications due to the noises and artifacts present, and the natural evolution and diversity of the pathology. The limitation of linear reconstruction methods (e.g., an average correlation value of lower than 0.5 in Reference [55]) motivated the multivariate nonlinear

approach presented in References [41, 42, 63], all of which require the simultaneous recording of the EGMs and 12-lead ECGs for every single patient to train a **time-delay artificial neural network (TDNN)**. While this method provided the best average correlation results for sinus rhythm heartbeats, it is still limited for practical uses, as it cannot effectively reconstruct diseased morphologies and 12 different TDNN models must be calculated to reconstruct each ECG lead.

Built upon the above prior works, RT-RCG targets reconstruction algorithms that are generally applicable in the presence of noise, artifacts, and diverse pathologies.

DNNs in Cardiology Applications. The recent breakthroughs of DNNs in various fields have sparked a growing interest in developing DNN-based solutions for cardiologic problems spanning from ECG classification to sleep status monitoring [22, 29, 30, 49, 88, 90]. In particular, Reference [88] adopted a DNN to remove noises contaminating the ECG signals; Reference [22] used two DNNs together with short-duration (5 seconds) ECG segments to detect pulses during out-of-hospital cardiac arrest; Reference [90] proposed to utilize DNNs for the classification of ECG signals into different heart rhythms (i.e., normal beat or different types of arrhythmias); Reference [49] made use of a DNN and a hidden Markov model to detect obstructive sleep apnea based on single lead ECG signals. The readers are referred to Reference [7] for a detailed survey on applying DNNs to cardiology applications. While these works demonstrate the great potential of DNN-based solutions for cardiologic problems, DNN-powered ECG-EGM reconstruction algorithms are still under-explored, let alone real-time reconstruction implementation, motivating us to propose and develop our RT-RCG framework.

Neural Architecture Search. Neural architecture search (NAS) [99] has emerged as one of the most significant sub-fields of AutoML [37] as it enables automatically searching for an optimal DNN structure from the given data and has outperformed manually designed DNNs on a range of tasks such as image classification [34, 52, 71, 72] and segmentation [13, 14, 51]. Early NAS works achieve SOTA performance at the cost of enormous search time [64, 99, 100]. Specifically, **reinforcement learning (RL)**-based NAS [34, 71, 72, 99, 100] and evolutionary algorithm-based NAS [62, 64] explored the search space and train each sampled network candidate from scratch, thus suffering from prohibitive search costs. Later, **differentiable NAS (DNAS)** [9, 52, 77, 82, 84] was proposed to update the weights and architecture in a differentiable manner through supernet weight sharing, reducing the search time to several hours [69]. Motivated by the promising performance achieved by those DNAS works, recent works have extended DNAS to more tasks such as segmentation [14, 51], image enhancement [25, 47], and language modeling [12]. As a result, we leverage the DNAS method integrated with a new search space to develop our proposed RT-RCG framework.

DNN Accelerators. DNNs' powerful performance comes at a cost of a prohibitive complexity, motivating extensive research in dedicated DNN accelerators, as specialized hardware has the potential to achieve orders-of-magnitude higher energy/time efficiency. Specifically, it has been shown that aggressive efficiency can be achieved by carefully designing the micro-architectures (e.g., the number of memory hierarchies or **processing element (PE)** units, the storage size of different memories, and the shape of the PE array) and algorithm-to-hardware mapping strategies (i.e., dataflow). For example, representative works, such as ShiDiannao [21] and Eyeriss [15], identified the performance bottleneck caused by the required massive data movements and proposed novel micro-architectures and dataflows that aim to maximize data reuse for reducing the energy/time cost to access higher cost memories. Early works mostly rely on experts' manual design, which can be very time-consuming (months or even years) and require cross-disciplinary knowledge in algorithm, micro-architecture, and circuit design. In response to the intense demands and challenges of manually designing DNN accelerators, we have seen rapid development of design flow [10, 11, 66, 86] and DNN design automation frameworks [27, 76, 78, 79, 96] to standardize

the design flow of DNN accelerators and to expedite the development process. For example, the DNNBuilder accelerator [97] applied an automated resource allocation strategy, fine-grained layer-based pipeline, and column-based cache to deliver high-quality FPGA-based DNN accelerators, and Reference [89] made the first step towards automatically generating both FPGA- and ASIC-based DNN accelerators without humans in the loop given the DNNs from machine learning frameworks (e.g., PyTorch) for a designated application and dataset.

Leveraging the learning from prior works, RT-RCG integrates an DAS engine to automatically generate micro-architectures and dataflows to achieve real-time reconstruction.

DNN Algorithm and Accelerator Co-exploration. Exploring the networks and the corresponding accelerators in a joint manner [1, 31, 39, 40, 50, 92] has shown great potential towards maximizing both accuracy and efficiency. Recent works have extended NAS to jointly search DNN accelerators in addition to DNN structures. In particular, References [1, 31, 40, 92] conducted RL-based searches to co-explore the network structures and design parameters of an FPGA-/ASIC-based accelerator, but their RL-based methods can suffer from large search costs, limiting their scalability to handle large joint spaces. Recently, References [19, 50] extended differentiable NAS to network and accelerator co-search. However, Reference [50] only considered one accelerator parameter (i.e., the parallel factor of an FPGA accelerator template), which is not always applicable to most naturally non-differentiable accelerator design parameters such as loop order and loop size, while Reference [19] adopted a DNN to generate accelerator designs with network structures as the DNN's inputs, which lack interpretability. In contrast, our work adopts differentiable joint search in a sequential manner to efficiently explore a generic network and accelerator design space.

3 PRELIMINARIES OF DEEP NEURAL NETWORKS (DNNS) AND THE EGM/ECG DATA FORMAT

Deep Neural Networks (DNNs). Modern DNNs usually consist of a cascade of multiple **convolutional (CONV)**, pooling, and **fully connected (FC)** layers through which the inputs are progressively processed. The CONV and FC layers can be described as:

$$\boldsymbol{O}[c_o][e][f] = \sigma\left(\left(\sum_{c_i=0}^{C-1}\sum_{k_r=0}^{R-1}\sum_{k_s=0}^{S-1}\boldsymbol{W}[c_o][c_i][k_r][k_s] \times \boldsymbol{I}[c_i][eU+k_r][fU+k_s]\right) + \mathbf{B}[c_o]\right) \quad (1)$$
$$0 \le c_o < M,\ 0 \le e < E, 0 \le f < F,$$

where $\boldsymbol{W}$, $\boldsymbol{I}$, $\boldsymbol{O}$, and $\boldsymbol{B}$ denote the weights, input activations, output activations, and biases, respectively. In the CONV layers (see an example in Figure 1), C and M, E and F, R and S, and U stand for the number of input and output channels, the size of input and output feature maps, and the size of weight filters, and stride, respectively; while in the FC layers, C and M represent the number of input and output neurons, respectively; with σ denoting the activation function, e.g., a *ReLU* function ($ReLU(x) = max(x, 0)$). The pooling layers reduce the dimension of feature maps via average or max pooling. The recently emerging compact DNNs (e.g., MobileNet [35] and EfficientNet [72]) introduce depth-wise CONV layers and squeeze-and-excite layers that can be expressed in the above description as well [18].

Pre-processing of the EGM/ECG Signals. Here, we describe the adopted pre-processing for the EGM and ECG signals, both of which were recorded simultaneously during the cardiac ablation procedure. In the first step, the signals were initially obtained at a sampling frequency of 1,000 Hz, and subsequently bandpass filtered using a 5th order Butterworth filter with a cutoff frequency at 3 Hz and 50 Hz. The cutoff at 3 Hz helps to eliminate potential baseline wanders and the cutoff at 50 Hz can eliminate powerline interferences, electromyographic noise, and electrode motion artifact noise [45]. To align with the phase change caused by the pre-processing filtering in the

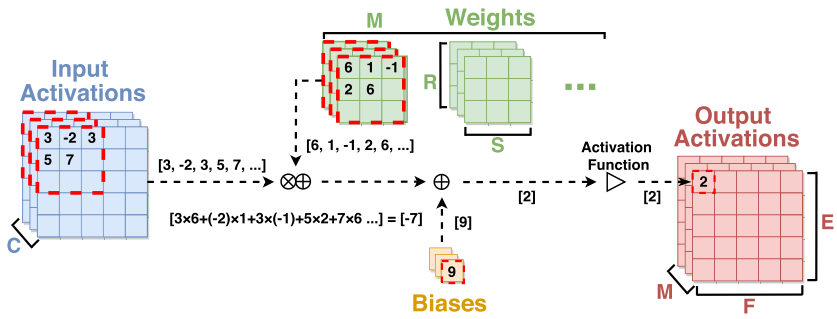

Fig. 1. An illustrative example of one CONV operation as formulated in Equation (1), where M/C (the number of input/output channels), E/F (the input feature map height/width), R/S (kernel height/width) and U (stride) are 3/3, 5/5, 3/3, and 1, respectively. This example assumes that ReLU is used as the activation function and the first output is 2.

forward direction, we adopt the zero phase filtering and also filter the signal backwards in time [46] to ensure that the pre-processing of the data does not introduce additional distortion. In the second step of the pre-processing, the data from the previous step is segmented to extract the QRS portion of ECG signals, which contains much information about the synchronization of the heart's ventricles and has been demonstrated to be a strong biomarker for overall cardiac health [56].

Time-frequency Representation of EGM/ECG Signals. To make use of DNNs to reconstruct ECG signals from EGM signals, we first transfer EGM signals' 2-**dimensional (2D)** multi-channel time-series representation into a 3-**dimensional (3D)** time-frequency representation with the help of STFT, inspired by the similar treatments in speech recognition and audio processing applications [3, 61, 91].

Assuming that the matrix $S_t \in \mathbb{R}^{M\times T}$ denotes the EGM time-series where M and T correspond to the number of channels and the number of time samples for each channel, respectively, then S_t can be re-formulated as $S_t = [\mathbf{s}_t^{(1)}, \ldots, \mathbf{s}_t^{(m)}, \ldots, \mathbf{s}_t^{(M)}]^\top$, with $\mathbf{s}_t^{(m)} \in \mathbb{R}^{T\times 1}$ and $\top$ denoting the time-series for each of the M channels and the transpose operator, respectively. As such, the corresponding 3D time-frequency signals, denoted as S_{tf}, can be represented as:

$$\begin{aligned} S_{tf} &= \left[\mathbf{s}_{tf}^{(1)}, \ldots, \mathbf{s}_{tf}^{(M)}\right]^\top \\ \mathbf{s}_{tf}^{(m)} &= \left[\mathbf{s}_t^{(m)} \circ \mathbf{h}_1, \ldots, \mathbf{s}_t^{(m)} \circ \mathbf{h}_K\right]^\top, \end{aligned} \tag{2}$$

where $\forall k \in \{1, \ldots, K\}$, $\mathbf{h}_k \in \mathbb{C}^T$ defines the kth time-frequency filter in the complex space corresponding to $\mathbf{s}_t^{(m)}$, and $\circ$ denotes the convolution operator. We set the length of each time-frequency filter (after filtering) as T_f, and thus $\mathbf{s}_{tf}^{(m)} \in \mathbb{C}^{K\times T_f}$ represents a 2D complex matrix with each row denoting the time domain information and each column denoting the frequency domain information. Concatenating all channels' time-frequency representation, we then have a 3D complex matrix $S_{tf} \in \mathbb{C}^{M\times K\times T_f}$. In this work, we use windowed Fourier filters as the filters $\mathbf{h}_k$, i.e., transferring the time-series representation into its time-frequency one that becomes the operation of applying a 3D **short-time Fourier transform (STFT)** operator to the time-series EGM signals.

4 THE PROPOSED RT-RCG FRAMEWORK

4.1 RT-RCG: Overview and Problem Formulation

Framework Overview. Figure 2 shows an overview of the proposed RT-RCG framework. Given the recorded EGM signals, user-specified demands (e.g., accuracy and latency), and hardware

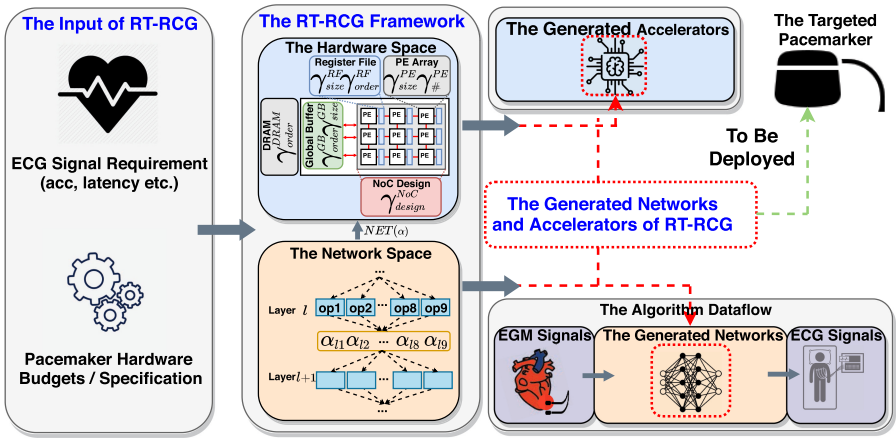

Fig. 2. An overview of the proposed RT-RCG framework, which accepts the recorded EGM/ECG signals dataset and the target hardware specification as inputs to automatically generate reconstruction networks and their corresponding accelerators to maximize the reconstruction quality and acceleration efficiency.

resource budgets/specification, our RT-RCG framework automatically searches for networks to maximize the reconstruction efficacy and then the corresponding accelerators to maximize the hardware acceleration efficiency, i.e., the outputs of RT-RCG include (1) the searched network to be used for reconstructing ECG signals from the input EGM signals and (2) the searched accelerator to process the searched network with optimized hardware efficiency. In particular, our RT-RCG framework consists of two components, i.e., a **differentiable network search (DNS)** engine and a DAS engine, which will be described in Section 4.2 and Section 4.3, respectively.

The Optimization Formulation. As stated in Section 1, RT-RCG is designed to reconstruct the full 12-lead ECG signals from the recorded (partial) EGM signals, with both originally being time-series signals. For notation, we denote the EGM and ECG samples using $\{X_{n_t}\}_{n=1}^{N}$ and $\{Y_{n_t}\}_{n=1}^{N}$, respectively, where N denotes the total number of heartbeats in the dataset (see Table 3). Meanwhile, the EGM and ECG signals can be represented using a 2D matrix, i.e., $X_{n_t} \in \mathbb{R}^{M_{EGM} \times T}$ and $Y_{n_t} \in \mathbb{R}^{M_{ECG} \times T}$, where M_{EGM} and M_{ECG} denote the number of channels (leads) for the ECG and EGM signals, respectively, and T denotes the number of time samples per heartbeat. As introduced in Section 3, the ECG and EGM signals will first be transferred into a time-frequency format denoted as $X_{n_{tf}} \in \mathbb{R}^{M_{EGM} \times K \times T_f}$ and $Y_{n_{tf}} \in \mathbb{R}^{M_{ECG} \times K \times T_f}$, respectively. In this work, we have $M_{EGM} = 5$ and $M_{ECG} = 12$, respectively, and both K and T_f are empirically fixed to 16 with a STFT window size of 30 and overlap of 6 during the filtering, based on the collected dataset (see Table 3). Through empirical studies, this STFT configuration gave us the best subsequent reconstruction accuracy with the least number of parameters. As such, the problem of reconstructing ECG from EGM becomes how to map the signals in $\mathbb{R}^{M_{EGM} \times K \times T_f}$ to that in $\mathbb{R}^{M_{ECG} \times K \times T_f}$, which can be considered as a problem of multivariate regression and the corresponding optimization can be formulated as follows:

$$\min_{f \in \mathcal{H}} \sum_{n=1}^{N} \mathcal{L}(f(X_{n_{tf}}), Y_{n_{tf}}), \tag{3}$$

where $\mathcal{H}$ denotes the function space, f denotes the reconstruction function that aims to reconstruct $Y_{n_{tf}}$ given $X_{n_{tf}}$, $\mathcal{L}$ denotes the loss function of reconstruction capturing the total difference (e.g., the mean square error) between the reconstructed samples $f(X_{n_{tf}})$ and the real-measured samples $Y_{n_{tf}}$ for all the N samples. The goal of the optimization is to find a reconstruction

Table 1. Visualizing RT-RCG's Network Backbone with 14 Searchable Blocks, where TBS Denotes "To Be Searched"

Operation type	CONV	Maxpool	CONV	Maxpool	**Searchable blocks × 14**	DECONV	Upsample	DECONV	Upsample	CONV	CONV	CONV
Output channels	48	-	96	-	TBS	48	-	96	-	24	24	24
Kernel size	7	2	5	2	TBS	5	2	7	2	3	3	3
Stride	1	2	1	1	TBS	1	2	1	2	1	1	1

function f that minimizes the reconstruction loss $\mathcal{L}$. In RT-RCG, we use a DNN to approximate and search for f using RT-RCG's DNS engine, with the direct output of f having a time-frequency format and then being transferred back into a time-series format for evaluating the reconstruction efficacy. During training, the negative Pearson correlation [6] of the flattened time-frequency data between the reconstructed ECG and the corresponding real-measured ECG signals will be used as the loss for optimization. For evaluation, the Pearson correlation will be calculated between the reconstructed and corresponding real-measured ECG signals on a test set (excluded in training) after both of them are converted back to the time domain through the inverse STFT. Note that the (inverse) STFT process will neither be accelerated by the proposed RT-RCG's hardware nor be counted towards the final latency in our experiments. This is because for a single piece of input, the combined operations of both STFT and inverse STFT only take up about 1% of the total operations in the inference when DNNs shown in Table 8 are considered, assuming a fast convolution algorithm is adopted. The (inverse) STFT operation can thus be easily conducted on the hardware accelerator's accompanying CPU incurring a negligible latency overhead.

4.2 RT-RCG: The DNS Engine

The Network Search Space. Motivated by the success of the encoder-decoder structure [65] that has demonstrated its efficacy in learning compressed, interpretable, or structured representation of data for denoising, compression, and data completion [20, 23, 24, 74], RT-RCG's DNS engine adopts a search space based on an encoder-decoder-based network backbone with searchable blocks to extract and process diverse and patient-specific features from the complex EGM signals. As shown in Table 1 and visualized in Figure 6, our network starts from a fixed downsample branch and ends in a fixed up-sample branch with the intermediate blocks being set to be searchable for better extracting and processing of the features hidden in the EGM signals. The hypothesis is that such an encoder-decoder structure, i.e., a cascade of convolutional transformations and nonlinearities with a bottleneck dimension, allows the approximation of the underlying data to be manifold as discussed in Reference [20].

For the searchable blocks, inspired by the SOTA hardware-friendly search space in Reference [82] that searches the kernel size, channel expansion ratio, and group number for each building block, we propose a sequential search space with 14 searchable blocks and 9 candidate operations for each block, including standard convolutions with a kernel size of 3/5, inverted residual blocks with a kernel size of 3/5, a channel expansion of 1/3/5, and skip connections, which leads to a search space with a total of 9^{14} network choices.

The Network Search Algorithm. We adopt the **differentiable NAS (DNAS)** algorithm [52] considering its excellent search efficiency. In particular, we formulate the network search as a one-level optimization [36, 84] by making use of the unbiased gradient estimation [32] to adapt to the complex EGM signals that are diverse for different patients:

$$\min_{\omega,\alpha} \; L_{rec}(\omega,\alpha) + \lambda L_{cost}^{MAC}(\alpha), \tag{4}$$

where ω and α denote the supernet weights and the network architecture parameters, respectively, the latter of which contains the probability of selecting each candidate operation; L_{rec} and L_{cost}^{MAC}

denote the ECG-EGM reconstruction loss and hardware-cost loss, which is determined by the number of **multiply-accumulate operations (MACs)** in the given DNNs, respectively; and λ is a tradeoff parameter to balance the resulting reconstruction networks' accuracy and efficiency. In particular, the output of the lth layer A_l in our DNS engine is a weighted sum of all candidate operations:

$$A_l = \sum_{k=1}^{K} GS(\alpha_{lk}) O_{lk}(A_{l-1}), \tag{5}$$

where K is the number of candidate operations, O_{lk} is the kth operation for the lth layer, α_{lk} is the probability of O_{lk}, and GS denotes the Gumbel-Softmax function [38] that samples the operations based on the distribution parameterized by α. In our DNS, we adopt a soft version of Gumbel-Softmax, i.e., we use the output of Gumbel-Softmax as the weighted coefficient of O_{lk} with a continuous relaxation during backward pass [82] for updating α. At the end of the search, we derive the final/searched network by selecting the operation with the highest probability for each searchable block.

4.3 RT-RCG: The DAS Engine

In this subsection, we introduce the three key components in our proposed DAS engine, i.e., the accelerator template, the search space extracted from the accelerator template, and the search algorithm used to explore the search space.

4.3.1 The Accelerator Template of Our DAS Engine. Our DAS engine leverages a parameterized accelerator template that features a total of $\sim 10^5$ choices for the micro-architecture and dataflow, the latter of which determines how the network is temporally and spatially scheduled to be executed on the micro-architecture, e.g., row stationary, output stationary, weight stationary, and so on.

The Micro-architecture Overview. Our DAS engine leverages an accelerator template inspired by a SOTA DNN accelerator [67], which adopts a multi-chunk micro-architecture for maintaining high resource utilization when accelerating DNN layers with different structures (e.g., different sizes of the input/output feature maps and kernel sizes) to balance the communication bandwidth and improve the acceleration throughput. Our accelerator template parameterizes the multi-chunk micro-architecture. As illustrated in Figure 3 later, each chunk of the micro-architecture corresponds to a sub-accelerator, which has hierarchical memories (e.g., on-chip buffer and local **register files (RF)** and **processing elements (PEs)** characterized by searchable design knobs such as the types of PE interconnections (i.e., **Network-on-chip (NoC)**), allocated buffer sizes, and the computing scheduling and tiling (i.e., dataflows) to facilitate data reuse and parallelism. Specifically, each sub-accelerator sequentially processes multiple but not necessarily consecutive layers with similar network structures, while different sub-accelerators can be pipelined.

The Sub-accelerator Design. As shown in Figure 3, each sub-accelerator consists of (1) a secondary buffer to facilitate more local data reuse and reduce the higher-cost DRAM accesses and (2) a PE array, where each PE includes a **multiply and accumulate (MAC)** unit and local **register files (RFs)** for the inputs, weights, and outputs, respectively. For each sub-accelerator, the dataflow determines the networks' temporal and spatial mapping into the PE array and thus the data movement patterns within different memories/buffers/RF, leading to orders of magnitude difference in the acceleration performance [76]. As our accelerator template can parameterize both the micro-architecture and the dataflow (see Section 4.3.2), it enables our DAS engine to search for dedicated micro-architecture and dataflow to match the networks' structure to maximize the target hardware performance.

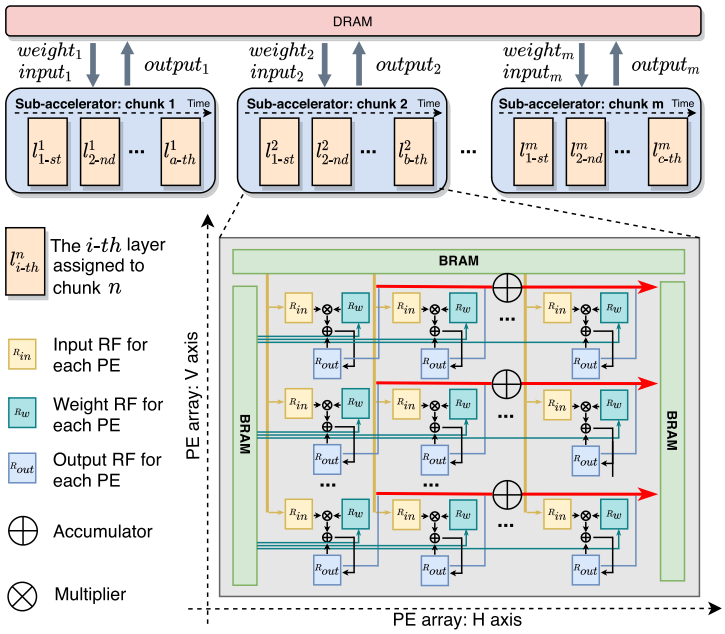

Fig. 3. An illustration of the parameterized micro-architecture adopted in the DAS engine of our RT-RCG framework.

Acceleration/Execution. Here, we describe the execution of the network within each sub-accelerator for better understanding. In Figure 3, if the data within the PEs process different input channels along the H (horizontal) axis of the PE array and different output channels along the V (vertical) axis of the PE array, then the weights with different input and output channels will be spatially mapped into all PEs and then stay stationary until all corresponding computations are finished. Meanwhile, the input corresponding to the weights that have been loaded into the PE array will be streamed in via multicast along the H axis and broadcast along the V axis, facilitating various weight reuse. The computed results along the H axis are accumulated while those along the V axis are moved to the output buffer via multicast. In general, the PEs along both axes can process different dimensions of the networks, including the input channels, output channels, feature map height, and feature map width, where the ordering of the subsequent operations and buffer read/write will determine the dataflow and are searchable in RT-ECG.

At any given time point, all sub-accelerators simultaneously process different clusters of the network layers with each sub-accelerator processing data of different input frames, where different layers within each sub-accelerator are executed sequentially to improve the throughput without the necessity of waiting. This is made possible because (1) sub-accelerators only communicate with the DRAM for fetching/storing the intermediate results and (2) an additional ping-pong buffer is introduced in the DRAM to accommodate simultaneous read/write. In this way, there are no communications needed among the sub-accelerators, leading to a more flexible and modular design. It is then possible to tailor the design of each sub-accelerator to better match the network structure and thus favor the achievable acceleration efficiency.

4.3.2 The Accelerator Search Space of Our DAS Engine. Based on the above accelerator template, we extract the searchable parameters, of which different combinations lead to different accelerators (i.e., micro-architecture and dataflow pairs), to form a generic accelerator space to be used by

Table 2. The Constructed Generic Accelerator Search Space Extracted from the Accelerator Template Introduced in Section 4.3.1

Memory Hierarchy	Loop-order	Loop-size
DRAM	TBS	-
Global Buffer	TBS	TBS
PE array	-	TBS
NoC design	**Max # of PEs**	**Layer assignment**
TBS	TBS	TBS

Where TBS Means "to be Searched" and the Searchable Parameters Include (1) the NoC Design, (2) Max # of PEs, (3) Layer Assignment, (4) Loop-order and (5) Loop-size Across Different Memory Hierarchies, i.e., the DRAM, Global Buffer, and PE Array.

our DAS engine. The micro-architecture is characterized by the number of memory hierarchies and PEs, the size of each memory hierarchy, the shape and size of the PE array, and the NoC design [15], and the dataflow is described by both the NoC design and the loop size/order. Specifically, we construct a generic accelerator search space as shown in Table 2 by leveraging the commonly used nested *for-loop* accelerator description [17, 60, 93, 95, 98] that naturally bridges the accelerator's micro-architectures and dataflows with DNNs' network parameters. Next, we introduce each accelerator parameter listed in Table 2:

Loop-order. The orders of the loops within each memory hierarchy, each of which has a total of n data dimensions. As such, n loops correspond to an n-item ordering problem. To be compatible with the proposed network search, where each accelerator parameter should have all possible choices parameterized by the corresponding γ vector (see Equation (6)), we formulate the *loop-order* search as a problem of picking one choice from a total of n options without replacement for n times (e.g., $n = 6$ considering the number of data dimensions in DNNs).

Loop-size. The size of each loop in the *for-loop* description. The product of all loop-sizes associated with each data dimension needs to be equal to the corresponding algorithmic dimension, because the nested loops' size as a whole dictates the total number of execution iterations. Then, intuitively, the possible choices for a certain loop's size are all the choices that the corresponding data dimension can be factorized into.

The NoC Design. The parallel execution patterns of the MAC operations when accelerating DNNs on an accelerator (e.g., those described in Section 4.3.1), which is determined by the PE array style. In this work, we consider three NoC options following the common practice, as inspired by SOTA accelerators [17, 95, 98]:

- Parallelizing the computation over the output partial sums, where the dimensions of output channels, output rows, and output columns are executed in parallel;
- Parallelizing the computation over the kernels, where the dimensions of output channels and input channels are executed in parallel;
- Parallelizing the computation over both the kernel and output dimensions, where the dimensions of output channels, kernel rows, and output columns are executed in parallel.

The Maximum Number of PEs. The maximal number of PEs in the design, which can range from 1 to a specified value determined by the area constraint and the tradeoff between the storage and computation partition. The PEs will be inter-connected with a pre-designed pattern according to the adopted NoC design, e.g., Figure 3 gives an example of parallelizing the kernels among the

PEs in the NoC across the input and output channel dimensions. In this work, where the latency is the primary objective, the maximum number of PEs is thus set to the hardware platform limit, e.g., the available **Digital Signal Processing units (DSPs)** in the given FPGAs. If other metrics like energy consumption are prioritized, then our proposed framework can automatically search for designs balancing the tradeoff between the consumed power and latency.

Layer assignment. The assignment of all the layers to be executed on a fixed number of sub-accelerators, which is set to 10 for this work, unless specified otherwise.

With maximum number of PEs fixed and all other parameters above taken into consideration, the space size can explode up to ~10^7.

4.3.3 The Search Algorithm of Our DAS Engine. To efficiently explore our constructed generic accelerator search space, our DAS engine iteratively updates the accelerator design choices in a differentiable manner. In particular, we parameterize the choice of each accelerator design factor with a vector γ and learn to update γ based on the objective formulated as:

$$\gamma^* = \min_{\gamma} \sum_{s=1}^{S} GS(\gamma^s)\, L_{cost}^{HW}(GS(\gamma^1), \ldots, GS(\gamma^S)), \tag{6}$$

where γ^s defines the probability distribution of the choices for the sth accelerator design parameter, $GS(\gamma^s)$ denotes Gumbel-Softmax sampling [28] of the sth accelerator parameter γ^s, and $L_{cost}^{HW}(GS(\gamma^1), \ldots, GS(\gamma^S))$ is the hardware cost of the target network on the sampled accelerator characterized by the S sampled design factors $GS(\gamma^1), \ldots, GS(\gamma^S)$. To be more specific, we apply Gumbel-Softmax sampling [28, 53] to sample only one choice $GS(\gamma^s)$ from all the options corresponding to the sth accelerator parameter. Once all the accelerator parameters are sampled, the corresponding accelerator's acceleration cost is estimated using SOTA accelerator performance estimators, where in this work, we adopt the performance estimator in Reference [89] for our prototyped FPGA-based accelerators. After that, we multiply the resulting acceleration cost by the sampled $GS(\gamma^s)$ and update the γ based on the continuous relaxation of Gumbel-Softmax during backward pass [82] for gradient estimation. When the gradient-based optimization converges, we derive the final accelerator by selecting the parameter options with the highest probability (i.e., γ_s) for each accelerator parameter. Note that we use the number of MACs as the complexity cost during the network search stage (see L_{cost}^{MAC} in Equation (4)) for better search efficiency, and we adopt the estimated accelerator cost L_{cost}^{HW} during the accelerator search stage to better align with the actual acceleration cost.

4.4 RT-RCG: The Complexity and Time Cost of the DNS and DAS Engines

4.4.1 The Complexity of the DNS and DAS Engines. The algorithm complexity of our DNS engine is tied with that of the supernet training, because we adopt the DNAS algorithm as mentioned in Section 4.2 where the supernet weights and network architecture parameters are updated at the same time. Additionally, picking the final network structure with the highest probability requires an additional complexity of O(k), where k denotes the number of possible operations per block and equals 9 considering our search space defined in Section 4.2. However, the entire DNS process, including re-training the final picked network, can finish within a GPU hour of 0.5, given the DNS search space size of 9^{14} in this work.

The algorithm complexity of our DAS engine is proportional to that of the Gumbel-Softmax, which is O(n) with n denoting the number of choices for each hardware design parameter. Thanks to the efficient hardware cost estimator [89], the entire DAS process only takes about 10 minutes with our space size being 10^7.

Note that our differentiable search method enables a much more directed and efficient search trajectory. Thus, there is no need to exhaustively evaluate every design choice within the search space, leading to a much shorter search time than that of an exhaustive search. Additionally, the search is terminated when the minimized objectives become stable.

4.4.2 The Amortized One-time Search Cost. For a given task, e.g., ECG reconstruction for a specific patient, merely a one-time effort is required to generate the network structure and its accelerator, and thus the search time cost is amortized throughout the implementation. Once the network structure and accelerator design are, respectively, generated by the DNS and DAS engine, they will be fixed throughout the task. If there are minor changes to the task settings like the patient's heart conditions, then the network's parameters (weights) can be fine-tuned with the patient's newly generated heart samples without the necessity of changing the network structure and accelerator design. The fine-tuning process can be conducted using a standard DNN training procedure on an external computer in a few minutes, considering the setup described in Section 5.1. Basically, the search only needs to be redone when there are necessary drastic changes, e.g., the change of the entire training dataset.

4.4.3 Generalization of the Searched Designs. The searched network structure and its accelerator together with the final fully trained network weights can be generalized to distinct patients' heart samples if the search and training is conducted on a diverse patient dataset, i.e., the searched designs (i.e., the network structure, accelerator design, and trained parameters) are expected to be effective for new patients, which are not present in the pre-trained dataset. As such, no additional cost or complexity are incurred for this generalization, as the original search and training process can holistically take the diverse training dataset into consideration. This is validated in Section 5.3, where the searched networks and accelerator designs consistently perform well on the newly included patients. This generalization capability can be significantly meaningful to real-life applications, where collecting data samples for new patients may not always be possible, and the search cost can thus be amortized across different patients.

5 EXPERIMENT RESULTS

In this section, we present the evaluation results of our proposed RT-RCG framework. Starting with the introduction to our dataset and experiment setup, we evaluate the effectiveness of the RT-RCG searched networks under various settings, including (1) patient-specific reconstruction (see Section 5.2), (2) reconstruction generalized to a new patient (see Section 5.3), and (3) robust reconstruction with deficient EGM channels (see Section 5.4). After that, we evaluate RT-RCG's hardware acceleration performance as compared to two SOTA DNN accelerators [85, 97], one edge platform [58], and a CPU platform, followed by the ablation studies on the initial latency and under a constrained search space.

5.1 Experiment Setup

Clinically Collected Dataset. To evaluate the effectiveness of the RT-RCG framework, data was collected retrospectively from 14 patients undergoing cardiac ablation for premature ventricular contractions, where both the ECG and EGM signals were recorded simultaneously during the cardiac ablation procedure and each record of the database is composed of:

- Twelve standard surface ECG channels, namely, leads I, II, III, aVR, aVL, aVF, and V1:V6.
- Five EGM channels measured by electrodes on a catheter placed inside the Coronary Sinus.

Specifically, the data was obtained from patients undergoing cardiac ablation procedures and was retrospectively collected under a protocol approved by an institutional review board at Baylor St.

Table 3. The Number of Heartbeat Samples for Each Patient and the Patient ID in Our Clinically Collected Dataset

Patient ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Number of heartbeats (N)	4,765	2,309	401	1,752	3,934	3,017	2,593	6,635	3,102	2,326	5,497	1,591	1,827	2,917

Luke's Medical Center [5]. During these procedures, the routine is to record both the surface ECG and the EGM signals via the mapping catheter. For each patient, the EGM was obtained from the coronary sinus. By virtue of the procedure, the recordings for each patient are of different lengths and contain a mix of sinus rhythms and diseased heartbeats, providing a diverse dataset to better emulate real-world scenarios while also making it more challenging to achieve high performance reconstruction on this dataset. This also means that the number of heartbeats (i.e., N in Table 3) are different for different patients. In our experiment, the data for each patient was first randomly shuffled and then segmented into halves, with the first half of concurrent ECGs and EGMs being used during the search/training step and the second half for testing and performance evaluation. The patient number and corresponding number of heartbeats are summarized in Table 3.

Algorithm Experiment Setup. Algorithm training setup: All the DNN training is carried out on a machine with one NVIDIA 2080TI GPU and an AMD EPYC 7742 64-Core power processor. Throughout the training, we use an Adam optimizer with a batch size of 16, a learning rate of 1E-3, and a weight decay factor of 1E-3. During the training, we incorporate the Pearson correlation coefficient between the network output and the ground truth (i.e., corresponding real-measured ECG signals) into the loss function (see Equation (3) in Section 4.1). Network search setup: We adopt the one-level optimization as in References [36, 84] and a fixed temperature of 1 for the Gumbel-Softmax function. We reuse the above training setting for the supernet weights and adopt an Adam optimizer with a constant learning rate of 1E-3 for the architecture parameters. We then derive the operations with the highest probability for each searchable block at the end of the search. Algorithm evaluation setup: To evaluate the reconstruction efficacy, we calculate the correlation between the reconstructed ECG signals and the real-measured ones on the half of the dataset for testing and performance evaluation. Specifically, we first convert the network output that is in the time-frequency domain to its time domain counterpart using the inverse STFT, and then calculate the Pearson correlation coefficient between the reconstructed signals and the original ECG signals, which are time-series waveforms.

Accelerator Experiment Setup. Accelerator search setup: Considering the real-time reconstruction goal, we adopt the commonly used **Frames Per Second (FPS)** metric. However, other metrics can be easily plugged into our RT-RCG framework, depending on the specification of the target applications and the user-specified preference. During the accelerator search process, RT-RCG makes use of a SOTA accelerator performance predictor AutoDNNChip [89] to obtain a fast and reliable estimation to guide the search towards the optimal solution. Accelerator evaluation setup: For evaluating FPGA-based accelerators, we adopt a Xilinx ZC706 evaluation board [87] with the same DSP limit as the baselines [85, 97] for a fair comparison. Specifically, we adopt a standard Vivado HLS design flow [86], where the FPS is obtained from the HLS synthesis results for our searched accelerators and the baseline ChaiDNN [85]. For DNNBuilder [97], we utilize their open source simulator to obtain its acceleration results. For the CPU baseline, we evaluate the achieved FPS of the networks being executed on an AMD EPYC 7742 64-Core CPU. For the edge platform baseline, we consider a commonly used edge device [48, 68, 80], i.e., the NVIDIA Edge GPU Jetson TX2 [58], where the networks are compiled using TensorRT [59], a C++ library for high-performance inference on NVIDIA GPUs. Additionally, the device is configured to be in max-N mode to make full use of the available resources following [80].

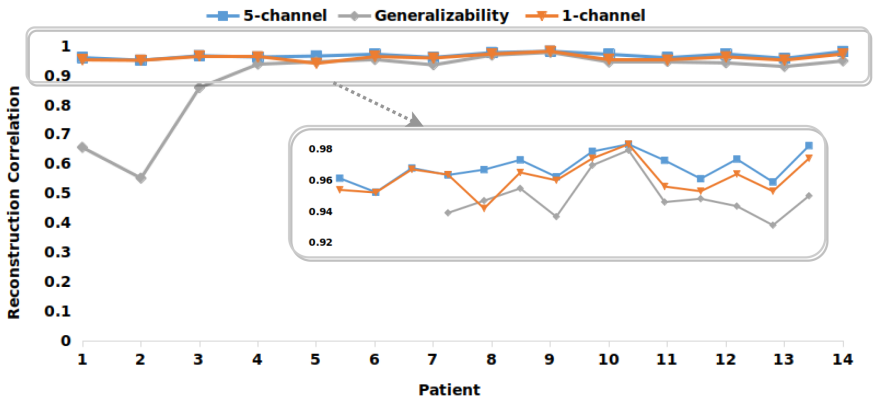

Fig. 4. The average Pearson correlation coefficient between RT-RCG's reconstructed and real-measured ECG time-series signals across all the 14 patients in our dataset, when considering (1) **Blue**: patient-specific reconstruction from five channels of EGM (see Section 5.2), (2) **Grey**: reconstruction generalized to new patients (see Section 5.3), and (3) **Orange**: patient-specific reconstruction with merely one EGM channel (see Section 5.4).

5.2 RT-RCG's Searched Algorithms: Patient-specific Reconstruction

In this subsection, we evaluate RT-RCG's searched networks in a patient specific setting, where all the search, training, and testing are based on the data collected from the same patients. This is to mimic the case where the pacemakers are customized to each patient. Specifically, for our clinical dataset, which contains sinus and diseased heartbeats of the 14 patients, we equally split it into two subsets for training and testing, respectively.

To thoroughly evaluate RT-RCG's searched networks, we consider all of the 14 patients in a patient-specific manner and plot the resulting correlation (between the constructed ECG and the real-measured ECG signals) in Figure 4 (the blue curve). We can see that the ECG signals reconstructed by RT-RCG's searched networks are highly correlated with the real-measured ones across all of the 14 patients, as evidenced by the resulting Pearson correlation coefficient value ranging from $0.952 \sim 0.983$, which is much improved as compared to the correlation value of 0.84 achieved with the SOTA method [63] using time delay neural networks. This improvement implies that RT-RCG's searched networks can accurately predict ECGs that are close to the corresponding real-measured ones as compared to the ones reconstructed by the SOTA method in Reference [63].

5.3 RT-RCG's Searched Algorithms: Reconstruction Generalized to New Patients

In this subsection, we evaluate the efficacy of our RT-RCG's searched networks when being generalized to new patients. Specifically, the networks are searched and trained based on the data of all patients with one of the patients excluded and then tested on the excluded patient. By doing so, this experiment can evaluate the searched networks' generalization capability to unseen new patients, i.e., how well the networks dedicated to a set of patients can perform when adapted to other patients. As shown in the grey curve in Figure 4, the correlation between the reconstructed and real-measured ECG signals is consistently higher than 0.93, except for Patients 1, 2, and 3, whose heartbeat samples are very distinct from the remaining ones, implying the importance of searching/training the algorithms on diverse patients before being generalized to other patients to ensure the efficacy. Overall, the above experiments indicate the excellent generalization capability of RT-RCG's searched networks. We can expect improved performance if RT-RCG's searched networks are obtained based on more data with diverse ventricular conditions, paving the way for

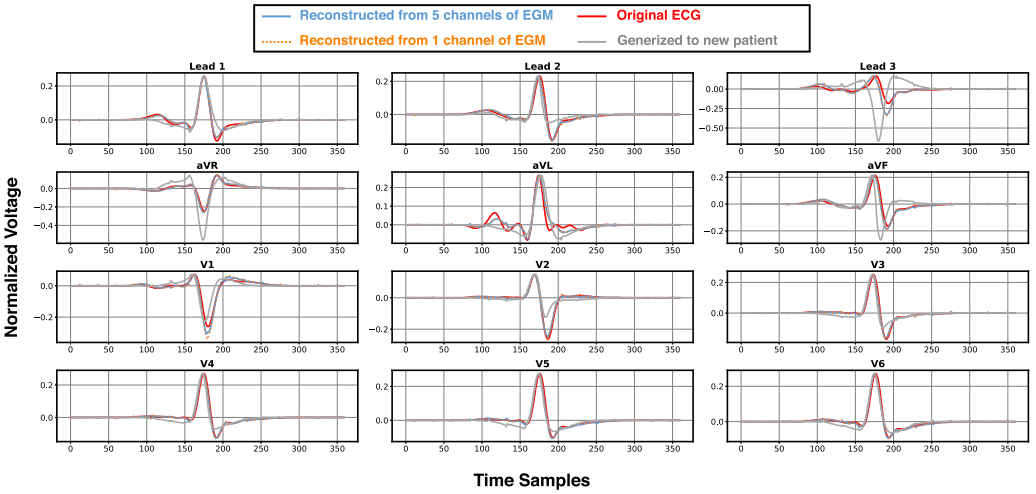

Fig. 5. Visualizing the reconstructed ECG signals under different experiment settings together with the corresponding real-measured ones for Patient 4, where the x axis is the time sample and the y axis is the normalized voltage of the waveforms.

developing "one-for-all" reconstruction algorithms that can save a large amount of the time and effort needed to collect data for each target patient; this is particularly useful when pre-collecting data for the target patient is not possible.

5.4 RT-RCG's Searched Algorithms: Reconstruction Robustness under EGM Deficiency

In practice, pacemakers only utilize 1–5 EGM channels and it is an imperative function of pacemakers to work with only one channel of EGM. Aiming towards practical uses, we thus evaluate our RT-RCG's searched networks under such scenarios, considering the most extreme case where only one out of the five EGM channels is available. Specifically, we search and train the networks based on data with only one EGM channel and evaluate the correlation between the reconstructed and real-measured ECG signals under the patient-specific setting (similar to Section 5.2). While we observe consistent results when picking different EGM channels as the one to be used, we here show the observations when picking the first channel. As shown in the orange curve (i.e., "1-channel") in Figure 4, we can see that the reconstruction quality under this extreme scenario is surprisingly close to that of the normal setting with all EGM channels on, achieving a correlation ranging from 0.942 to 0.983. Furthermore, Figure 5 shows that the reconstructed ECG from only one channel of EGM does not have noticeable degradation when compared with the original ECG signals. This set of experiments demonstrates the excellent robustness of RT-RCG's searched networks in the presence of EGM channel deficiency.

5.5 RT-RCG's Searched Algorithms: Visualizing the Searched Network and Reconstructed ECG Signals

To better understand and visualize RT-RCG's searched networks, here, we provide a visualization to show RT-RCG's searched network and RT-RCG's reconstructed ECG signals. First, as an illustrative example, we visualize the searched network for Patient 4 under a constraint of 28.87M MACs, as illustrated in Figure 6(c). In particular, this searched network contains 36 layers excluding the pooling and upsampling layers and a total of 28.87M MACs. In addition, the searched networks

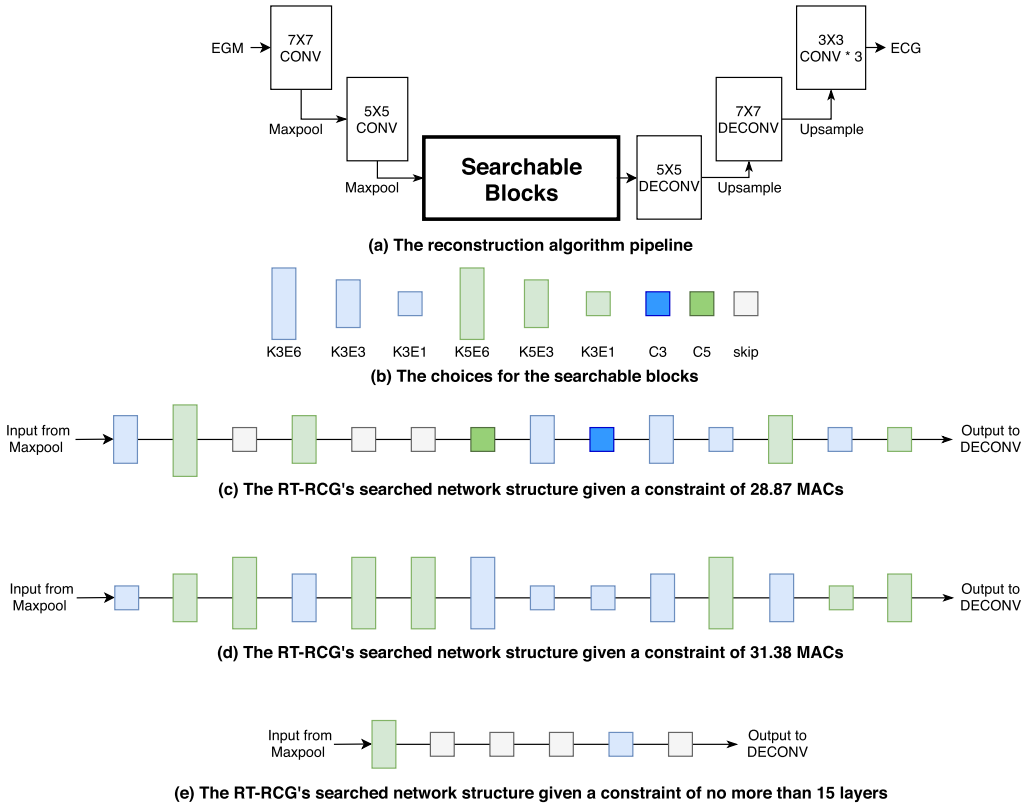

Fig. 6. An illustration of the (a) reconstruction algorithm pipeline, consisting of the fixed earlier blocks, searchable blocks, and fixed later blocks, (b) choices for the searchable blocks following Reference [82], and the RT-RCG's searched network structures when given a constraint of (c) 28.87 MACs, (d) 31.38 MACs, and (e) no more than 15 layers. In (b), KaEb denotes a convolutional building block with a kernel size of a and a channel expansion ratio of b, and Ca denotes a standard convolution layer with a kernel size of a.

under different MACs constraints are similar in terms of the kernel size and expansion ratio choices, yet with different preferences in the networks' depth. As shown in Figure 6(d), when the number of MACs is increased to 31.38M, the proposed DNS opts to reduce the frequency of skip connections, while the layer structures in terms of kernel sizes and expansion ratios are similar to those under a constraint of 28.87M MACs. Second, Figure 5 visualizes the reconstructed ECG signals of RT-RCG's searched networks under various settings, when the reconstruction is performed using (1) 5 EGM channels and (2) 1 EGM channel, or generalized to new patients. While we observe consistent results across different patients, here, we only show the visualization for Patient 4 for a better illustration. We can see that the reconstructed ECG signals are close to the real-measured ones when the network structure in Figure 6(c) is used, with the largest deviation happening when the algorithm is generalized to a new patient, as expected.

5.6 RT-RCG's Searched Accelerators: Achieved FPS over SOTA DNN Accelerators/Platforms

In this subsection, we evaluate RT-RCG's searched accelerators by comparing their achieved FPS with that of (1) two SOTA DNN accelerators (DNNBuilder [97] and ChaiDNN [85]), (2) the

Table 4. The Achieved FPS of the RT-RCG's Searched Accelerator and the Four SOTA DNN Accelerators/platforms Given the Same Network (See Figure 6(c)), Network Bit Precision, and Clock Frequency (Except for the Edge GPU and CPU Cases)

Platform	Clock frequency	# of PEs	Bit precision	FPS
DNNBuilder [97]	200 MHz	435	16	228
RT-RCG	**200 MHz**	**428**	**16**	**427 (1.87×)**
ChaiDNN [85]	200 MHz	212	8	401
RT-RCG	**200 MHz**	**185**	**8**	**696 (1.73×)**
Jetson-TX2 [58]	1.3 GHz	/	32	1,190 (183.07 @ 200 MHz)
RT-RCG	**200 MHz**	**870**	**32**	**229** (0.19× w/1.3 Ghz; **1.22×** w/200 Mhz)
CPU [2]	2.25 GHz	/	32	21 (3.23 @ 200 MHz)
RT-RCG	**200 MHz**	**870**	**32**	**229** (11.39× w/1.3 Ghz; **70.90×** w/200 Mhz)

Where the number of PEs indicates the peak usage of the processing elements, corresponding to the number of used DSPs for FPGA-based accelerators.

Table 5. The Resulting Subgroups for the DNNBuilder Implementation of the Searched Network Shown in Figure 6(c), which is to Enable DNNBuilder's Feasible Implementation of DNNs with over 15 Layers

Group ID	1	2	3	4	5	6	7	8	9
Layer ID	(1)	(2, 3)	(4 ∼ 13)	(15 ∼ 18)	(19 ∼ 26)	(27 ∼ 33)	(34)	(35)	(36)

Note that each subgroup assumes one pipeline stage and layers within each subgroup share the same pipeline stage.

edge GPU (Jetson TX2 [58]), and (3) a general DNN deployment platform (an AMD EPYC 7742 64-Core CPU [2]) under the same conditions. Specifically, we ensure that the reconstruction algorithm (i.e., the searched network for Patient 4 under 28.87M MACs, as shown in Figure 6(c)) and the network precision be the same as the baselines'. The comparison results are summarized in Table 4. We can see that RT-RCG's searched accelerator consistently achieves a better FPS than all of the four baselines, based on the same network structure and hardware constraints. Specifically, the RT-RCG searched accelerator improves the achieved FPS, which in turn can be translated to processed heartbeat samples per second, by 1.87×, 1.73×, 1.22×, and 70.90×, as compared to the DNNBuilder, ChaiDNN, the edge GPU, and the CPU, respectively. This set of experiments indicates that the integrated DAS engine of RT-RCG is effective and RT-RCG's automatically searched accelerator can even outperform expert-designed SOTA DNN accelerators, paving the way for the fast development of reconstruction accelerators.

More details regarding the experiment settings for each baseline are described below:

DNNBuilder. For the comparison with the SOTA DNN accelerator named DNNBuilder, we adopt a DSP limit of 450, a 16-bit precision, and a frequency of 200 MHz to be the same as the original setting in DNNBuilder [97]. As the reported DNNBuilder design uses a layer-wise pipeline micro-architecture, it requires to constrain the maximum number of DNN layers to be smaller than 15, for meeting the DRAM access bandwidth constraint, as shown in their open source codes [97]. To support RT-RCG's searched networks, which has more than 15 layers, we first divide the network into 9 subgroups with each having some layers from the original network processed sequentially and then execute these subgroups in a pipeline fashion based on the open source design of DNNBuilder. The subgroups are formed to balance the latency among them and thus maximize the achieved throughput of DNNBuilder given the specific DNN structure. Specifically, the 9 subgroups for the network (see Figure 6(c)) are shown in Table 5. Note that we also evaluate

Table 6. The Start-up Latency and FPS of the RT-RCG Generated Accelerator Given the Network Generated for Patient 4 (See Section 5.2) under Different Platforms

Platform	# of PEs	Start-up latency (ms)	FPS
ChaiDNN [85]	212	3.01	401
RT-RCG	185	3.29 (+9.3%)	696 (+73.6%)
RT-RCG-latency	**171**	**2.39 (+20.6%)**	**419 (+4.5%)**

RT-RCG's searched accelerators over DNNBuilder when constraining RT-RCG's network search space to have networks with smaller than 15 layers, as discussed in Section 5.8.

ChaiDNN. We also benchmark RT-RCG's searched accelerator with another SOTA FPGA DNN accelerator named ChaiDNN [85], with its DietChai_z variant enabled to optimize its performance under more resource-constrained scenarios. Specifically, we select its 128-compute-DSP mode that results in a DSP limit of 212 when accelerating the given searched network.

Jetson TX2. When comparing with the edge GPU Jetson TX2, which is a commonly used IoT device, we set the DSP limit to be 900 (the maximum amount available), so our implementations have roughly the same power consumption as the edge GPU Jetson TX2. Note that the operating clock frequency of Jetson TX2 is 1.3 GHz, which is far higher than the maximum supported stable frequency of our platform ZC706. We thus scale the Jetson TX2's throughput to that corresponding to a frequency of 200 MHz for a fair comparison as shown in Figure 6(c), under which the achieved FPS of the RT-RCG's searched accelerator outperforms the edge GPU by 1.22×.

CPUs. Considering that CPUs are currently the mainstream computing platforms, we also evaluate RT-RCG's searched accelerator over an AMD EPYC 7742 64-Core processor given the same network. For a fair comparison, we adopt a DSP limit of 900, which is the maximum available DSP resource on our adopted ZC706 board. Note that the power consumption of the CPU is ∼225W, which significantly dwarfs that of the ZC706 board, which is ∼10W.

Discussion and Implication. There are several levels of implication from our experiments (including the latency evaluation in Table 6). First, we can see that our proposed RT-RCG indeed can automatically generate (1) reconstruction networks that can provide high-quality reconstruction, which outperforms SOTA techniques and has excellent generalization capability, and (2) accelerators to run the reconstruction networks achieving a better acceleration efficiency than diverse SOTA accelerators/platforms under the same conditions. Second, the performance achieved by RT-RCG shows that it is indeed possible for doctors to remotely monitor the status of pacemakers and patients via reconstructed ECG signals, given the achieved FPS. Specifically, in our case, such real-time monitoring is possible, as the achieved FPS (229 ∼ 606 FPS in our proposed RT-RCG) is much higher than the required 2 FPS (the highest input rate in our dataset is 2 Hz, as it requires at least 0.5 s to collect each piece of input). More importantly, the high FPS achieved is necessary, as it implies that real-time intervention is possible, especially considering that certain cardiac patients, particularly patients diagnosed with lethal ventricular arrhythmias, under which the higher the FPS, the sooner doctors can respond to provide the necessary intervention in life-critical situations. Despite the promising reconstruction efficacy and efficiency achieved by RT-RCG, our effort in this article is merely a heuristic step towards next-generation pacemakers equipped with real-time monitoring and intervention. In particular, the energy cost of the RT-RCG framework currently implemented on FPGA is still way higher than the stringent energy consumption required by the pacemakers. We recognize that applying RT-RCG searched networks and accelerators to real-world pacemakers would require ultra-energy-efficient ASIC implementation, which we leave as one of our most exciting future works.

Table 7. The Reconstruction Accuracy of the Searched Network with the Constraint of <15 Layers, Compared with the Searched Network in Figure 6(c) Searched without the Layer Number Constraint

Patient ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14
36-layer-net	0.9613	0.9524	0.9678	0.9634	0.9668	0.9730	0.9622	0.9784	0.9830	0.9727	0.9610	0.9735	0.9589	0.9821
15-layer-net	**0.9601**	**0.9463**	**0.9641**	**0.9640**	**0.9674**	**0.9714**	**0.9626**	**0.9762**	**0.9829**	**0.9656**	**0.9571**	**0.9620**	**0.9580**	**0.9824**
Improvements from 36-layer	**-0.0012**	**-0.0061**	**-0.0037**	**0.00057**	**0.00058**	**-0.0016**	**0.00041**	**-0.0022**	**-0.00011**	**-0.0071**	**-0.0039**	**-0.011**	**-0.00088**	**0.00026**

Table 8. RT-RCG's Searched Accelerators vs. DNNBuilder When Constraining the Networks to Have Fewer Than 15 Layers

Platform	# of PEs	Network MACs (M)	FPS
DNNBuilder-36-layer	435	28.87	228
DNNBuilder-15-layer	441	24.23	340 (+49.1%)
RT-RCG-36-layer	**428**	**28.87**	**427 (+87.3%)**
RT-RCG-15-layer	**433**	**24.23**	**447 (+96.1%)**

5.7 RT-RCG's Searched Accelerators: Achieved Latency over SOTA DNN Accelerators/Platforms

As ECG signals can be used to detect irregular ventricular rhythms that trigger a corresponding alert mechanism [70], where the latency from the occurrence of the rhythms to the mechanism being triggered, denoted as start-up latency, can be of great significance to the patients' health and life, the latency of the EGM-ECG conversion contributing a considerable portion of the whole pipeline is thus important. Therefore, we also evaluate RT-RCG's searched accelerators over SOTA DNN accelerators/platforms in terms of this latency. Note that the achieved start-up latency and FPS have a tradeoff relationship. An advantage of our RT-RCG framework is that users can customize their own desired tradeoff given their priority and conditions. As shown in Table 6, we provide two searched accelerators of RT-RCG, which favor the achieved start-up latency and FPS, respectively, with the former achieving a 27.36% better start-up latency at a cost of 38.8% lower FPS. We can see that RT-RCG's automatically searched accelerators achieve a smaller start-up latency as compared to the baseline under the same hardware constraint, i.e., 20.60% over the expert-designed accelerator ChaiDNN [85]. This set of experiments again validates the effectiveness of our RT-RCG framework's DAS engine.

5.8 RT-RCG's Searched Accelerators: Constrained Networks with <15 Layers

As mentioned in Section 5.6, the baseline DNNBuilder [97] adopts a layer-wise acceleration micro-architecture, which favors networks with fewer than 15 layers. To validate the general efficacy of our RT-RCG framework, here, we present experiments where we constrain the network search space to ensure that the searched networks have fewer than 15 layers and then compare the acceleration performance of RT-RCG's searched accelerator with that of the DNNBuilder baseline under the patient-specific setting. Specifically, we adaptively adjust λ in Equation (4) when the depth of the derived network surpasses 15 layers by doubling λ. As shown in Figure 6(e), with the number of layers being constrained to 15, the searched network contains only a 16.07% lower number of MACs as compared to the unconstrained case (see Figure 6(c)), while Table 7 indicates that our RT-RCG framework's DNS engine is able to adapt to different constraints while maintaining the networks' performance (i.e., reconstruction quality in terms of the correlation): 0.9601 vs. 0.9613 for Patient 4. In particular, RT-RCG results in a wider network under this depth constraint to maintain the network capacity and thus reconstruction efficacy. Meanwhile, as shown in Table 8, we can see that (1) DNNBuilder's achieved FPS is improved by 49.1% as compared to

the unconstrained case presented in Section 5.6, which has a 36-layer network, under the same DSP constraint, and (2) RT-RCG's automatically searched accelerator again outperforms the expert designed accelerator DNNBuilder with a 23.94% higher FPS. This set of experiments together with the ones in Section 5.6 validates the general effectiveness of our RT-RCG framework across different network search spaces and accelerated networks.

6 CONCLUSION

The costly and time-consuming hospital visits required for patients with implanted pacemakers and the recent advances in the IoT technologies have motivated an increasing need for remote monitoring of pacemakers to reduce hospital visit costs and to provide continuous monitoring and potential real-time intervention, which can be life-critical under some irregular and infrequent ventricular rhythms. However, the signals provided by pacemakers and the ones doctors use for diagnosis during in-person clinical visits are different, with the former being EGM signals and the latter being ECG signals, calling for high-quality and real-time ECG reconstruction from the recorded EGM signals. To this end, we propose, design, and validate a first-of-its-kind framework dubbed RT-RCG, which can automatically search for (1) efficient DNN structures and then (2) corresponding hardware accelerators to implement the ECG-EGM reconstruction process, respectively, tackling both the reconstruction efficacy and efficiency. Specifically, RT-RCG integrates a new DNN search space tailored for required ECG-EGM reconstruction to enable automated search for DNNs that conduct ECG reconstruction with much improved quality over SOTA solutions and incorporates a differentiable acceleration search engine that can automatically generate optimal accelerators to accelerate the resulting DNNs from the previous step. Extensive experiments and ablation studies under various settings consistently validate the effectiveness and advantages of the proposed RT-RCG at leading to higher reconstruction quality and better reconstruction efficiency as compared to SOTA reconstruction algorithms and DNN accelerators. Our RT-RCG has made the first heuristic step towards automated generation of ECG-EGM reconstruction DNNs along with the matched accelerators, which enable real-time critical intervention in instant response to irregular and infrequent ventricular rhythms that require timely treatment, paving the way for more pervasive remote monitoring of the pacemakers via ECG-EGM reconstruction.

REFERENCES

[1] Mohamed S. Abdelfattah, Łukasz Dudziak, Thomas Chau, Royson Lee, Hyeji Kim, and Nicholas D. Lane. 2020. Best of both worlds: AutoML codesign of a CNN and its hardware accelerator. *arXiv preprint arXiv:2002.05022* (2020).

[2] AMD Inc. 2020. 2nd Gen AMD EPYC™ 7742 | Server Processor | AMD. Retrieved from https://www.amd.com/en/products/cpu/amd-epyc-7742.

[3] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Erich Elsen, Jesse Engel, Weiwei Fang, Linxi Fan, Christopher Fougner, Liang Gao, Caixia Gong, Awni Hannun, Tony Han, Lappi Vaino Johannes, Bing Jiang, Cai Ju, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangang Li, Dongpeng Ma, Sharan Narang, Andrew Ng, Sherjil Ozair, Yiping Peng, Ryan Prenger, Sheng Qian, Zongfeng Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Srinet, Anuroop Sriram, Haiyuan Tang, Liliang Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, and Zhenyao Zhu. 2016. Deep speech 2: End-to-end speech recognition in English and Mandarin. In *International Conference on Machine Learning*. 173–182.

[4] Lanfranco Antonini, Antonio Auriti, Vincenzo Pasceri, Antonella Meo, Christian Pristipino, Antonio Varveri, Salvatore Greco, and Massimo Santini. 2012. Optimization of the atrioventricular delay in sequential and biventricular pacing: Physiological bases, critical review, and new purposes. *Europace* 14, 7 (2012), 929–938.

[5] Baylor College of Medicine. 2020. Baylor St. Luke's Medical Center. Retrieved from https://www.bcm.edu/about-us/affiliates/baylor-st-lukes-medical-center.

[6] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*. Springer, 1–4.

[7] Paschalis Bizopoulos and Dimitrios Koutsouris. 2018. Deep learning in cardiology. *IEEE Rev. Biomed. Eng.* 12 (2018), 168–193.
[8] Anh L. Bui, Tamara B. Horwich, and Gregg C. Fonarow. 2011. Epidemiology and risk profile of heart failure. *Nat. Rev. Cardiol.* 8, 1 (2011), 30.
[9] Han Cai, Ligeng Zhu, and Song Han. 2018. ProxylessNAs: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*.
[10] Deming Chen, Jason Cong, Yiping Fan, Guoling Han, Wei Jiang, and Zhiru Zhang. 2005. xPilot: A platform-based behavioral synthesis system. *SRC TechCon* 5.
[11] Deming Chen, Jason Cong, Yiping Fan, and Lu Wan. 2009. LOPASS: A low-power architectural synthesis system for FPGAs with interconnect estimation and optimization. *IEEE Trans. Very Large Scale Integ. (VLSI) Syst.* 18, 4 (2009), 564–577.
[12] Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2020. AdaBERT: Task-adaptive BERT compression with differentiable neural architecture search. *arXiv preprint arXiv:2001.04246*.
[13] Liang-Chieh Chen, Maxwell Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jon Shlens. 2018. Searching for efficient multi-scale architectures for dense image prediction. In *Conference on Advances in Neural Information Processing Systems*. 8699–8710.
[14] Wuyang Chen, Xinyu Gong, Xianming Liu, Qian Zhang, Yuan Li, and Zhangyang Wang. 2019. FasterSeg: Searching for faster real-time semantic segmentation. *arXiv preprint arXiv:1912.10917*.
[15] Y. Chen, T. Krishna, J. Emer, and V. Sze. 2017. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE J. Solid-state Circ.* 52, 1 (2017), 127–138.
[16] Y. Chen, Y. Maguire, C. Tapscott, C. Chivetta, Y. Chen, B. Aazhang, J. Cavallaro, and M. Razavi. 2018. An energy harvesting wireless leadless multisite pacemaker prototype. In *52nd Asilomar Conference on Signals, Systems, and Computers*. 218–222. DOI: https://doi.org/10.1109/ACSSC.2018.8645421
[17] Yu-Hsin Chen, Joel Emer, and Vivienne Sze. 2016. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. *ACM SIGARCH Comput. Archit. News* 44, 3 (2016), 367–379.
[18] Yu-Hsin Chen, Tien-Ju Yang, Joel Emer, and Vivienne Sze. 2019. Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices. *IEEE J. Emerg. Select. Topics Circ. Syst.* 9, 2 (2019), 292–308. DOI: 10.1109/JETCAS.2019.2910232
[19] Kanghyun Choi, Deokki Hong, Hojae Yoon, Joonsang Yu, Youngsok Kim, and Jinho Lee. 2020. DANCE: Differentiable accelerator/network co-exploration. *arXiv preprint arXiv:2009.06237*.
[20] Romain Cosentino, Randall Balestriero, Richard Baraniuk, and Behnaam Aazhang. 2020. Provable finite data generalization with group autoencoder. *arXiv preprint arXiv:2009.09525*.
[21] Zidong Du, Robert Fasthuber, Tianshi Chen, Paolo Ienne, Ling Li, Tao Luo, Xiaobing Feng, Yunji Chen, and Olivier Temam. 2015. ShiDianNao: Shifting vision processing closer to the sensor. In *42nd Annual International Symposium on Computer Architecture*. 92–104.
[22] Andoni Elola, Elisabete Aramendi, Unai Irusta, Artzai Picón, Erik Alonso, Pamela Owens, and Ahamed Idris. 2019. Deep neural networks for ECG-based pulse detection during out-of-hospital cardiac arrest. *Entropy* 21, 3 (2019), 305.
[23] G. Eraslan, L. M. Simon, M. Mircea, N. S. Mueller, and F. J. Theis. 2019. Single-cell RNA-seq denoising using a deep count autoencoder. *Nat. Commun.* 10 (2019), 1–14.
[24] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio. 2010. Why does unsupervised pretraining help deep learning? *J. Mach. Learn. Res.* 11 (2010), 625–660.
[25] Yonggan Fu, Wuyang Chen, Haotao Wang, Haoran Li, Yingyan Lin, and Zhangyang Wang. 2020. AutoGAN-distiller: Searching to compress generative adversarial networks. *arXiv preprint arXiv:2006.08198*.
[26] M. Gentil, F. Porée, A. I. Hernández, and G. Carrault. 2005. Surface electrocardiogram reconstruction from cardiac prothesis electrograms. *EMBEC05* (2005), 2028F1–6.
[27] Y. Guan, H. Liang, N. Xu, W. Wang, S. Shi, X. Chen, G. Sun, W. Zhang, and J. Cong. 2017. FP-DNN: An automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. In *IEEE 25th Annual International Symposium on Field-programmable Custom Computing Machines (FCCM)*. 152–159.
[28] Emil Julius Gumbel. 1948. *Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures*. Vol. 33. US Government Printing Office.
[29] Kazım Hanbay. 2018. Deep neural network-based approach for ECG classification using hybrid differential features and active learning. *IET Sig. Process.* 13, 2 (2018), 165–175.
[30] Awni Y. Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H. Tison, Codie Bourn, Mintu P. Turakhia, and Andrew Y. Ng. 2019. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nat. Med.* 25, 1 (2019), 65.
[31] Cong Hao, Xiaofan Zhang, Yuhong Li, Sitao Huang, Jinjun Xiong, Kyle Rupnow, Wen-mei Hwu, and Deming Chen. 2019. FPGA/DNN Co-Design: An efficient design methodology for IoT intelligence on the edge. In *56th Annual Design*

Automation Conference (DAC'19). Association for Computing Machinery, New York, NY. DOI: https://doi.org/10.1145/3316781.3317829

[32] Chaoyang He, Haishan Ye, Li Shen, and Tong Zhang. 2020. Milenas: Efficient neural architecture search via mixed-level reformulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11993–12002.

[33] Melonie Heron and Robert N. Anderson. 2016. Changes in the leading cause of death: Recent patterns in heart disease and cancer mortality. *NCHS Data Brief* 254.

[34] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. 2019. Searching for mobileNetV3. In *Proceedings of the IEEE International Conference on Computer Vision*. 1314–1324.

[35] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

[36] Shoukang Hu, Sirui Xie, Hehui Zheng, Chunxiao Liu, Jianping Shi, Xunying Liu, and Dahua Lin. 2020. DSNAS: Direct neural architecture search without parameter retraining. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12084–12092.

[37] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. 2019. *Automated Machine Learning*. Springer.

[38] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*.

[39] Weiwen Jiang, Qiuwen Lou, Zheyu Yan, Lei Yang, Jingtong Hu, X. Sharon Hu, and Yiyu Shi. 2020. Device-circuit-architecture co-exploration for computing-in-memory neural accelerators. *IEEE Trans. Comput.* 70, 4 (2020), 595–605. DOI: 10.1109/TC.2020.2991575

[40] Weiwen Jiang, Lei Yang, Edwin Sha, Qingfeng Zhuge, Shouzhen Gu, Yiyu Shi, and Jingtong Hu. 2019. Hardware/Software co-exploration of neural architectures. *arXiv preprint arXiv:1907.04650*.

[41] Amar Kachenoura, Fabienne Porée, Guy Carrault, and A. I. Hernández. 2009. Non-linear 12-lead ECG synthesis from two intracardiac recordings. In *36th Annual Computers in Cardiology Conference (CinC)*. IEEE, 577–580.

[42] Amar Kachenoura, Fabienne Porée, Guy Carrault, and Alfredo I. Hernández. 2009. Comparison of four estimators of the 3D cardiac electrical activity for surface ECG synthesis from intracardiac recordings. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 485–488.

[43] Amar Kachenoura, Fabienne Porée, A. I. Hernández, and Guy Carrault. 2007. Surface ECG reconstruction from intracardiac EGM: A PCA-vectorcardiogarm method. In *Conference Record of the 41st Asilomar Conference on Signals, Systems and Computers*. IEEE, 761–764.

[44] Amar Kachenoura, Fabienne Porée, Alfredo I. Hernández, and Guy Carrault. 2008. Using intracardiac vectorcardiographic loop for surface ECG synthesis. *EURASIP J. Adv. Sig. Process.* 2008 (2008), 1–8.

[45] Rahul Kher. 2019. Signal processing techniques for removing noise from ECG signals. *J. Biomed. Eng. Res* 3 (2019), 1–9.

[46] J. Kormylo and V. Jain. 1974. Two-pass recursive digital filter with zero phase shift. *IEEE Trans. Acoust. Speech Sig. Process.* 22, 5 (1974), 384–387.

[47] Royson Lee, Łukasz Dudziak, Mohamed Abdelfattah, Stylianos I. Venieris, Hyeji Kim, Hongkai Wen, and Nicholas D. Lane. 2020. Journey towards tiny perceptual super-resolution. *arXiv preprint arXiv:2007.04356*.

[48] Chaojian Li, Tianlong Chen, Haoran You, Zhangyang Wang, and Yingyan Lin. 2020. HALO: Hardware-aware learning to optimize. In *European Conference on Computer Vision (ECCV)*.

[49] Kunyang Li, Weifeng Pan, Yifan Li, Qing Jiang, and Guanzheng Liu. 2018. A method to detect sleep apnea based on deep neural network and hidden Markov model using single-lead ECG signal. *Neurocomputing* 294 (2018), 94–101.

[50] Yuhong Li, Cong Hao, Xiaofan Zhang, Xinheng Liu, Yao Chen, Jinjun Xiong, Wen-mei Hwu, and Deming Chen. 2020. EDD: Efficient differentiable DNN architecture and implementation co-search for embedded AI solutions. *arXiv preprint arXiv:2005.02563*.

[51] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L. Yuille, and Li Fei-Fei. 2019. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 82–92.

[52] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

[53] Chris J. Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Conference on Advances in Neural Information Processing Systems*. 3086–3094.

[54] G. Stuart Mendenhall. 2010. Implantable and surface electrocardiography: Complementary technologies. *J. Electrocardiol.* 6, 43 (2010), 619–623.

[55] G. Stuart Mendenhall and Samir Saba. 2010. 12-lead surface electrocardiogram reconstruction from implanted device electrograms. *Europace* 12, 7 (2010), 991–998.

[56] Kriegh P. Moulton, Tim Medcalf, and Ralph Lazzara. 1990. Premature ventricular complex morphology. A marker for left ventricular structure and function. *Circulation* 81, 4 (1990), 1245–1251.
[57] Eyal Nof, William G. Stevenson, and Roy M. John. 2013. Catheter ablation for ventricular arrhythmias. *Arrhyth. Electrophys. Rev.* 2, 1 (2013), 45.
[58] NVIDIA Inc. 2020. NVIDIA Jetson TX2. Retrieved from https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/.
[59] NVIDIA Inc. 2020. TensorRT. Retrieved 1 september, 2021 https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html.
[60] Angshuman Parashar, Priyanka Raina, Yakun Sophia Shao, Yu-Hsin Chen, Victor A. Ying, Anurag Mukkara, Rangharajan Venkatesan, Brucek Khailany, Stephen W. Keckler, and Joel Emer. 2019. Timeloop: A systematic approach to DNN accelerator evaluation. In *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 304–315.
[61] Shahla Parveen and Phil Green. 2004. Speech enhancement with missing data techniques using recurrent neural networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1. IEEE, I–733.
[62] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. 2018. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*.
[63] Fabienne Porée, Amar Kachenoura, Guy Carrault, Renzo Dal Molin, Philippe Mabo, and Alfredo I. Hernández. 2012. Surface electrocardiogram reconstruction from intracardiac electrograms using a dynamic time delay artificial neural network. *IEEE Trans. Biomed. Eng.* 60, 1 (2012), 106–114.
[64] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. 2019. Regularized evolution for image classifier architecture search. In *AAAI Conference on Artificial Intelligence*, Vol. 33. 4780–4789.
[65] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 234–241.
[66] Kyle Rupnow, Yun Liang, Yinan Li, Dongbo Min, Minh Do, and Deming Chen. 2011. High level synthesis of stereo matching: Productivity, performance, and software constraints. In *International Conference on Field-programmable Technology*. IEEE, 1–8.
[67] Yongming Shen, Michael Ferdman, and Peter Milder. 2017. Maximizing CNN accelerator efficiency through resource partitioning. In *44th Annual International Symposium on Computer Architecture (ISCA'17)*. Association for Computing Machinery, New York, NY, 535–547. DOI: https://doi.org/10.1145/3079856.3080221
[68] Mennatullah Siam, Mostafa Gamal, Moemen Abdel-Razek, Senthil Yogamani, Martin Jagersand, and Hong Zhang. 2018. A comparative study of real-time semantic segmentation for autonomous driving. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 587–597.
[69] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. 2019. Single-path NAS: Designing hardware-efficient convnets in less than 4 hours. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 481–497.
[70] R. Sukanesh, S. Palanivel Rajan, S. Vijayprasath, S. Janardhana Prabhu, and P. Subathra. 2010. GSM-based ECG telealert system. In *International Conference on Innovative Computing Technologies (ICICT)*. IEEE, 1–5.
[71] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. 2019. MnasNet: Platform-aware neural architecture search for mobile. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2820–2828.
[72] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*.
[73] Fleur V. Y. Tjong and Vivek Y. Reddy. 2017. Permanent leadless cardiac pacemaker therapy: A comprehensive review. *Circulation* 135, 15 (2017), 1458–1470.
[74] L. Tran, X. Liu, J. Zhou, and R. Jin. 2017. Missing modalities imputation via cascaded residual autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition*. 1405–1414.
[75] Caroline J. M. van Deursen, Yuri Blaauw, Maryvonne I. Witjens, Luuk Debie, Liliane Wecke, Harry J. G. M. Crijns, Frits W. Prinzen, and Kevin Vernooy. 2014. The value of the 12-lead ECG for evaluation and optimization of cardiac resynchronization therapy in daily clinical practice. *J. Electrocardiol.* 47, 2 (2014), 202–211.
[76] Rangharajan Venkatesan, Yakun Sophia Shao, Miaorong Wang, Jason Clemons, Steve Dai, Matthew Fojtik, Ben Keller, Alicia Klinefelter, Nathaniel Pinckney, Priyanka Raina, Yanqing Zhang, Brian Zimmer, William J. Dally, Joel Emer, Stephen W. Keckler, and Brucek Khailany. 2019. MAGNet: A modular accelerator generator for neural networks. In *International Conference on Computer-Aided Design (ICCAD)*.
[77] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, Peter Vajda, and Joseph E. Gonzalez 2020. FBNetV2: Differentiable neural architecture search for spatial and channel dimensions. *arXiv preprint arXiv:2004.05565*.

[78] Junsong Wang, Qiuwen Lou, Xiaofan Zhang, Chao Zhu, Yonghua Lin, and Deming Chen. 2018. Design flow of accelerating hybrid extremely low bit-width neural network in embedded FPGA. In *28th International Conference on Field-Programmable Logic and Applications (FPL)*.

[79] Y. Wang, J. Xu, Y. Han, H. Li, and X. Li. 2016. DeepBurning: Automatic generation of FPGA-based learning accelerators for the neural network family. In *53rd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6.

[80] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. 2019. FastDepth: Fast monocular depth estimation on embedded systems. In *International Conference on Robotics and Automation (ICRA)*. IEEE, 6101–6108.

[81] Mark A. Wood and Kenneth A. Ellenbogen. 2002. Cardiac pacemakers from the patient's perspective. *Circulation* 105, 18 (2002), 2136–2138.

[82] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. 2019. FBNet: Hardware-aware efficient ConvNet design via differentiable neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition*. 10734–10742.

[83] Junru Wu, Yue Wang, Zhenyu Wu, Zhangyang Wang, Ashok Veeraraghavan, and Yingyan Lin. 2018. Deep k-Means: Re-training and parameter sharing with harder cluster assignments for compressing deep convolutions. *arXiv preprint arXiv:1806.09228*.

[84] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. 2018. SNAS: Stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*.

[85] Xilinx Inc. 2020. Chaidnnv2: HLS-based Deep Neural Network Accelerator Library for Xilinx Ultrascale+ MPSoCs. Retrieved from https://github.com/Xilinx/CHaiDNN.

[86] Xilinx Inc. 2020. Vivado High-Level Synthesis. Retrieved from https://https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html.

[87] Xilinx Inc. 2020. Xilinx Zynq-7000 SoC ZC706 Evaluation Kit. Retrieved from https://www.xilinx.com/products/boards-and-kits/ek-z7-zc706-g.html.

[88] Peng Xiong, Hongrui Wang, Ming Liu, Suiping Zhou, Zengguang Hou, and Xiuling Liu. 2016. ECG signal enhancement based on improved denoising auto-encoder. *Eng. Applic. Artif. Intell.* 52 (2016), 194–202.

[89] Pengfei Xu, Xiaofan Zhang, Cong Hao, Yang Zhao, Yongan Zhang, Yue Wang, Chaojian Li, Zetong Guan, Deming Chen, and Yingyan Lin. 2020. AutoDNNchip: An automated DNN chip predictor and builder for both FPGAs and ASICs. *arXiv preprint arXiv:2001.03535*.

[90] Sean Shensheng Xu, Man-Wai Mak, and Chi-Chung Cheung. 2018. Towards end-to-end ECG classification with raw signal extraction and deep neural networks. *IEEE J. Biomed. Health Inform.* 23, 4 (2018), 1574–1584.

[91] Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee. 2014. A regression approach to speech enhancement based on deep neural networks. *IEEE/ACM Trans. Audio, Speech Lang. Process.* 23, 1 (2014), 7–19.

[92] Lei Yang, Zheyu Yan, Meng Li, Hyoukjun Kwon, Liangzhen Lai, Tushar Krishna, Vikas Chandra, Weiwen Jiang, and Yiyu Shi. 2020. Co-exploration of neural architectures and heterogeneous ASIC accelerator designs targeting multiple tasks. *arXiv preprint arXiv:2002.04116*.

[93] Xuan Yang, Jing Pu, Blaine Burton Rister, Nikhil Bhagdikar, Stephen Richardson, Shahar Kvatinsky, Jonathan Ragan-Kelley, Ardavan Pedram, and Mark Horowitz. 2016. A systematic approach to blocking convolutional neural networks. *CoRR* abs/1606.04209.

[94] Anjali S. Yeole and Dhananjay R. Kalbande. 2016. Use of Internet of Things (IoT) in healthcare: A survey. In *ACM Symposium on Women in Research*. 71–76.

[95] Chen Zhang, Peng Li, Guangyu Sun, Yijin Guan, Bingjun Xiao, and Jason Cong. 2015. Optimizing FPGA-based accelerator design for deep convolutional neural networks. In *International Symposium on Field-programmable Gate Arrays*. ACM, 161–170.

[96] Chen Zhang, Guangyu Sun, Zhenman Fang, Peipei Zhou, Peichen Pan, and Jason Cong. 2018. Caffeine: Towards uniformed representation and acceleration for deep convolutional neural networks. *IEEE Trans. Comput.-Aided Des. Integ. Circ. Syst.* 38, 11 (2018), 2072–2085. DOI: 10.1109/TCAD.2017.2785257

[97] Xiaofan Zhang, Junsong Wang, Chao Zhu, Yonghua Lin, Jinjun Xiong, Wen-Mei Hwu, and Deming Chen. 2018. DNNBuilder: An automated tool for building high-performance DNN hardware accelerators for FPGAs *(ICCAD'18)*. Association for Computing Machinery, New York, NY. DOI: https://doi.org/10.1145/3240765.3240801

[98] Y. Zhao, C. Li, Y. Wang, P. Xu, Y. Zhang, and Y. Lin. 2020. DNN-Chip predictor: An analytical performance predictor for DNN accelerators with various dataflows and hardware architectures. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1593–1597.

[99] Barret Zoph and Quoc V. Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.

[100] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. 2018. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8697–8710.

Received December 2020; revised April 2021; accepted May 2021