Personalized Neural Network for Patient-Specific Health Monitoring in IoT: A Meta-Learning Approach

Zhenge Jia, Yiyu Shi, Jingtong Hu

***Abstract*—The Internet of Things (IoT) has been widely applied in personal health monitoring on biosignals. Conventional detection methods in the field count on a variety of heuristic criteria by utilizing extracted features, which are carefully selected through extensive clinical trials and experts' experiences. Recently, deep learning (DL) gains rapidly growing attention in health monitoring. The most significant advantage of DL-based methods is that DL could execute feature engineering automatically with only labeled data, which results in a great reduction in the expertise involved and manual works in the detection methods design. However, individual differences among various patients (subjects) can lead to accuracy degradation of the pre-trained deep model. Simply fine-tuning the deep model with the patient-specific data cannot alleviate the problem since the pre-trained model may not generalize well to new data. To address the problem, we propose a meta-learning based personalization method to generate the personalized neural network for each patient to conduct patient-specific detection. Specifically, the proposed meta-learning method leverages a novel patient-wise training tasks formatting strategy to train the neural network that ends up with a well-generalized model initialization containing across-patient knowledge. The well-generalized model initialization would then be utilized to perform a quick adaptation to the specific patient's data domain. In this way, a new patient could be immediately assigned with a personalized neural network using limited labeled data. Experimental results show that the proposed meta-learning based personalization method achieves 8.2%, 2.5%, and 6.4% higher accuracy when compared with the existing deep learning detection methods in VF detection, AF detection, and human activity recognition respectively.**

***Index Terms*—Deep Learning, IoT, Personalization, Health Monitoring, Biosignal.**

I. Introduction

Biomedical sensors have been widely utilized to perform continuous and real-time health monitoring by being embedded into the Internet-of-Thing (IoT) devices. These sensors detect a broad range of biomedical signals, including electrocardiogram (ECG) signal, intracardiac electrogram (IEGM) signal, electroencephalography (EEG) signal, Microelectromechanical systems (MEMS) motion signal, etc. The IoT devices embedded with biomedical sensors are increasingly considered as a promising alternative to conventional analytical instruments in the personal healthcare industry due to their accuracy, low cost, and simplicity [1], [2], [3].

The conventional computer-aided methods are deployed in the IoT monitors to conduct various tasks such as arrhythmia detection [4], [5], [6] and human activity recognition [7], [8], [9]. To find the most effective detection methods, essential features extracted from the sensed data and the corresponding detection rules are first derived from clinical trials. The detection criteria are then transformed into a program that is deployable and runnable on the IoT monitors. There are hundreds of programmable parameters such as threshold value affecting the detection in these methods. However, the process of conventional detection methods design requires considerable expertise and labor works. Moreover, to achieve optimal performance over all patients, experts have to carefully select the features and adjust the programmable parameters [10], [11], [12].

Recently, deep learning (DL) based detection on biosignals has achieved outstanding performance in terms of accuracy [13], [14], [15], [16], [17]. When compared with conventional detection methods, the most significant advantage of DL is the reduction of the expertise required in the detection method design. The labeled biosignals could be directly utilized as the training material and the DL model outputs the prediction without cumbersome criteria design, which is done by experts or doctors in conventional methods.

However, there are three main challenges in applying the aforementioned DL-based detection in health monitoring on the resource-constrained IoT devices: 1) The detection performance of the pre-trained deep model would degrade significantly on some patients due to individual differences. There is a high inter-patient variability on the biosignals in terms of morphological characteristics [18], [19], [20]; 2) Fine-tuning the pre-trained deep model using the biosignals of the specific patient can be a straightforward but effective personalization approach. However, the performance of the fine-tuned model is highly dependent on the generalization of the initialization of the pre-trained model. 3) To keep personal data confidential and perform personalized monitoring, it is highly preferable to conduct inference and model personalization on the user end. However, most state-of-the-art deep models cannot satisfy the resource constraints if deployed on the resource-constrained IoT monitor such as microcontrollers (MCU). It is even more impractical to conduct model personalization on the monitor due to its confined computational resources.

Zhenge Jia and Jingtong Hu are with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, 15213 USA (e-mail: zhenge.jia, jthu@pitt.edu).

Yiyu Shi is with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, 46556 USA (e-mail: yshi4@nd.edu).

This project is supported in part by National Science Foundation under grants CNS-2007274 and CNS-1822099

To address the aforementioned challenges, we first propose a meta-learning method to personalize the convolutional neural network (CNN) for patient-specific detection. Differing from the *N-way-K-shot* classification targeted by conventional meta-learning approaches such as MAML [21], our meta-learning method accommodates the patient-specific detection through a novel *patient-wise training tasks formatting strategy*. In this way, the CNN could learn across-patient knowledge and end up with a well-generalized model initialization. For the meta-learning method, there are two essential processes defined as the *inner-loop update* and the *outer-loop update* [21], [22]. The inner-loop update is conducted to acquire patient-specific knowledge, whereas the outer-loop update acquires across-patient knowledge. We propose an inner-loop update steps annealing strategy to reduce the update steps to speed up the meta-learning process and further propose a cyclical outer-loop learning rate mechanism to improve the generalization of the initial model. When a new patient is assigned, a quick adaption would be performed on the well-generalized CNN to the specific patient's biosignals with a limited amount of labeled data of the patient. We further present an edge computing framework, where the model personalization can be conducted on the user end without uploading their personal data to the cloud. By leveraging the STM32 Cube.AI development tool, the personalized model can be deployed on low-power MCU to evaluate the practical performances in terms of latency, memory overhead, and power. The effectiveness of our meta-learning method is examined by three health monitoring applications including Ventricular Arrhythmia (VA) detection on IEGM, Atrial Fibrillation (AF) detection on ECG, and Human Activity Recognition (HAR) on MEMS motion signals.

The main contributions are summarized as follows:

- We propose a patient-wise task formatting strategy that enables a general-purpose meta-learning method to effectively learn across-patient knowledge.
- We introduce two optimizations techniques for the inner- and outer-loop update to speed up the meta-learning process while improving the model generalization.
- Our meta-learning method is shown to be effective and generalized for various detection tasks on biosignals while meeting the requirements of IoT health monitoring applications in terms of memory, response time, and energy consumption.

The rest of the paper is organized as follows. Section II gives the background and related works. Section III introduces the motivation of this work. Section IV presents the proposed meta-learning based personalization method. Section V evaluates the proposed personalization method. Finally, Section VI concludes the paper.

II. Background and Related Works

A. Health Monitoring in IoT on Biosignals

With the rapid development of biomedical sensors, IoT devices are gaining more attention in health monitoring. IoT monitoring device embedded with biomedical sensors presents a way to monitor patients' health conditions in a continuous, real-time, and connective manner. The disease that occurs sporadically and acutely could be detected by the computer-aided method, diagnosed by doctors through data uploading, and even treated by the IoT device in time. IoT-based health monitoring is considered to be a promising alternative to in-hospital instruments in personal healthcare industry.

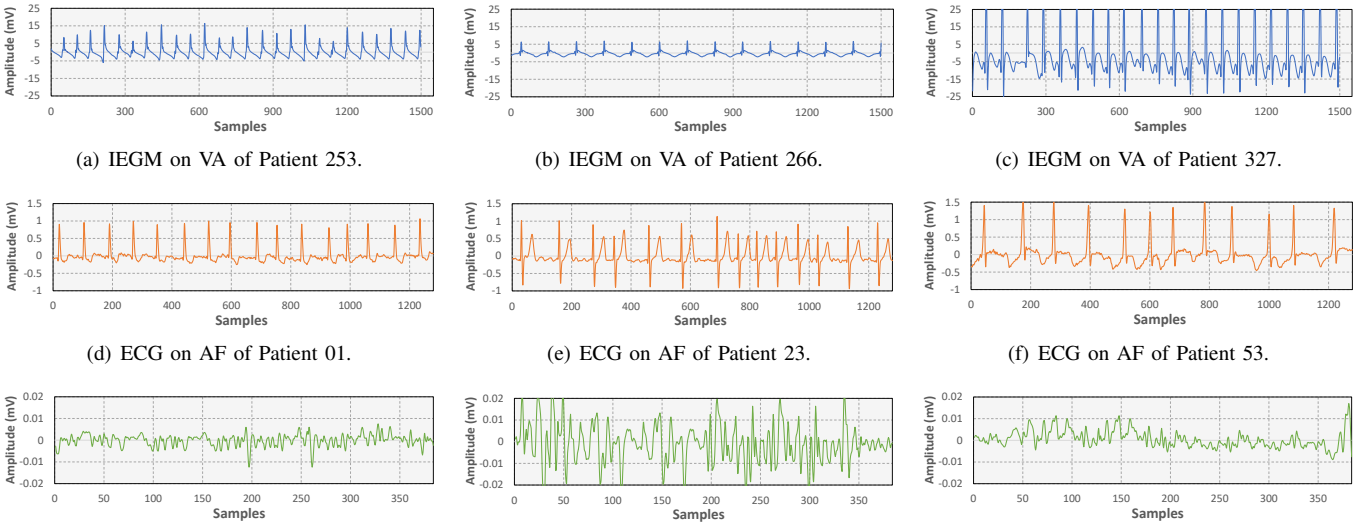
There are various applications in health monitoring using IoT devices. In the field of cardiac monitoring, the biosignals reflected cardiac rhythm have been utilized for arrhythmias detection. Implantable Cardioverter Defibrillator (ICD) is a small device implanted to reduce Sudden Cardiac Death (SCD) risk and improve the survival rate by detecting Ventricular Tachycardia (VT) and Ventricular Fibrillation (VF) on intracardiac electrograms (IEGMs) and delivering defibrillation [4]. The arrhythmia detection methods deployed on ICD count on a wide variety of criteria and there are hundreds of parameters affecting the defibrillation decision [11], [23]. With the capability of IoT, the sensed rhythm and treatment history can be uploaded to help doctors to fine-tune the parameters for each ICD recipient [24]. Long-term rhythm monitors with connectivity are leveraged for Atrial Fibrillation (AF) detection. The devices such as Insertable Cardiac Monitor (ICM) [5], [10], wearable patch cardiac monitors [6], [25], and smart watches [26] are programmed with AF detection methods. Those devices could transmit the self-detected suspicious AF-rhythm ECG episodes to the cardiologists for further diagnosis.

In Human Activity Recognition (HAR), the biosignals of accelerometer and gyroscope have been frequently utilized as the input for activity classification. A real-time and automatic activity recognition algorithm using wireless accelerometers and heart rate monitors is devised in [27]. ActiServ system is proposed to utilize fuzzy inference based classifiers to perform real-time activity recognition on the mobile phone [7]. In [28], authors introduce an Android-based application named ActiWare that could perform real-time recognition with the Naive Bayes classifier on the extracted features from the biosignals obtained with the built-in accelerometer. Authors in [9] present a activity recognition based on hierarchical classification approach on resource-constrained IoT platforms. The devised approach could be executed either on gateway or IoT devices.

B. Deep Learning in Healthcare

Deep learning is a subset of machine learning algorithms and has been shown to outperform conventional methods in various fields such as visual object detection and speech recognition. The most significant advantage of deep learning is that the deep models could automatically learn to extract essential features through training with labeled data. On the other hand, conventional methods normally require an extensive amount of domain-level expertise to first define features and perform classification based on the extracted features.

With the development of the technique, deep learning is being accepted in health monitoring applications on biosignals. Deep learning based arrhythmia detection has achieved cardiologist-level performance in terms of accuracy [14]. In their works, the authors utilize a convolutional neural network (CNN) with ResNet architecture to perform twelve arrhythmias classifications on 12-lead ECG. Authors in [16] develop



(a) IEGM on VA of Patient 253. (b) IEGM on VA of Patient 266. (c) IEGM on VA of Patient 327.

(d) ECG on AF of Patient 01. (e) ECG on AF of Patient 23. (f) ECG on AF of Patient 53.

(g) Acceleration on Standing of Subject 11. (h) Acceleration on Standing of Subject 12. (i) Acceleration on Standing of Subject 16.

Fig. 1. Signal segments from different patients over three types of biosignals.

a 37-layer deep neural network to classify 4 triage categories derived from clinical experiences on 12-lead ECGs. Authors in [13] propose a CNN with a simple structure for ventricular arrhythmia detection using single-lead ECG. Authors in [29] propose a simple but effective 1D CNN for atrial fibrillation detection on ECG. In [30], two 2D CNNs are presented to detect AF using the ECG signals processed with Short-Term Fourier Transform (STFT) and Stationary Wavelet Transform (SWT). In [15], a two-channel deep neural network is proposed to detect the presence of AF on ECG. The first channel learns where to attend for detection and the second channel learns to extract the features from ECG.

Deep learning has also been applied in HAR. Authors in [31] propose a CNN model to detect human activity based on the tri-axial acceleration signals. Authors in [17] introduce a 1D CNN-based method to recognize 3 human activities (i.e., walking, running, and staying still) based on the acceleration signals obtained from the built-in accelerometer of the smartphone. Authors in [32] utilize CNNs with varying kernel dimensions along with Bi-directional LSTM (BiLSTM) to automatically capture spatial and temporal features on the biosignals obtained from accelerometer and gyroscope. Authors in [33] propose a simple CNN to accurately classify the action while being able to conduct inference in real-time.

Apart from arrhythmia detection and HAR, authors in [34] use Long Short-Term Memory (LSTM)-based siamese networks to detect Parkinson's disease on the speech signal. Authors in [35] utilize the data obtained from accelerometer and touchscreen typing to construct multi-modal deep learning to detect Parkinson's disease. Moreover, DL has achieved outstanding performance in sleep stage classification. A DL approach is proposed to conduct an end-to-end sleep stage classification based on multivariate signals (EEG, EOG, and EMG) without calculating spectrograms or extracting features [36]. Authors of the work [37] proposed a multimodal salient wave detection network that accurately detects the sleep stage and adaptively targets essential information from multimodal input.

III. Motivations

We have performed some investigations and preliminary experiments to evaluate the feasibility of deep learning to health monitoring on biosignals in IoT. From our experimental results, we find that there are several challenges in applying deep learning for personalized and real-time health monitoring on the resource-constrained IoT monitor.

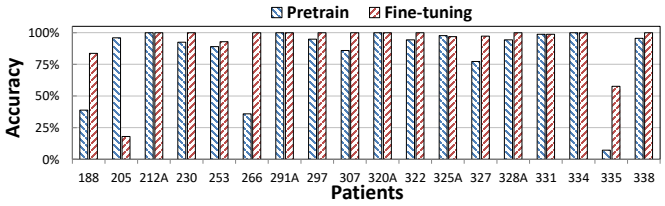
A. Individual Differences on Deep Model Performance

We first evaluate the impact of individual differences on the detection performance of the pre-trained deep models. Individual differences could cause inter-patient variability on biosignals in terms of morphological characteristics.

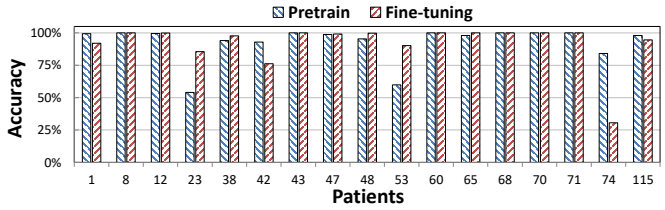
Here, we utilize the biosignal datasets of IEGMs [38], ECG [39] and MEMS motion sensors [40] for Ventricular Arrhythmias (VA) detection, Atrial Fibrillation (AF) detection, and Human Activities Recognition (HAR) respectively as our case study applications. For illustration, Fig. 1 shows the biosignals of IEGMs, ECG, and acceleration over three patients (subjects) on the same event (i.e., VA in Fig. 1(a-c), AF in Fig. 1(d-f), and Action Standing in Fig. 1(g-i) respectively). As shown in Fig. 1, the morphological characteristics of biosignals are highly variable even on the event (arrhythmia) of the same type over different patients (subjects).

We then choose three CNNs (denoted as *CNN-VA* [41], *CNN-AF* [29], and *CNN-HAR* [33]) that are proposed for health monitoring on biosignals for each application. The detailed dataset descriptions, experimental setup, and results are illustrated in Section V.

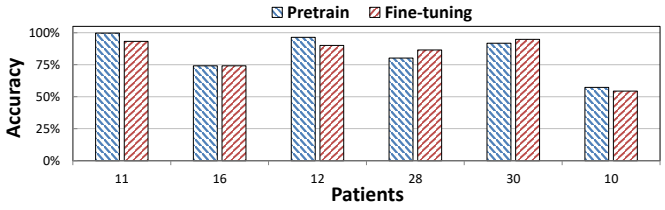
Fig. 2 shows the experimental results in terms of accuracy of each testing patient for all three applications. The individuals are chosen from the testing fold and are denoted with their corresponding numbering on the X-axis. From the figure, we observe that a pre-trained CNN cannot achieve the expected high detection accuracy for every patient in all cases. For



(a) Individual accuracy in VA detection.



(b) Individual accuracy in AF detection.



(c) Individual accuracy in HAR.

Fig. 2. Individual detection accuracy on pre-trained and fine-tuned deep models over three case study applications.

example, in terms of AF detection, there is a large degradation of accuracy for some patients (e.g., Patient 23 and 53) while the pre-trained model could achieve a near-perfect detection accuracy for the others. It is because that individual differences cause inter-patient variability, which appears to have unique morphological characteristics and dynamics in biosignals (as shown in Fig. 1) [18], [19], [42]. Therefore, personalizing the pre-trained deep model is necessary for each patient.

B. Fine-tuning for Personalized Deep Model

As we stated in the previous subsection, personalization is a necessary procedure in biosignal health monitoring to achieve a better patient-specific detection performance. In conventional methods, domain-level expertise is highly demanded since the detection criteria, extracted features and programmable parameters are carefully determined and modified for each individual [4], [5], [43]. Comparing with conventional methods, personalization in deep learning can be achieved through fine-tuning, which requires much less domain knowledge. Fine-tuning the pre-trained deep model using a limited amount of specific patient's data to obtain the personalized model has been investigated in various health monitoring applications such as arrhythmias detection [18], epileptic seizure detection [44], [45], human activity recognition [46], etc.

We have conducted experiments to further evaluate the performance of the fine-tuning based personalization on the same testing patients from the three case study applications. For each dataset, the fine-tuning is conducted on the pre-trained CNN using each testing patient's limited data with a uniform hyperparameter setting. Fig. 2 shows the accuracy of each testing patient after fine-tuning. As shown in the figure, some deep models personalized with the patient-specific data even experience a significant accuracy degradation (e.g., patient 205 in VA detection and patient 74 in AF detection). The performances indicate that the pre-trained CNN could easily overfit the patient-specific data during fine-tuning. A straightforward solution is to optimize the hyperparameters (e.g., update steps, learning rate, etc) of the fine-tuning process for each patient. However, because of the inherent individual differences of biosignals, each patient is expected to have varying convergence rates during model fine-tuning.

In other words, it is hard to determine when the pre-trained model starts overfitting and it is impractical to determine the proper hyperparameters with limited patient-specific data. Therefore, improving the generalization of the pre-trained model's initialization is an alternative but necessary approach to the problem.

Meta-learning approaches (e.g. MAML [47] and Reptile [48]) provide a learning strategy that generates a pre-trained model with well-generalized initialization by training on the tasks (i.e., *N-way-K-shot* classification) containing support and query set. The pre-trained model with well-generalized initialization is shown to be able to quickly adapt to the new task by fine-tuning. Such meta-learning approaches are compatible with our application scenarios where the model personalization is necessary and there is a limited amount of patient-specific data. However, directly applying those methods would mix training samples with different classes from various patients in a single training task during the meta-learning process. The training tasks with such biased data distribution could cause gradient diminishing and training instability, which degrade the generalization of the model and lead to low detection accuracy after fine-tuning. Therefore, it is necessary to further optimize the meta-learning process to adapt the patient-specific detection scenarios.

C. Feasibility of Deep Model Deployment and Personalization

We further evaluate the feasibility of the deployment of deep model on resource-constrained IoT monitors. The resource-constrained IoT monitors are considered to be wearable or implantable devices using microcontrollers (MCUs) since only the MCU-based IoT monitors could perform long-term, real-time and continuous detection with limited-capacity batteries in the health monitoring scenarios.

Most MCU-based IoT monitors have less than 256 KB on-chip memory, 2 MB off-chip memory, and 200 MHz CPU frequency [49]. As a result, some deep models with a great number of weight parameters for accurate detection [14], [15] cannot be deployed or executed on the MCU-based IoT monitors. Even if optimizations such as pruning and compression are applied to the models, the work-in memory required during inference could easily exceed the capacity of the on-chip memory of the low-power MCUs and the execution latency of a large model may not satisfy the time constraints.

An alternative solution to the constrained hardware resources problem is to perform hierarchical detection, where the MCU-based IoT monitor carries lightweight classification

and offloads compute-intensive classification tasks to a local gateway or cloud server [9], [50]. However, the approach still requires a relatively frequent data exchange between the IoT monitor and the offloaded device. Moreover, the cybersecurity vulnerabilities in data transmission to the cloud could lead to sensitive data leakage and device manipulation [51], [52]. It is highly demanded to perform inference and model updates on the user end without uploading personal health data. Therefore, it is necessary to provide a system-level solution for privacy-preserving model personalization and efficient inference on the recourse-constrained IoT monitor.

IV. Method

In this section, we present the details of the personalized deep learning based patient-specific health monitoring for the resource-constrained IoT monitors. There are two key steps: 1) Meta-learning: Based on the problem formulation, the proposed meta-learning is conducted on the server with collected patients' data to obtain a well-generalized deep model initialization containing across-patient knowledge; 2) Personalization: A quick personalization is performed on the meta-model to adapt to the specific patient's data.

Compared with the original method introduced in [41], the proposed meta-learning method further optimizes the patient-wise tasks formatting strategy to accommodate different health monitoring applications other than ventricular arrhythmias detection. The strategy enables the method to conduct the meta-learning process even when the patients are not with a certain class of data. Furthermore, optimization techniques such as cyclical learning rate mechanism are applied in the proposed method to improve the initial model generalization. We further introduce the updates for the model personalization in [41]. The model personalization would be executed on the user end without data uploading by leveraging an edge computing framework.

A. Meta-Learning

Existing meta-learning algorithms [47], [48] focus on solving *N-way-K-shot* classification and are not compatible to the patient-specific detection scenarios. To address the challenges, we propose a patient-wise training tasks formatting strategy and two other optimization techniques to ensure that the meta-learning process is stabilized and the across-patient knowledge is well-learned by the meta-model.

We first introduce the meta-learning process along with necessary definitions. The initial model parameters of the deep learning model is denoted as ϕ. There is a *TaskSet* defined as $\mathcal{T}$, which contains tasks τ extracted from training dataset. The meta-learning process would iteratively train the deep model ϕ over tasks extracted from the $\mathcal{T}$ to obtain the well-generalized meta-model parameters ϕ^*.

In the meta-learning process, the patient-wise training tasks formatting strategy is proposed to formulate the tasks such that the across-patient knowledge could be properly learned by the model. For each task of $\mathcal{T}$, following the strategy, we randomly select $2N$ patients from the training patients dataset for each class. Each one of those $2N$ patients must contain the data labeled with the targeted class. Next, we randomly collect p samples labeled with the targeted class of each patient in the first N selected patients and q samples of each patient in the rest N patients Such process would be repeated for c times to form a task τ_i, where c is the total number of classes. The total number of tasks in $\mathcal{T}$ is then denoted as TS.

Once the task τ_i is formed, the collected p samples of each class from the first N patients would be extracted to form the *support set*, denoted as τ_i^{spt}. The q samples of each class from the other N patients in τ_i would be formed as *query set*, denoted as τ_i^{qry}. The formal definition of both set are shown as follows:

$$\begin{aligned}\tau_i^{spt} &= \{(x,y)_{i,M_j^{spt}}\} \text{ for } j = 1, ..., c \cdot p \cdot N, \\ \tau_i^{qry} &= \{(x,y)_{i,M_j^{qry}}\} \text{ for } j = 1, ..., c \cdot q \cdot N,\end{aligned} \tag{1}$$

where $(x,y)_i$ is the data-label pairs in τ_i, and M^{spt} and M^{qry} are the pair indices (in τ_i) set for τ_i^{spt} and τ_i^{qry} separately. There are total $c \cdot p \cdot N$ indices for pairs in τ_i^{spt} and $c \cdot q \cdot N$ indices for τ_i^{qry}. A *TaskSet* $\mathcal{T}$ is constructed by repeating the process to extract tasks τ_i for a pre-defined number TS times.

In meta-learning, on each formulated task, there is an important procedure defined as *inner-loop update* [47]. The inner-loop update is the process of fine-tuning the initial model parameters ϕ on the given new task to acquire task-specific knowledge. Here, the deep model inference is denoted as $f_\theta(x)$, where x is the input data and θ is deep model parameters. When adapting the model to the task τ_i in inner-loop update, k-step gradient update is applied to update the model parameter from θ_i^0 $(= \phi)$ to θ_i^k using the segment and label pairs in the support set τ_i^{spt}. The gradient update on step m (where $0 < m \leq k$) is defined as follows:

$$\theta_i^m = \theta_i^{m-1} - \alpha \frac{1}{|\tau_i^{spt}|} \sum_{(x,y)\in\tau_i^{spt}} \nabla_{\theta_i^{m-1}} \mathcal{L}(f_{\theta_i^{m-1}}(x), y), \tag{2}$$

where $\mathcal{L}$ is the loss function and α is the inner-loop learning rate. The gradient descent (GD) in Eqn. (2) would be processed for k steps to end up with the task-specific model with parameters θ_i^k for the task τ_i:

$$\theta_i^k = \text{GD}_k(\theta_i^0), \tag{3}$$

which is inner-loop model update in meta-learning.

Once obtaining a series of task-specific models from the inner-loop update for each task $\tau_i \in \mathcal{T}$, we then evaluate the generalization of the task-specific models and obtain a well-generalized meta-model. This procedure is defined as *outer-loop update*, which acquires across-task knowledge by meta-learning the parameters of each task-specific model obtained from inner-loop update [47].

The outer-loop update begins with calculating the loss of each task-specific model on the corresponding task's query set. The loss is calculated as follows:

$$L_{\tau_i}(\theta_i^k) = \frac{1}{|\tau_i^{qry}|} \sum_{(x,y)\in\tau_i^{qry}} \mathcal{L}(f_{\theta_i^k}(x), y). \tag{4}$$

Next, we form a task batch using the *mini-batch* methodology (i.e., the batch-size number of tasks are grouped as a batch)

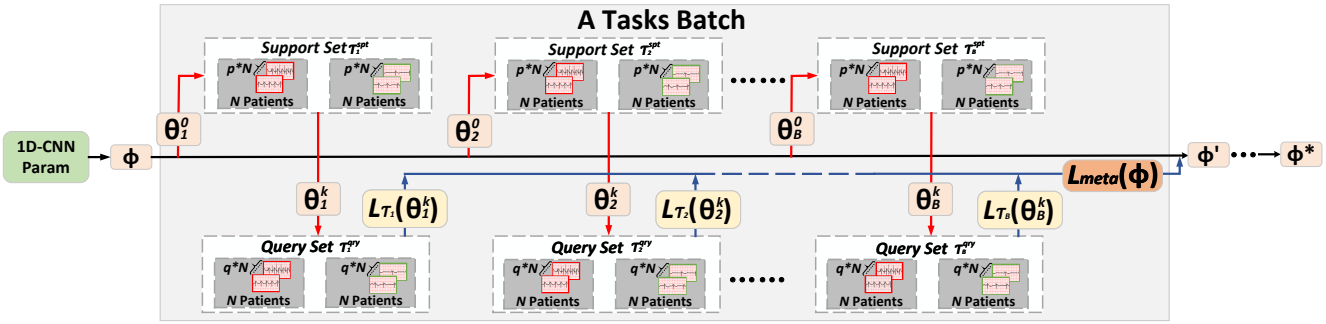


Fig. 3. Illustration of the meta-learning process. The red line indicates the *inner-loop* model update and the blue line indicates the *outer-loop* meta-model update. The meta-model learns the across-patient knowledge and iteratively updates its parameters towards the well-generalized initialization ϕ^*.

from $\mathcal{T}$. The objective is to minimize the loss based on one batch of tasks. The loss is defined as follows:

$$\mathcal{L}_{meta}(\phi) = \sum_{b=1}^{B} L_{\tau_b}(\theta_b^k), \tag{5}$$

where b is the index of the task in the batch and B is the batch size. The minimization of the loss defined in Eqn. (5) is to improve the generalization of meta-model parameters ϕ for all tasks in the batch. Such objective is known as outer-loop meta-model update [47]. The optimizer utilized here could be Adam or Stochastic Gradient Descent (SGD). The meta-model parameters updating is shown as follows:

$$\phi' = \phi - \beta \nabla_{\phi} \frac{1}{|B|} \sum_{b=1}^{B} L_{\tau_b}(\theta_b^k), \tag{6}$$

where β is the outer-loop learning rate and ϕ' is the updated meta-model parameters that contain the across-patient knowledge. The parameters ϕ' would be utilized as the initial parameters for inner-loop and outer-loop update on the next-round batch of tasks. To obtain the well-generalized meta-model parameters ϕ^*, the meta-model would be updated iteratively with the aforementioned inner-loop and outer-loop update on batches of tasks.

Fig. 3 demonstrates the meta-learning process beginning with initial model parameters ϕ and ending up with the well-generalized ϕ^*. The number of classes c for detection is set 2 in the example. Each batch is formed by extracting B number of tasks from $\mathcal{T}$ and the batch size B is fixed. As shown in Fig. 3, for each task τ in a batch, the samples labeled as the first class are denoted with a red border and the samples labeled as the second class are denoted with a green border. The first N patients' p samples of each class form the support set τ^{spt} and the q segments from the other N patients of each class form the query set τ^{qry} of τ. As shown in Fig. 3, the inner-loop update is conducted within each task (indicated by the red line) and the outer-loop update is conducted batch by batch (indicated by blue line). Note that the batches would be re-formulated for the next epoch.

To increase the generalization of the final meta-model parameters and speed up the learning process, we further propose two optimization techniques:

1) Inner-loop Update Optimization: The meta-learning process of conventional meta-learning methods is time-consuming and may come with training instability problem [22]. To speed up the meta-learning process while obtaining the well-generalized meta-model parameters, we propose an inner-loop update steps annealing strategy.

This strategy is based on the observation that the meta-model would quickly converge in the preceding epochs of the meta-learning process. As a consequence, for the task batches utilized in the latter epochs, the losses calculated in the inner-loop update tend to be stable. Therefore, we propose an annealing strategy based on the observation. That is, inner-loop update steps k in Eqn. (3) would be reduced along with epoch numbers. The inner-loop update steps are calculated as follows:

$$\theta_i^k = \mathrm{GD}_k(\theta_i^0), \text{ for } k = \max[k - (e \bmod W), K], \tag{7}$$

where e is the number of current epoch, W is the preset updating stride, and K is the minimal inner-loop update steps.

To stabilize the gradients, we further invoke the multi-step loss optimization to take the loss of the task-specific model calculated from *each* inner-loop update step [22]. The Eqn. (4) can be re-written as:

$$L_{\tau_i}(\theta_i^k) = \sum_{m=0}^{k} v_k(m) \left(\frac{1}{|\tau_i^{qry}|} \sum_{(x,y)\in\tau_i^{qry}} \mathcal{L}(f_{\theta_i^m}(x), y) \right), \tag{8}$$

where k is the inner-loop steps calculated in Eqn. (7) and $v_k(m)$ is the weight of the loss at the step m. A slight difference between our formulation and the method in [22] is that there are k different weights vector v for each calculated inner-loop update step. The loss calculated in step 0 (i.e., the initial parameters θ_0 before the inner-loop update) is also considered in our multi-step loss optimization.

2) Cyclical Outer-loop Learning Rate Mechanism: In MAML, authors utilize a static learning rate for outer-loop learning rate for the meta-model update. Authors in MAML++ [22] further optimize the outer-loop update by setting a cosine annealing of outer-loop learning rate. However, the outer-loop update with a cosine annealing learning rate may suffer from the early overfitting, where the initial model quickly overfits on a small cluster of strongly-featured training data with a high starting learning rate. What makes the problem more severe is that the model is likely not to be back on the "right" track in the following training steps with an annealing learning rate. Another possible problem is that

Algorithm 1: Patient-Specific Meta-Learning

Given ϕ: deep model initial parameters.
Given D, M: samples set and sample-label indices pair set over all training patients.
Given α, β: inner-loop and outer-loop learning rate.
Given p, q: number of samples for support and query set of each class over each patient.
Given N: total number of chosen patients for a task.
Given c, B, TS: total number of classes, task batch size, TaskSet size.
Given W, k, K: preset updating stride, inner-loop update steps, and the minimal inner-loop update steps.
Given v: weights vectors for inner-loop update.

1 **Initialize** TaskSet $\mathcal{T}$: **for** $i = 1, 2, ..., TS$ **do**
2 Randomly select $2N$ patients containing the data labeled with the target class, format patient-wise task with:
3 $\tau_i^{spt} = \{(x, y)_{i, M_j^{spt}}\}$ for $j = 1, ..., c \cdot p \cdot N$
4 $\tau_i^{qry} = \{(x, y)_{i, M_j^{qry}}\}$ for $j = 1, ..., c \cdot q \cdot N$
5 **end**
6 **for** *each epoch e = 1, 2, ...* **do**
7 Formulate batches of tasks from $\mathcal{T}$
8 $k \leftarrow \max[k - (e \bmod W), K]$
9 $\beta \leftarrow$ cyclic_LR_scheduler(e)
10 **for** *each batch* **do**
11 **for** *b = 1, 2, ..., B* **do**
12 $\theta_b^0 \leftarrow \phi$
13 $\mathcal{L}_b(\theta_b^0) \leftarrow \frac{1}{|\tau_b^{qry}|} \sum_{(x,y) \in \tau_b^{qry}} \mathcal{L}(f_{\theta_b^0}(x), y)$
14 **for** $n = 1, ..., k$ **do**
15 $\theta_b^n \leftarrow \mathrm{GD}_n(\theta_b^0)$
16 $\mathcal{L}_b(\theta_b^n) \leftarrow \frac{1}{|\tau_b^{qry}|} \sum_{(x,y) \in \tau_b^{qry}} \mathcal{L}(f_{\theta_b^n}(x), y)$
17 **end**
18 $\mathcal{L}_b^{qry}(\theta_b^k) \leftarrow \sum_{m=0}^{k} v_k(m) \mathcal{L}_b(\theta_b^m)$
19 **end**
20 $\mathcal{L}_{meta}(\phi) \leftarrow \sum_{b=1}^{B} \mathcal{L}_b^{qry}(\theta_b^k)$
21 $\phi \leftarrow \phi - \beta \nabla_\phi \frac{1}{|B|} \mathcal{L}_{meta}(\phi)$
22 **end**
23 **end**

the meta-model may be hard to jump out of the local minimum with an annealing learning rate during meta-learning.

In this work, we develop a cyclical outer-loop learning rate mechanism by applying cyclical learning rate [53] for meta-model update. To be more specific, the outer-loop learning rate would change cyclically along with the epoch number increase during meta-learning. The primary benefit is that the meta-model would be updated with a relatively lower learning rate in the first few epochs as a "warm-up" starting, which is a way to reduce the chance of causing overfitting in the early epochs. The learning rate changing cyclically could also force the meta-model to jump out of the local minimal for a better-generalized initialization.

Algorithm 1 is devised to comprehensively illustrate the process of our meta-learning approach. The training TaskSet

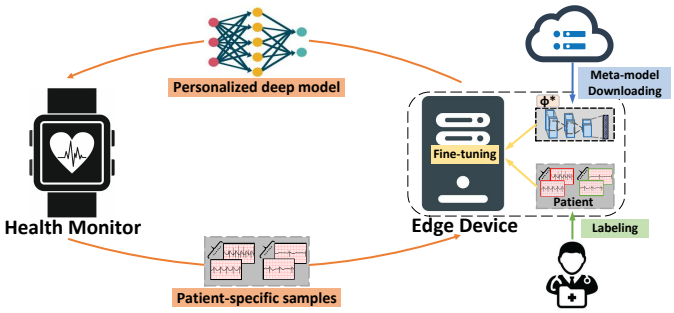


Fig. 4. Illustration of the deep model personalization process.

$\mathcal{T}$ is firstly constructed with the tasks extracted using the proposed patient-wise training task strategy (Line 1-5). Within each epoch, the task batches are formulated with the given batch size B. The inner-loop update step is determined with the mechanism defined in Eqn. 7 of the inner-loop update optimization (Line 8). The outer-loop learning rate is then determined by the cyclical learning rate scheduler (Line 9). Next, for each task in a task batch, the meta-model parameters ϕ is assigned as the task-specific model parameters θ_0 (Line 12). The multi-step loss is calculated based on the task-specific model's inference on the query set from step 0 to k during inner-loop update (Line 13-17). The task-specific model is updated on the task's support set using gradient descent defined in Eqn. 3. The multi-step loss is accumulated with weight vector v_k for each step from 0 to k (Line 18). The accumulated multi-step loss from each batch is then added as the meta-loss and the meta-model is updated with the outer-loop learning rate β (Line 20-21). The aforementioned processes of meta-learning would be executed iteratively.

B. Personalization

Once the well-generalized meta-model ϕ^* is obtained, the next essential step is to personalize the model to adapt to the specific patient's data. The personalized deep model is expected to be deployed on the resource-constrained IoT monitor and perform patient-specific detection on the sensed biosignals. To overcome the resource constraints problem, in this work, we focus on the deep model with a reasonable amount of weight parameters that can perform inference solely on the resource-constrained IoT monitor. We further introduce an edge device to perform the model personalization on the user end without data sharing.

As shown in Fig. 4, the IoT health monitor would firstly transmit some recorded patient-specific samples to the edge device. Once the limited patient-specific samples are received by the edge device, all the doctor is supposed to do is to label those samples through a physical follow-up. The labeling would be processed purely on the edge device without data sharing to keep the sensitive health data confidential.

Next, the edge device would automatically personalize the meta-model ϕ^* downloaded from the cloud by fine-tuning with the labeled patient-specific samples. This process can be conducted with gradient descent based model updating. The meta-model could quickly adapt to the specific patient's rhythm feature since the model with ϕ^* contains across-patient

knowledge through meta-learning. Specifically, the parameters of the deep model with ϕ^* would be updated by SGD and back-propagation with those limited but labeled samples of the new patient. The model would be personalized to adapt to the specific patient with a few training iterations. As shown in Fig. 4, the personalized deep model would be further deployed back on the health monitoring to perform real-time, continuous, and patient-specific detection.

The reason why we devise the computing framework is that the fine-tuning process requires relatively extensive computing resources and cannot be executed solely by the IoT monitor itself. Moreover, it is privacy-preserving to conduct the personalization on the edge device without sending personal data to the cloud. In this case, we set the architecture shown in Fig. 4, which enables the model personalization to be conducted efficiently by the edge device while keeping the personal data on the user end.

It is also worth noting that our proposed method enables the model personalization to be conducted semi-automatically. Doctors in our method are only expected to provide labels on a limited amount of data without receiving qualification training on how to fine-tune programmable parameters of conventional computer-aided methods [23]. The process of parameters fine-tuning in conventional methods requires the doctors to not only read and label the recorded rhythm, but also modify the parameter value based on their expertise and experience. Our method could significantly reduce the workload by eliminating the manual personalized programmable parameters finding process. Once the labeled patient-specific data is received, the deep model could be automatically and effectively adapted to patient-specific data by the proposed meta-learning method.

V. Experiments

In this section, we first introduce data preparation. We then introduce the experimental setup including the evaluation paradigm, evaluated methods, and implementation details. We finally present experimental results in terms of detection metrics and practical performances.

A. Data Preparation

There are three health monitoring applications chosen in the experiments to evaluate the generalization and effectiveness of our proposed method. The applications include Ventricular Arrhythmia (VA) detection on intracardiac electrograms (IEGMs) signal, Atrial Fibrillation (AF) detection on electrocardiogram (ECG) signal, and Human Activities Recognition (HAR) on MEMS motion signal. We would introduce the dataset and data pre-processing for each application.

TABLE I
Data profile of VA dataset.

Data	2-second Segments		Events
	Non-Overlapping	Overlapping	
VA	2,318	10,613	155
Non-VA	6,513	13,047	266

1) VA detection: The dataset of IEGMs is retrieved from volume I & II of Ann Arbor Electrogram Libraries (AAEL), one of the largest dataset for IEGMs and used by all manufacturers developing implantable defibrillators to evaluate their methods [38]. The sampling rate of all recordings is $1,000$ Hz. Different episodes of recordings have been annotated and reviewed by cardiac electrophysiologists to ensure an accurate interpretation of arrhythmia.

Here, we select all recordings over 95 patients to form the dataset. Each selected recording contains the one-channel IEGMs sensed by RVA-Bi lead. First, we apply a band-pass FIR filter with a pass-band frequency of 0.5 Hz and a stop-band frequency of 50 Hz. All recordings are then resampled to 250 Hz as the sampling rate is widely utilized in implantable devices [54]. Then, the recording is divided into various VA or non-VA *events* based on the rhythm diagnostic annotations on the IEGMs. Finally, each event is segmented into *segments* using a 2-second sliding window (250 Hz $\times$ 2 s = 500 samples) with and without a overlap (0.2 s for VA events and 0.5 s for non-VA events). The overlap is set to perform data augmentation for training only. The detailed segments and events statistics are illustrated in Table II.

TABLE II
Data profile of AF dataset.

Data	10-second Segments	Events
AF	358,474	7,358
Non-AF	299,840	46,347

2) AF detection: The dataset of ECG is retrieved from Long Term AF Database (LTAFDB) [39], [55], which includes 84 long-term ECG recordings of human subjects with AF. The sampling rate of each recording is 128 Hz.

Here, we select the recordings of all 84 patients to form the dataset. Each selected recording contains the ECG of the lead I. The first step is to apply a band-pass FIR filter with a pass-band frequency of 0.5 Hz and a stop-band frequency of 50 Hz. Then, the episodes annotated with AF of each recording would be labeled as AF while the other episodes are labeled as non-AF. Each episode is then segmented into non-overlapped 10-second segments and the label (i.e., AF or non-AF) on the segments is the same as the corresponding episode. The detailed data profile of the AF dataset are shown in Table III.

TABLE III
Data profile of HAR dataset.

Data	**Walking**	**Upstairs**	**Downstairs**	**Sitting**	**Standing**	**Laying**
Segments	1,722	1,544	1,406	1,777	1,906	1,944
Events	30	30	30	30	30	30

3) HAR: The dataset of MEMS human motion signals is retrieved from UCI-HAR [40]. UCI-HAR is a commonly used dataset in HAR. It includes the smartphone accelerometer and gyroscope data at a sampling rate of 50Hz over 30 volunteers. There are six activities (i.e., Walking, Upstairs, Downstairs, Sitting, Standing, Laying) wearing a smartphone (i.e., Samsung Galaxy S II) on the waist. The labels of the signal data are made manually based on the recorded video [40]. Here, we select all segments that have been pre-processed by UCI-HAR (i.e., apply noise filters and then sampled in fixed-width sliding

windows of 2.56 seconds and 50% overlap [40]). Each sample consists of 9-channel signals including triaxial acceleration from the accelerometer, triaxial body acceleration, and triaxial angular velocity from the gyroscope. The detailed data profile of the HAR dataset are shown in Table III.

B. Experimental Setup

1) Inter-Patient Evaluation Paradigms: In the experiments, each dataset is firstly partitioned patient-wisely to ensure that the same patients' data can only be in either training or testing set. For example, once the patient is selected for training, the data of the patient would only be utilized in the training set. The split is conducted randomly on the patients and we perform 10-time random sampling on patients for each dataset. The partition ratio is 80%-20% for training and testing patients for each split. The performance is reported based on the average performance of each testing patient.

For the validation, the training patients' signal segments serve as the *training set* and are used to perform learning to learn a well-generalized deep model initialization in our method or training in the baseline methods. For each patient in the testing split, a small portion of segments is extracted to serve as the *personalizing set* and used to personalize or fine-tune the model for the specific patient. There are 12-second (6 segments), 50-second (5 segments), and 10.42-second (4 segments) recordings of each class selected as personalization data in VA detection, AF detection, and HAR respectively. The rest segments of the testing patient serve as *testing set* and are used to report the detection performance. During testing, for each patient, the patient-specific segments from the personalizing set are utilized to personalize the deep model, and the segments from the testing set are utilized to evaluate the personalized model.

In the personalization process of conventional computer-aided methods and deep learning based methods, the personalization set is highly expected to contain all targeted classes. Only in this way, the characteristics of the data could be learned either by doctors or deep learning models to further personalize the method. The detection performance could be degraded if the method is modified without knowing the data with targeted classes. However, it is not always capable to collect data with all targeted classes in a personalization setting. In our experiments, for each testing patient, the predefined number of segments are only collected when there are more than doubled amount of segments of each class.

2) Evaluated Methods: We implement the following detection methods for performance comparison:

Criteria-based detection. We implement conventional detection methods based on detection criteria and handcrafted features for each case study application:

- For VA detection on IEGMs, we simulate the VA detection method used in single-chamber ICDs [24], denoted as *Classic-VA*. This method continuously monitors each heartbeat and reports VA if the criteria are satisfied. We set two detection zones for VT and VF respectively. The heart rate boundary of the VT/VF zone and fast/slow interval threshold are carefully selected for each testing patient to simulate the manual intervention such that the best discrimination performance could be achieved.
- For AF detection on ECG, we simulate an AF detection method deployed in ICM [10], denoted as *Classic-AF*. This method detects AF rhythm based on AF evidence score and P-wave evidence score [10]. The first one is derived from the Lorenz plot. The second one is derived from the features extracted on the P-waves portions of the ECG. The programmable parameters of detection criteria are carefully adjusted for each testing patient.
- For HAR on MEMS motion signals, we implement the detection method using support vector machine (SVM) [40], denoted as *Classic-HAR*. The features used in SVM are 561 hand-designed features extracted in [40].

CNN-based detection. We implement existing deep learning based detection methods for each case study application:

- For VA detection on IEGMs, we invoke two existing CNN models. One is proposed in [41], defined as *CNN1-VA*. The other is proposed in [13], defined as *CNN2-VA*.
- For AF detection on ECG, we implement the method in [29], defined as *CNN-AF*.
- For HAR on MEMS motion signals, we implement the method proposed in [33], defined as *CNN-HAR*.

We invoke the same network structure with necessary modifications (e.g., change filter size and reduce number of conv layers) to fit the input dimensions and recourse-constrained embedded devices for each case. Moreover, the pre-trained CNN would be fine-tuned with the personalizing set and then evaluated on the testing set. We denote them as *CNN1-FT-VA*, *CNN2-FT-VA*, *CNN-FT-AF* and *CNN-FT-HAR*.

Conventional meta learning-based detection. We implement three conventional meta-learning methods to evaluate the effectiveness of the proposed method.

- MAML: a conventional meta-learning method [47], denoted as *MAML-VA*, *MAML-AF* and *MAML-HAR*.
- FOMAML: MAML with first-order approximation in [56], denoted as *FOMAML-VA*, *FOMAML-AF* and *FOMAML-HAR* for each application.
- Reptile: a conventional meta-learning method proposed in [48], denoted as *Reptile-VA*, *Reptile-AF* and *Reptile-HAR* for each application.

Proposed meta learning-based detection. We implement the proposed meta-learning method and denote it as *Meta-VA*, *Meta-AF*, *Meta-HAR* for each application. We also implement the original meta-learning based model proposed in [41] for comparison purpose, denoted as *Meta-Origin-VA*, *Meta-Origin-AF*, and *Meta-Origin-HAR*.

Note that all meta-learning based detection methods would first obtain the meta-model from the training set, fine-tune the meta-model with the personalizing set, and finally evaluate the personalized model on the testing set of each testing patient. The architecture of the meta-model is the one proposed in [41] for VA detection, [29] for AF detection, and [33] for HAR.

3) Implementation Details: We adopt PyTorch (1.6.0) for deep models training and personalization. We have set the random seed using *manual_seed* and *manual_seed_all* for CNN models. All random numbers used in those methods are based

TABLE IV
PERFORMANCES OF METHODS ON SEGMENTS IN VA DETECTION.

Methods	F1	Se/Sp	BAC/ACC	PPV/NPV
CNN1-VA [41]	.893	.966/.885	.925/.885	.884/.971
CNN1-FT-VA	.952	.978/.958	.968/.962	.943/.990
CNN2-VA [13]	.803	.942/.879	.910/.893	.703/.980
CNN2-FT-VA	.931	.977/.962	.970/.966	.891/.990
FOMAML-VA [56]	.943	.973/.947	.960/.955	.933/.986
MAML-VA [47]	.940	.972/.951	.961/.956	.930/.987
Reptile-VA [48]	.872	.913/.905	.909/.898	.868/.957
Meta-Origin-VA [41]	.956	.974/.949	.961/.948	.949/.987
Meta-VA	**.967**	**.982/.961**	**.974/.967**	**.956/.994**

TABLE V
PERFORMANCES OF METHODS ON EVENTS IN VA DETECTION.

Methods	F1	Se/Sp	BAC/ACC	PPV/NPV
Classic-VA [24]	.945	.972/.917	.936/.935	.912/.952
CNN1-VA [41]	.936	.969/.882	.926/.890	.925/.946
CNN1-FT-VA	.960	.967/.960	.963/.965	.959/.966
CNN2-VA [13]	.876	.966/.864	.915/.900	.805/.978
CNN2-FT-VA	.946	.968/.952	.962/.959	.923/.983
FOMAML-VA [56]	.962	.969/.944	.957/.956	.965/.957
MAML-VA [47]	.969	.972/.953	.963/.965	.979/.965
Reptile-VA [48]	.874	.876/.907	.892/.879	.907/.925
Meta-Origin-VA [41]	.970	.977/.950	.963/.960	.974/.971
Meta-VA	**.982**	**.983/.963**	**.970/.972**	**.989/.992**

on the *random.seed* of *numpy*. The SVM method (i.e., Classic-HAR) is implemented using sklearn library of Python. The other methods (i.e., Classic-VA and Classic-AF) are simulated using Python as well. All those experiments run on the PC with 8 cores of Intel i9 9900K CPU, 32 GB RAM, 512 GB SSD, and an NVIDIA GeForce GTX 2080Ti GPU on Ubuntu 16.04. The STM32F469NI discovery kit (with 2 MB flash and 324 KB SRAM) [57] is utilized as the IoT health monitor. *STM32Cube.AI* [58] developed by ST is utilized to deploy the model on the board. A Raspberry Pi 4B (with Cortex-A72, 8 GB RAM, and 3.5 W in operation) [59] is utilized as the edge device for CNN fine-tuning (personalization).

C. Results

1) Detection Performance: We evaluate our meta-learning method against other methods in terms of various metrics including F1 score (F1), Sensitivity (Se), Specificity (Sp), balanced accuracy (BAC), accuracy (ACC), positive predictive value (PPV), and negative predictive value (NPV). Note that all the metrics are calculated based on the average performance of each patient in the testing set.

We first present VA detection performance on IEGMs from AAEL. The condition positive is VA and the condition negative is non-VA. Table IV demonstrates the detection performance on VA segments. The performances indicate that fine-tuning is necessary for CNN models to perform patient-specific VA detection since almost all metrics of CNN1-FT-VA and CNN2-FT-VA improve after being fine-tuned on the personal data. As shown in Table IV, Meta-VA achieves the best performance on all metrics compared with other evaluated methods. It indicates that the generalization of the CNN initialization is critical in model personalization and our proposed meta-learning method provides an effective solution to the problem.

TABLE VI
PERFORMANCES OF METHODS ON SEGMENTS IN AF DETECTION.

Methods	F1	Se/Sp	BAC/ACC	PPV/NPV
CNN-AF [29]	.838	**.952**/.926	.939/.939	.823/**.966**
CNN-FT-AF	.859	.932/.958	.945/.958	.854/.956
FOMAML-AF [56]	.839	.888/.972	.930/.941	.864/.935
MAML-AF [47]	.841	.879/.972	.926/.936	**.875**/.928
Reptile-AF [48]	.791	.708/.854	.811/.823	.870/.875
Meta-Origin-AF [41]	.856	.933/.952	.943/.950	.846/.961
Meta-AF	**.866**	.918/**.973**	**.946/.960**	.872/.946

TABLE VII
PERFORMANCES OF METHODS ON EVENTS IN AF DETECTION.

Methods	F1	Se/Sp	BAC/ACC	PPV/NPV
Classic-AF [10]	.846	.921/.865	.917/.911	.797/**.958**
CNN-AF [29]	.800	**.945**/.898	.921/.922	.776/.943
CNN-FT-AF	.844	.920/.939	.930/.941	.837/.953
FOMAML-AF [56]	.839	.886/.956	.921/.939	.861/.943
MAML-AF [47]	.844	.882/.965	.924/.941	**.875**/.940
Reptile-AF [48]	.746	.753/.860	.796/.800	.749/.878
Meta-Origin-AF [41]	.823	.926/.938	.932/.947	.813/.949
Meta-AF	**.852**	.910/**.965**	**.936/.947**	.861/.949

Table V illustrates the detection performance on VA events. The performance on events is more practical in real-world scenarios since the defibrillation therapy should be determined based on the rhythm episodes instead of segment in conventional VA detection in ICDs [11]. Therefore, we leverage a simple but effective mechanism to determine VA events for all CNN models. That is, the VA rhythm would be determined if there are 4 consecutive VA predictions on the 2-second segments. In other words, the monitor would consistently monitor the latest four inferences, and the detection period is 8 seconds. The criteria is set since the detection period of the classic method for VAs detection in ICDs is usually 5 to 10 seconds [23].

Compared with Classic-VA, CNN1-VA achieves a 0.3% deduction from a baseline of 97.2% on VA event detection rate represented Se and a 3.5% deduction on non-VA event detection rate from a baseline of 91.7% represented by Sp. The two metrics, Se and Sp, become 96.7% and 96.0% respectively after fine-tuning in CNN1-FT-VA. The performances of CNN2-FT-VA also indicate that the fine-tuning could further improve the detection performance for the pre-trained deep model. The two SOTA meta-learning approaches, FOMAML-VA and MAML-VA, achieve better performances on VA detection compared with CNN1-VA-FT and CNN2-VA-FT in terms of F1 score. As for Meta-VA, it achieves the best performance on all evaluated metrics. It has the near-optimal detection rate on VA events (98.3%) and non-VA events (96.3%), and the highest F1 score (0.982). When compared with Meta-Origin-VA [41], the performances of Meta-VA show that the proposed two optimization techniques could further improve the generalization of the model initialization to fit patient-specific detection.

Table VI shows the AF detection performance on segments level. The condition positive is AF and the condition negative is non-AF. As shown in the table, simply fine-tuning on the pre-trained model CNN-AF is not an effective approach since some metrics (e.g., Se and NPV) degrades when comparing

TABLE VIII
PERFORMANCES OF METHODS ON SEGMENTS IN HAR.

Methods	Macro-F1	ACC	F1-Walking	F1-Upstairs	F1-Downstairs	F1-Sitting	F1-Standing	F1-Laying
Classic-HAR [40]	**0.968**	**0.967**	0.979	**0.981**	**0.994**	**0.920**	**0.932**	**1.000**
CNN-HAR [33]	0.919	0.916	**0.990**	0.973	0.984	0.778	0.803	0.987
CNN-FT-HAR	0.916	0.914	0.973	0.966	0.982	0.763	0.820	0.989
FOMAML-HAR [56]	0.941	0.937	0.981	0.981	0.982	0.844	0.870	0.990
MAML-HAR [47]	0.936	0.933	0.982	0.966	0.979	0.841	0.866	0.980
Reptile-HAR [48]	0.690	0.714	0.512	0.614	0.533	0.698	0.804	0.981
Meta-Origin-HAR [41]	0.934	0.931	0.983	0.968	0.976	0.834	0.851	0.992
Meta-HAR	0.945	0.941	0.988	0.974	0.981	0.860	0.882	0.990

TABLE IX
PERFORMANCES OF METHODS ON EVENTS IN HAR.

Methods	Macro-F1	ACC	F1-Walking	F1-Upstairs	F1-Downstairs	F1-Sitting	F1-Standing	F1-Laying
Classic-HAR [40]	**0.995**	**0.993**	**1.000**	0.983	**1.000**	**0.983**	0.983	**1.000**
CNN-HAR [33]	0.913	0.925	0.983	**1.000**	0.994	0.739	0.761	1.000
CNN-FT-HAR	0.928	0.944	1.000	0.983	0.994	0.767	0.822	1.000
FOMAML-HAR [56]	0.959	0.969	0.983	1.000	0.994	0.900	0.878	1.000
MAML-HAR [47]	0.959	0.969	1.000	0.967	0.989	0.906	0.911	0.983
Reptile-HAR [48]	0.812	0.856	0.689	0.911	0.858	0.644	0.783	0.983
Meta-Origin-HAR [41]	0.965	0.972	1.000	1.000	1.000	0.906	0.883	1.000
Meta-HAR	0.985	0.989	1.000	0.983	0.994	0.950	**0.983**	1.000

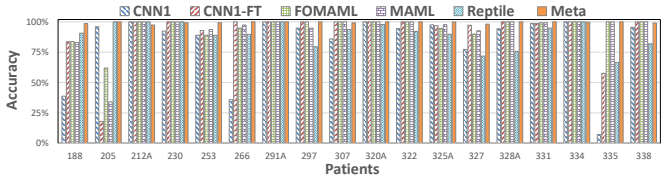
CNN-FT-AF with CNN-AF. It indicates that the generalization of deep model initialization is essential for patient-specific detection. In meta-learning methods, both FOMAML-AF and MAML-AF achieve relatively similar performance when compared with CNN-FT-AF. Reptile-AF achieves the worst detection performances among all evaluated methods. On the other hand, Meta-AF achieves the best detection performances on almost all metrics except Se, PPV and NPV. As shown in Table VI, Meta-AF achieves the highest F1 score (0.866) among all methods. The total accuracy of Meta-AF is 96.0%, together with detection accuracy on AF segments being 91.8% and non-AF segments being 97.3%. It indicates that the proposed meta-learning method could also adapt to the ECG domain by generating a well-generalized model initialization.

Table VII shows the AF detection performance on events level. Here, we leverage a mechanism to determine AF events for all CNN models, where the AF event would be determined if there are 3 consecutive AF predictions on the 10-second segments. As shown in the table, Meta-AF achieves a 10.0% increase from a baseline of 86.5% on non-AF event detection rate represented by Sp, and a 3.6% increase on accuracy from a baseline of 91.1% of Classic-AF. As for SOTA meta-learning methods, FOMAML-AF and MAML-AF achieve relatively comparable performance when compared with CNN-FT-AF in terms of F1. It indicates that the devised patient-wise tasks formatting strategy in Meta-AF could increase the meta-model generalization and further improve the detection performance.

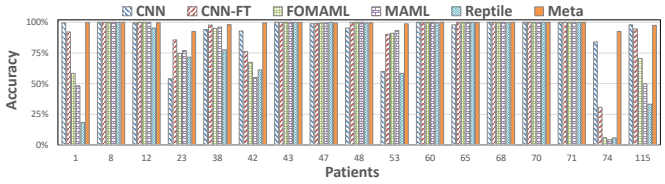
Table VIII shows the activity recognition performance in terms of Macro-F1 (i.e., average F1 score over 6 activities), total accuracy, and F1 over each activity classification over segments. As shown in the table, Classic-HAR achieves the best performance over almost all metrics except F1-Walking. As for CNN-FT-HAR, its performance degrades after fine-tuning using a limited amount of personal data. It again indicates that the quality of the generalization of model initialization is critical in patient-specific detection. SOTA meta-learning methods such as FOMAML-HAR, MAML-HAR, and Retiple-HAR do not achieve significant performance improvement due to the training tasks formatting issue as introduced in Section III-B. As for Meta-HAR, it achieves 0.945 Macro-F1 score and 94.1% total accuracy, which are the second-best activity recognition performance among all evaluated methods.

Table IX shows the performance on events level of HAR. Here, we devise a mechanism to classify events for all methods. The classification of the event would be considered correct if there are more than half of the number of segments to be predicted correctly by the method. As shown in the table, Meta-HAR achieves even better performances than other deep learning based methods in the classification of the action Sitting (i.e., 0.950 in F1-Sitting) and Standing (i.e., 0.983 in F1-Standing). Although Classic-HAR still achieves the best performances over almost all metrics, the performance gap between Meta-HAR and Classic-HAR has been significantly narrowed. When compared with Classic-HAR, Meta-HAR achieves 0.985 F1 score with only 1.0% differences, and 98.9% total accuracy with only 0.4% differences. Furthermore, Meta-HAR achieves the best activity recognition performance among all deep learning based HAR methods in terms of Macro-F1 and accuracy.

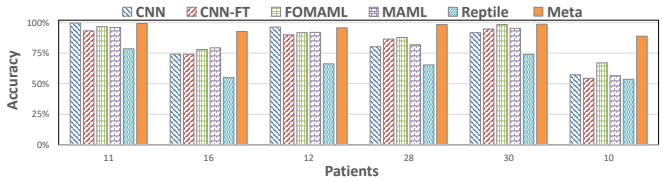
Here, we just give out a discussion on the performance gap between Classic-HAR and other deep learning based methods. The extensive amount of the hand-designed and carefully-chosen features enables Classic-HAR to properly classify the action based on the knowledge that has been fully explored by the experts first. For the other deep learning based methods, the deep model could only learn to correctly classify the action from being trained on the relatively limited amount of the labeled signal segments, which severely restricts the detection performance. However, on the other hand, as shown in Table IX, the performance gap between our method and Classic-HAR has been greatly narrowed on event-level detection through the proposed optimizations. Moreover, deep learning based methods could learn to classify the action by itself with only labeled data. When compared with Classic-HAR, the



(a) Individual accuracy in VA detection.



(b) Individual accuracy in AF detection.



(c) Individual accuracy in HAR.

Fig. 5. Individual detection accuracy on all methods over three applications.

deep learning based methods require much less expertise in the development process. Moreover, the deep models can be deployed on the IoT monitors while Classic-HAR is hardly runnable due to the hardware resources constraints.

To recall the performances reported in Fig. 2, we have demonstrated the individual accuracy on the same patients chosen in Fig. 2 of each application. In VA detection shown in Fig. 5(a), Meta-VA achieves a stable and high detection accuracy over all testing patients while other methods' detection accuracy would degrade on different certain patients. In AF detection shown in Fig. 5(b), on patient 42 and 74, Meta-AF could achieve a much higher detection accuracy while other methods do not detect well. The same case has been observed in Fig. 5(c) on patient 10 and 16 as well. It indicates that our method could effectively personalize the deep model for the individual with well-generalized model parameters.

2) Model Generalization: To demonstrate the generalization of model initialization obtained from different methods, we present the average accuracy and loss curve over all testing patients during personalization on each case study application. The 5-step gradient descent (GD) is applied to personalize each pre-trained model. The evaluated methods include fine-tuning, FOMAML, MAML, Reptile, and the proposed method Meta.

As shown in Fig. 6, at step 0, the initial model of our Meta method does not perform better compared with other methods in terms of accuracy. The model initialization of Meta could rapidly adapt to the specific patient's rhythm and end up with higher accuracy. This trend is shown in all three applications in Fig. 6. The other methods such as FOMAML, and MAML do not appear in the same trend during personalization as demonstrated in Fig. 6(b) and Fig. 6(c). The averaged individual accuracy of those methods does not increase along with update steps and even degrades after fine-tuning. It indicates that the proposed meta-learning method delivers a well-generalized model initialization for model personalization. The loss curves shown in Fig. 6 illustrate that Meta could rapidly converge on the dataset of different applications.

3) Performance on Hardware: We deploy all evaluated CNN models on the board STM32F469NI discovery kit (with ARM Cortex M4) [57] to test its inference performance in terms of energy, latency, and memory overhead on real hardware. Since all meta-learning methods utilize the same CNN architecture as CNN-based detection methods, the performances of model inference would be relatively similar over the same application. In other words, the practical performances of 4 CNN models could represent the performances of all evaluated deep learning based approaches since meta-learning methods do not interfere with the processes of model inference and fine-tuning. The models require only 36 KB, 28 KB, 319 KB, and 295 KB to store model parameters for CNN1-VA, CNN2-VA, CNN-AF, and CNN-HAR respectively. The average latency on the inference over a segment is 9.94 ms, 8.44 ms, 64.5 ms, and 37.04 ms for CNN1-VA, CNN2-VA, CNN-AF, and CNN-HAR respectively. The power of the testing board is 161 mW (supplied with 5 V). It indicates that the model could meet the hardware constraints of the implantable or wearable devices for detection tasks [60], [9]. The fine-tuning overhead on the edge device (i.e., Raspberry Pi 4B in the experiments) is 1.74 s, 1.73 s, 5.32 s, and 5.12 s over CNN1-VA, CNN2-VA, CNN-AF, and CNN-HAR respectively. The performance of fine-tuning overhead indicates that it is capable to conduct the deep model personalization on the user end. With the design of the edge computing framework, local personalization could significantly reduce the risks of personal data and model leakage.

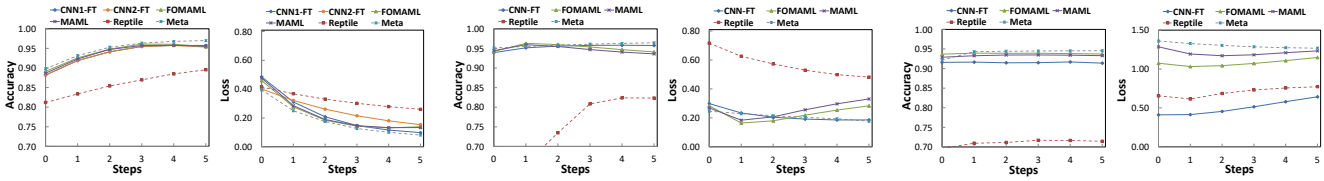
VI. Conclusions

In this paper, we propose a novel meta-learning method for patient-specific detection on the resource-constrained IoT monitors. The meta-learning method aims to generate a well-generalized model initialization for the model to be personalized on patient-specific data. A novel patient-wise training tasks formatting strategy is presented to address the training sample mixture problem in conventional meta-learning methods. The inner- and outer-loop optimizations are proposed to further improve the generalization of the meta-model initialization. A computing framework is further developed to provide the capability of local model personalization on the edge devices to avoid data breaches and model manipulation. The deep models personalized by our method achieve 8.2%, 2.5%, and 6.4% higher detection accuracy compared with the existing deep learning methods in VA detection, AF detection, and HAR respectively.

References

[1] A. M. Ghosh, D. Halder, and S. A. Hossain, "Remote health monitoring system through IoT," in *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*. IEEE, 2016, pp. 921–926.

[2] P. Valsalan, T. A. B. Baomar, and A. H. O. Baabood, "IoT based health monitoring system," *Journal of Critical Reviews*, vol. 7, no. 4, pp. 739–743, 2020.



(a) Accuracy and loss curve in VA detection. (b) Accuracy and loss curve in AF detection. (c) Accuracy and loss curve in HAR.

Fig. 6. Accuracy and loss trend during personalization of deep learning based methods over three applications.

[3] J. Wan, M. A. Al-awlaqi, M. Li, M. O'Grady, X. Gu, J. Wang, and N. Cao, "Wearable IoT enabled real-time health monitoring system," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, pp. 1–10, 2018.

[4] J. Zdarek and C. W. Israel, "Detection and discrimination of tachycardia in ICDs manufactured by St. Jude Medical," *Herzschrittmachertherapie+ Elektrophysiologie*, vol. 27, no. 3, pp. 226–239, 2016.

[5] Boston Scientific, Inc, "LUX-Dx™ Insertable Cardiac Monitor (ICM) System," 2017. [Online]. Available: https://www.bostonscientific.com/content/dam/bostonscientific/Rhythm%20Management/portfolio-group/lux-dx-icm/pdf/LUX-Dx-Clinic-Resource-Guide.pdf

[6] iRhythm Technologies, Inc, "iRhythm Zio Patch," 2021. [Online]. Available: https://www.irhythmtech.com

[7] M. Berchtold, M. Budde, D. Gordon, H. R. Schmidtke, and M. Beigl, "Actiserv: Activity recognition service for mobile phones," in *International Symposium on Wearable Computers (ISWC)*, 2010, pp. 1–8.

[8] D. Castro, W. Coral, C. Rodriguez, J. Cabra, and J. Colorado, "Wearable-based human activity recognition using an IoT approach," *Journal of Sensor and Actuator Networks*, vol. 6, no. 4, p. 28, 2017.

[9] F. Samie, L. Bauer, and J. Henkel, "Hierarchical classification for constrained IoT devices: A case study on human activity recognition," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8287–8295, 2020.

[10] H. Pürerfellner, E. Pokushalov, S. Sarkar, J. Koehler, R. Zhou, L. Urban, and G. Hindricks, "P-wave evidence as a method for improving algorithm to detect atrial fibrillation in insertable cardiac monitors," *Heart Rhythm*, vol. 11, no. 9, pp. 1575–1583, 2014.

[11] M. Biffi, "ICD programming," *Indian Heart Journal*, vol. 66, pp. S88–S100, 2014.

[12] A. J. Moss, C. Schuger, C. A. Beck, M. W. Brown, D. S. Cannom, J. P. Daubert, N. M. Estes III, H. Greenberg, W. J. Hall, D. T. Huang *et al.*, "Reduction in inappropriate therapy and mortality through ICD programming," *New England Journal of Medicine*, vol. 367, no. 24, pp. 2275–2283, 2012.

[13] U. R. Acharya, H. Fujita, S. L. Oh, U. Raghavendra, J. H. Tan, M. Adam, A. Gertych, and Y. Hagiwara, "Automated identification of shockable and non-shockable life-threatening ventricular arrhythmias using convolutional neural network," *Future Generation Computer Systems*, vol. 79, pp. 952–959, 2018.

[14] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network," *Nature Medicine*, vol. 25, no. 1, p. 65, 2019.

[15] S. Mousavi, F. Afghah, A. Razi, and U. R. Acharya, "ECGNET: Learning where to attend for detection of atrial fibrillation with deep visual attention," in *2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*. IEEE, 2019, pp. 1–4.

[16] R. R. van de Leur, L. J. Blom, E. Gavves, I. E. Hof, J. F. van der Heijden, N. C. Clappers, P. A. Doevendans, R. J. Hassink, and R. van Es, "Automatic triage of 12-lead ECGs using deep convolutional neural networks," *Journal of the American Heart Association*, vol. 9, no. 10, p. e015138, 2020.

[17] S.-M. Lee, S. M. Yoon, and H. Cho, "Human activity recognition from accelerometer data using convolutional neural network," in *2017 International Conference on Big Data and Smart Computing (BIGCOMP)*. IEEE, 2017, pp. 131–134.

[18] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2015.

[19] T. Golany and K. Radinsky, "PGANs: Personalized generative adversarial networks for ECG synthesis to improve patient-specific deep ECG classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 557–564.

[20] S. A. Rokni, M. Nourollahi, and H. Ghasemzadeh, "Personalized human activity recognition using convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[21] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *arXiv preprint arXiv:1703.03400*, 2017.

[22] A. Antoniou, H. Edwards, and A. Storkey, "How to train your maml," *arXiv preprint arXiv:1810.09502*, 2018.

[23] M. Madhavan and P. A. Friedman, "Optimal programming of implantable cardiac-defibrillators," *Circulation*, vol. 128, no. 6, pp. 659–672, 2013.

[24] N. Zanker, D. Schuster, J. Gilkerson, and K. Stein, "Tachycardia detection in icds by Boston Scientific," *Herzschrittmachertherapie+ Elektrophysiologie*, vol. 27, no. 3, pp. 186–192, 2016.

[25] P. M. Barrett, R. Komatireddy, S. Haaser, S. Topol, J. Sheard, J. Encinas, A. J. Fought, and E. J. Topol, "Comparison of 24-hour holter monitoring with 14-day novel adhesive patch electrocardiographic monitoring," *The American journal of medicine*, vol. 127, no. 1, pp. 95–e11, 2014.

[26] D. R. Seshadri, B. Bittel, D. Browsky, P. Houghtaling, C. K. Drummond, M. Y. Desai, and A. M. Gillinov, "Accuracy of apple watch for detection of atrial fibrillation," *Circulation*, vol. 141, no. 8, pp. 702–703, 2020.

[27] E. M. Tapia, S. S. Intille, W. Haskell, K. Larson, J. Wright, A. King, and R. Friedman, "Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor," in *2007 11th IEEE International Symposium on Wearable Computers*. IEEE, 2007, pp. 37–40.

[28] M. A. Ayu, S. A. Ismail, T. Mantoro, and A. F. A. Matin, "Real-time activity recognition in mobile phones based on its accelerometer data," in *2016 International Conference on Informatics and Computing (ICIC)*. IEEE, 2016, pp. 292–297.

[29] C.-H. Hsieh, Y.-S. Li, B.-J. Hwang, and C.-H. Hsiao, "Detection of atrial fibrillation using 1D convolutional neural network," *Sensors*, vol. 20, no. 7, p. 2136, 2020.

[30] Y. Xia, N. Wulan, K. Wang, and H. Zhang, "Detecting atrial fibrillation by deep convolutional neural networks," *Computers in Biology and Medicine*, vol. 93, pp. 84–92, 2018.

[31] Y. Chen and Y. Xue, "A deep learning approach to human activity recognition based on single accelerometer," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2015, pp. 1488–1492.

[32] H. H. Pham, H. Salmane, L. Khoudour, A. Crouzil, P. Zegers, and S. A. Velastin, "Spatio–temporal image representation of 3D skeletal movements for view-invariant action recognition with deep convolutional neural networks," *Sensors*, vol. 19, no. 8, p. 1932, 2019.

[33] A. Ignatov, "Real-time human activity recognition from accelerometer data using convolutional neural networks," *Applied Soft Computing*, vol. 62, pp. 915–922, 2018.

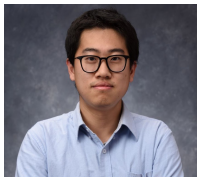
[34] S. Bhati, L. M. Velazquez, J. Villalba, and N. Dehak, "Lstm siamese network for parkinson's disease detection from speech," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2019, pp. 1–5.

[35] A. Papadopoulos, D. Iakovakis, L. Klingelhoefer, S. Bostantjopoulou, K. R. Chaudhuri, K. Kyritsis, S. Hadjidimitriou, V. Charisis, L. J. Hadjileontiadis, and A. Delopoulos, "Unobtrusive detection of parkinson's disease from multi-modal and in-the-wild sensor data using deep learning techniques," *Scientific Reports*, vol. 10, no. 1, pp. 1–13, 2020.

[36] S. Chambon, M. N. Galtier, P. J. Arnal, G. Wainrib, and A. Gramfort, "A deep learning architecture for temporal sleep stage classification using multivariate and multimodal time series," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 4, pp. 758–769, 2018.

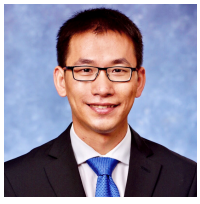
[37] Z. Jia, Y. Lin, J. Wang, X. Wang, P. Xie, and Y. Zhang, "SalientSleepNet: Multimodal salient wave detection network for sleep staging," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021, pp. 2614–2620.

[38] AnnArbor, "Ann Arbor Electrogram Libraries, Chicago IL, USA," 2003.
[39] S. Petrutiu, A. V. Sahakian, and S. Swiryn, "Abrupt changes in fibrillatory wave characteristics at the termination of paroxysmal atrial fibrillation in humans," *Europace*, vol. 9, no. 7, pp. 466–470, 2007.
[40] D. Micucci, M. Mobilio, and P. Napoletano, "Unimib shar: A dataset for human activity recognition using acceleration data from smartphones," *Applied Sciences*, vol. 7, no. 10, p. 1101, 2017.
[41] Z. Jia, Z. Wang, F. Hong, L. Ping, Y. Shi, and J. Hu, "Learning to learn personalized neural network for ventricular arrhythmias detection on intracardiac EGMs," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021, pp. 2606–2613.
[42] A. Kamrud, B. Borghetti, and C. Schubert Kabban, "The effects of individual differences, non-stationarity, and the importance of data partitioning decisions for training and testing of eeg cross-participant models," *Sensors*, vol. 21, no. 9, p. 3225, 2021.
[43] A. M. Thøgersen, J. M. Larsen, J. B. Johansen, M. Abedin, and C. D. Swerdlow, "Failure to treat life-threatening ventricular tachyarrhythmias in contemporary implantable cardioverter–defibrillators: Implications for strategic programming," *Circulation: Arrhythmia and Electrophysiology*, vol. 10, no. 9, p. e005305, 2017.
[44] S. Baghersalimi, T. Teijeiro, D. Atienza, and A. Aminifar, "Personalized real-time federated learning for epileptic seizure detection," *IEEE Journal of Biomedical and Health Informatics*, 2021.
[45] F. Forooghifar, A. Aminifar, L. Cammoun, I. Wisniewski, C. Ciumas, P. Ryvlin, and D. Atienza, "A self-aware epilepsy monitoring system for real-time epileptic seizure detection," *Mobile Networks and Applications*, pp. 1–14, 2019.
[46] G. Bhat, N. Tran, H. Shill, and U. Y. Ogras, "w-HAR: An activity recognition dataset and framework using low-power wearable devices," *Sensors*, vol. 20, no. 18, p. 5356, 2020.
[47] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *arXiv preprint arXiv:1703.03400*, 2017.
[48] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.
[49] P. Verma and S. K. Sood, "Fog assisted-IoT enabled patient health monitoring in smart homes," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1789–1796, 2018.
[50] I. Azimi, A. Anzanpour, A. M. Rahmani, T. Pahikkala, M. Levorato, P. Liljeberg, and N. Dutt, "HiCH: Hierarchical fog-assisted computing architecture for healthcare IoT," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 5s, pp. 1–20, 2017.
[51] "Cybersecurity Vulnerabilities - Medtronic Implantable Cardia Devices." [Online]. Available: https://www.fda.gov/medical-devices/safety-communications/cybersecurity-vulnerabilities-affecting-medtronic-implantable-cardiac-devices-programmers-and-home
[52] "SweynTooth Cybersecurity Vulnerabilities May Affect Certain Medical Devices: FDA Safety Communication." [Online]. Available: https://www.fda.gov/medical-devices/safety-communications/sweyntooth-cybersecurity-vulnerabilities-may-affect-certain-medical-devices-fda-safety-communication
[53] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 464–472.
[54] F. Simon, J. P. Martinez, P. Laguna, B. van Grinsven, C. Rutten, and R. Houben, "Impact of sampling rate reduction on automatic ECG delineation," in *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2007, pp. 2587–2590.
[55] E. A. P. Alday, A. Gu, A. J. Shah, C. Robichaux, A.-K. I. Wong, C. Liu, F. Liu, A. B. Rad, A. Elola, S. Seyedi *et al.*, "Classification of 12-lead ECGs: The physionet/computing in cardiology challenge 2020," *Physiological measurement*, 2020.
[56] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the convergence theory of gradient-based model-agnostic meta-learning algorithms," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 1082–1092.
[57] STMicroelectronics, "Discovery kit with STM32F469NI MCU," 2020. [Online]. Available: https://www.st.com/resource/en/user_manual/dm00218846-discovery-kit-with-stm32f469ni-mcu-stmicroelectronics.pdf
[58] STMicroelectronics, "STM32 solutions for Artificial Neural Networks," 2020. [Online]. Available: https://www.st.com/content/st_com/en/ecosystems/stm32-ann.html
[59] Raspberry Pi Foundation, "RASPBERRY PI 4 MODEL B," 2019. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b
[60] C. Strydis, R. M. Seepers, P. Peris-Lopez, D. Siskos, and I. Sourdis, "A system architecture, processor, and communication protocol for secure implants," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 10, no. 4, pp. 1–23, 2013.



Zhenge Jia is currently Ph.D. candidate at the Department of Electrical and Computer Engineering, University of Pittsburgh. He received B.S. degree with H2A from College of Engineering and Computer Science at Australian National University in 2017. His research interests include machine learning in medical applications, embedded system, and edge computing.

Yiyu Shi is currently a professor in the Department of Computer Science and Engineering at the University of Notre Dame, the site director of National Science Foundation I/UCRC Alternative and Sustainable Intelligent Computing, and the director of the Sustainable Computing Lab (SCL). He received his B.S. in Electronic Engineering from Tsinghua University, Beijing, China in 2005, the M.S and Ph.D. degree in Electrical Engineering from the University of California, Los Angeles in 2007 and 2009 respectively. His current research interests focus on hardware intelligence and biomedical applications. In recognition of his research, more than a dozen of his papers have been nominated for or awarded as the best paper in top journals and conferences, including the 2021 IEEE Transactions on Computer-Aided Design Donald O Pederson Best Paper Award. He is also the recipient of Facebook Research Award, NSF CAREER Award, IEEE Region 5 Outstanding Individual Achievement Award, IEEE Computer Society Mid-Career Research Achievement Award, among others. He has served on the technical program committee of many international conferences. He is the deputy editor-in-chief of IEEE VLSI CAS Newsletter, and an associate editor of various IEEE and ACM journals. He is an IEEE CEDA distinguished lecturer and an ACM distinguished speaker.



Jingtong Hu is currently an Associate Professor in the Department of Electrical and Computer Engineering at University of Pittsburgh, Pittsburgh, PA, USA. Before that, he was an Assistant Professor at Oklahoma State University from 2013 to 2017. His current research interests include hardware/software co-design for machine learning algorithms, on-device AI, embedded systems. In the past several years, his works have received one best paper award and 5 best paper nominations. He is also the recipient of Oklahoma State University Outstanding New Faculty Award, Air Force Summer Faculty Fellowship, and ACM SIGDA Meritorious Service Award. He has served on the technical program committee of many international conferences. He served as a guest editor for IEEE Transactions on Computers and is currently an associate editor for IEEE Embedded Systems Letters and ACM Transactions on Cyber-Physical Systems.