Energy-Efficient Missing Data Recovery in Wearable Devices: A Novel Search-based Approach

Dina Hussein*, Taha Belkhouja*, Ganapati Bhat, and Janardhan Rao Doppa
School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, 99164

***Abstract*—Wearable and internet of things (IoT) devices are transforming a number of high-impact applications. Machine learning (ML) algorithms on wearable devices assume that data from all sensors is available at runtime. However, one or more sensors may be unavailable at runtime due to malfunction, energy constraints or communication challenges. Loss of sensor data can potentially lead to severe degradation in application accuracy and quality of service. Commonly employed generative ML methods to recover missing data are not suitable for resource-constrained wearables because they incur significant memory, execution time, and energy overhead at runtime. In contrast to prior methods, this paper presents a novel search-based accuracy-preserving imputation (AIM) algorithm that obtains most likely imputation patterns of sensor data for each missing data scenario via *offline* analytics. Specifically, for each missing data condition, we store the most likely recovery patterns which preserve ML classifier-based application accuracy in a look up table and use it appropriately at runtime. The key insight behind AIM is that we do not need exact recovery of the missing data as long as the ML classifier-based application accuracy (e.g., health assessment) is preserved. To further improve the overall effectiveness of AIM, we train the ML classifiers to be robust to small errors in data recovery. Experiments on four diverse wearable sensor based time-series benchmarks demonstrate that AIM is able to maintain accuracy within 5% of the baseline with no missing data when one sensor is missing, and improves the overall accuracy by 15% compared to a state-of-the-art baseline. AIM achieves this improvement with negligible energy consumption overhead.**

I. Introduction

Wearable and internet of things (IoT) devices are enabling several challenging and high-impact applications such as health monitoring, rehabilitation, and fitness tracking [1–3]. They are also employed in mobile health applications including diagnosis of movement disorders such as essential tremor, and Parkinson's disease by monitoring the respective biomarkers [4, 5]. Wearable devices typically implement these applications by collecting data from various sensors mounted on the body and processing them using machine learning (ML) algorithms to make data-driven predictions and decisions.

One of the key assumptions made by wearable applications including the above-mentioned ones is that data from all sensors is available at runtime. However, data from one or more sensors might be missing during real-world, runtime usage. Sensor data may be missing due to energy limitations, user error, sensor malfunction, or data communication challenges [6, 7]. Missing data leads to significant degradation in the overall quality of applications since the underlying ML models are trained with the assumption that data from all sensors is available. Specifically, our experiments on real-world applications show that missing data from even a single sensor can degrade the accuracy of ML models by as much as 20%, as shown in Figure 1. Training multiple models for each combination of sensors is also not feasible due to increased memory overhead and context switching requirements at runtime. Therefore, there is a strong need to develop approaches that **1)** recover missing data at runtime to preserve the accuracy of a given ML-driven application; and **2)** have low-overhead in terms of execution time, power consumption, and memory footprint as wearable devices are resource-constrained.

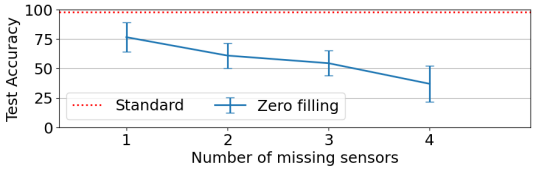

Fig. 1: Classification accuracy (mean and standard deviation) of a deployed ML classifier on Shoaib [8] dataset as a function of k missing sensors.

Recent work has proposed imputation methods that aim to recover raw sensor data at runtine [9–12]. These approaches typically use statistical methods or deep generative networks. While imputation networks are able to recover accuracy loss, they suffer from high memory overhead since we have to store parameters for $2^n - 2$ possible missing data scenarios with n sensors on the wearable device. They also incur additional execution overhead to produce imputed data.

This paper presents a novel and energy-efficient search-based *Accuracy-Preserving Imputation (AIM)* approach to produce accuracy-preserving imputations for missing sensor data at runtime. The AIM method meets the desiderata of an effective solution for missing sensor data based on two *synergistic* principles. **First**, we do not need to recover the exact missing sensor data as long as the application accuracy is preserved. **Second**, we can train the ML model to be robust to small deviations from the exact sensor data, i.e., ability to make accurate predictions for sensor data with small perturbations. To instantiate the first principle, AIM starts with a set of possible scenarios for missing sensor data and formulates a search problem whose goal is to obtain the most likely imputation pattern for the missing sensors using the given training data with no missingness. When we run the search algorithm for each of the $2^n - 2$ missing data scenarios, we get a table of imputation patterns for each scenario. To

*D. Hussein and T. Belkhouja contributed equally.

979-8-3503-1175-4/23/$31.00 ©2023 IEEE

instantiate the second principle, we train robust ML models using augmented data (i.e., perturbations of the clean and complete training data) to make accurate predictions when the imputed data deviates from the true data.

We validate the proposed AIM approach on four diverse wearable sensor-based time series benchmark datasets [8, 13–15]. For each of these applications, we enumerate all possible scenarios of missing sensor data and execute AIM for imputing the missing data. Our results demonstrate that for up to 2 missing sensors, AIM achieves accuracy within 5% of the upper-bound baseline with no missing data. This is a remarkable result because the likelihood of one or two sensors missing is higher than a larger number of sensors being unavailable in the real-world. The application accuracy with AIM drops with more than two missing sensors, however, it is still higher than the accuracy with no data recovery and generative imputation networks. We also compare AIM with a recent generative imputation network approach (GAIN) [12]. Compared to GAIN, our AIM approach achieves over 15% higher prediction accuracy on average with significantly lower energy and memory overhead. Specifically, our measurements on the Odroid-XU3 device [16] show that AIM enables 78–98% energy savings over GAIN, thus almost doubling the operating time of wearable devices.

Contributions: This paper makes the following contributions:

- Characterization of the accuracy loss when one or more sensors are unavailable in wearable applications.
- Novel and energy-efficient accuracy-preserving imputation approach called AIM to identify the most likely data patterns for imputation of missing sensor data.
- Experimental validation on four diverse wearable datasets to demonstrate that AIM enables reliable imputation of missing sensor data with minimal overhead.

II. Related Work

Wearable devices are being increasingly used in health applications [1–3, 17]. Integrating multiple sensors increases the likelihood of one or more sensors being unavailable due to energy constraints, user error, or communication challenges. Therefore, there is a strong need for energy-efficient and accuracy-preserving methods to recover missing data.

Recent work has proposed several methods to handle missing data in sensor based applications [9, 10, 12, 18, 19]. These approaches typically use statistical or deep generative methods to recover the missing data. Statistical methods, such as mean, median, or regression use available data around the missing instances to obtain an imputation [9, 10]. As such, they are suitable to handle isolated missing data instances where data around the missing samples is available. However, they are not suitable for long sequences of sensor unavailability with no reference data for statistical methods, which is the focus of this paper. Deep generative methods have been recently proposed to handle longer sequences of missing data [12, 20]. For instance, [12] employs a generative adversarial imputation network (GAIN) to impute the data. Specifically, the GAIN approach imputes the missing data conditioned upon observed data. Similarly, the work in [20] employs deep auto-encoder models to impute EEG recordings from multiple patients and data collection days. However, the primary limitation of deep learning methods is the high memory overhead and the energy cost of performing imputation at runtime.

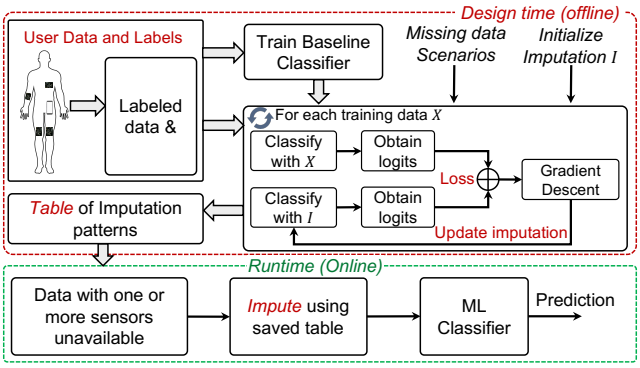

Fig. 2: Overview of the proposed accuracy-preserving imputation approach.

To precisely fill this gap in the current knowledge, this paper develops a novel and energy-efficient accuracy-preserving imputation method to handle missing sensor data at runtime.

III. Background and Problem Setup

This section first provides the background on wearable devices and introduces the missing sensor data problem.

A. Wearable Devices Preliminaries

We consider wearable systems with multiple sensors mounted on the body, as shown in Figure 2. The sensors are used to monitor physical and physiological parameters for applications including mobile health, which we use as the running example. Since the sensor data is continuous and streaming in nature, wearable devices perform the following steps to enable health assessment.

Data Segmentation: The streaming time series sensor data must be divided into equal sized windows for periodic health assessment and to provide fixed-sized inputs to the ML models. Given n sensors and T samples in each window, we denote the time-series sensor data with the variable $X \in \mathbb{R}^{n \times T}$.

Feature Generation and Classification: The time-series sensor data in each window $X \in \mathbb{R}^{n \times T}$ are fed into feature generation and classifier blocks to perform the health assessment. Labeled pairs of sensor data $X \in \mathbb{R}^{n \times T}$ and the class label y are used to train a classifier F_θ, where θ are the parameters for the classifier. At runtime, the sensor data and trained classifier are used to predict class labels of interest.

B. Missing Sensor Data and Imputation Challenges

Depending on the length of unavailability, we can classify the missing data patterns into two main categories as follows.

Random Missing Data: In this case, the sensor encounters isolated missing samples that are not clustered around any particular time instance. Prior work has proposed a number of approaches to handle random missing data [9, 10]. Random missing data is typically easier to handle since data around the

missing instance is available for imputation. Therefore, we do not consider the random missing case and focus our attention on the more *challenging* block missing case described next.

Block Missing Data: Block missing data occurs when long sequences of sensor data are unavailable at runtime. The block missing data is challenging to recover since it does not contain any reference data for the missing sensors. Moreover, in a system with n sensors, we can have $2^n - 2$ possible combinations of block missing data.

C. Problem Setup

Let $X \in \mathbb{R}^{n \times T}$ be the time-series sensor data, where n is the total number of sensor channels and T is the number of samples in each input window. We denote every input $X \in \mathbb{R}^{n \times T}$ as $[X_1, \cdots, X_i, \cdots, X_n]$ where $X_i \in \mathbb{R}^T$ corresponds to a channel i of X. The class label for each window is denoted by y. We denote the set of training examples $\mathcal{D}$ as several input X and ground-truth output y pairs. Standard ML algorithms use the given data $\mathcal{D}$ to train a classifier F_θ which takes time-series X as input to predict the corresponding class label $\hat{y}$, where θ stands for parameters of the classifier.

During inference, the data from one or more sensors might be missing. Let $\{j\}_{0 \le j \le n}$ represent the subset of channels that are missing at runtime for a given input X. We denote the new input that has $\{j\}$ missing channels by $\tilde{X}_{\{j\}} \in \mathbb{R}^{n \times T}$:

$$\tilde{X}_{\{j\}} = \begin{cases} 0^T & \text{if } i \in \{j\} \\ X_i & \text{if } i \notin \{j\} \end{cases} \tag{1}$$

For example, for a human activity recognition (HAR) application, when one of the sensors goes missing, its three accelerometer channels data $\{j\} = \{1, 2, 3\}$ will have 0 value.

This missing data in the input results in misclassifications ($\hat{y} \neq y$) by the classifier F_θ as seen in Figure 1. Consequently, the overall performance of the classifier and quality of application service during deployment will degrade significantly.

IV. Search based Accuracy-Preserving Imputation

This section describes the accuracy-preserving imputation (AIM) algorithm to solve missing sensor data challenge for wearable applications. We first provide an overview of the AIM approach and list the two synergistic design principles behind AIM. Next, we describe the details of the algorithmic approaches which instantiate those two design principles.

Overview of AIM Approach: During the offline configuration of AIM, we perform the following two steps sequentially, as shown in Figure 2. First, we train a robust ML classifier F_θ that can make accurate predictions for small perturbations of time-series signals in the training data (i.e., no missingness). This step ensures accuracy even if there are small errors in the imputed values. Second, given the trained ML classifier F_θ, for each candidate missing sensors configuration, we execute a search algorithm to compute the most likely imputation pattern that preserves the accuracy of the classifier F_θ using the training data. The output of this step is a lookup table $\mathcal{I}$ that stores one imputation pattern for each missing configuration. At runtime, given an input X with some missing channels (one for each sensor), we impute the missing data using appropriate imputation pattern from the lookup table $\mathcal{I}$ (i.e., negligible overhead) and then use the classifier F_θ to make prediction.

Design Principles: AIM meets the desiderata of an effective solution for missing sensor data based on two synergistic principles. *1) Accuracy-preserving imputation:* there is no need to recover the exact missing sensor data as long as the accuracy of ML classifier is preserved. *2) Training robust classifiers:* training the ML classifier to make accurate predictions even with small deviations from the exact sensor data distribution will exhibit robustness to small errors in imputed data. Below we provide algorithms to instantiate these two principles.

A. Search Algorithm for Accuracy-Preserving Imputation

Intuition: During the offline training phase with no missing data, the ML classifier for wearable application achieves high accuracy on the given classification task. Since the inputs are multivariate time-series signals, the classifier relies on the information spread across n different channels to make accurate predictions. Prior work has shown that classifiers rely on a subset of critical channels to predict output labels [21]. This means that in case of missing channels, the input possesses enough information to allow the classifier to predict its true label. Therefore, we set our goal to find a recovery data pattern that lets the classifier predict the correct labels from the available channels. The recovery pattern pushes the classifier to predict the output label as if the data from all sensors is available. In summary, we can view our solution as the search for the most likely data pattern to impute the data for missing sensors for preserving the accuracy of the given ML classifier.

Algorithm: Formally, given a ML classifier F_θ, for any input $X = [X_1, \cdots, X_n]$ and a fixed set of missing channels $\{j\}$, we search for an imputation pattern $\mathcal{I}_{j \in \{j\}} \in \mathbb{R}^T$ s.t.:

$$\mathcal{I}_{\{j\}} = \begin{cases} \mathcal{I}_j & \text{if } i \in \{j\} \quad \text{and} \quad F_\theta(X) \approx F_\theta(\mathcal{I}_{\{j\}}) \\ X_i & \text{if } i \notin \{j\} \end{cases} \tag{2}$$

We note that $\mathcal{I}_{j \in \{j\}}$ *does not depend* on the available sensor data of the input X. Every imputation pattern is stored in a look up table indexed by the combination of the missing channels $\{j\}_{0 \le j \le n}$ where the samples are missing. We conduct this search for each missing sensor data configuration during design time (offline) where the goal is to find for every combination of $\{j\}$ missing channels, the corresponding imputation pattern $\mathcal{I}_{j \in \{j\}}$ that yields a prediction similar to the prediction on the original input without any missingness. Hence, at runtime, we use the stored lookup table to select the appropriate imputation pattern with negligible overhead.

We find imputation patterns based on a given set of missing channels $\{j\}_{0 \le j \le n}$ (missing configuration) that preserves the accuracy of classifier F_θ. We define our overall objective for the search of imputation pattern as shown below:

$$\text{Given } \{j\}: \text{ We find } \mathcal{I}_{\{j\}} \text{ s.t. } \forall X, F_\theta(X) \approx F_\theta(\mathcal{I}_{\{j\}}) \tag{3}$$

We compute the imputation pattern to fill values of the missing channels by solving the minimization problem below:

$$\min_{\mathcal{I}_{j \in \{j\}}} \mathcal{L}\Big(\text{Logits}(F_\theta(X)), \text{Logits}(F_\theta(\mathcal{I}_{\{j\}}))\Big) \tag{4}$$

Algorithm 1 AIM Search for accuracy-preserving imputation

Input: Training set $\mathcal{D} = \{X\}$; F_θ, pre-trained classifier on $\mathcal{D} = \{(X, y)\}$; $\{j\}$, missing sensors configuration; MAX_G, maximum iterations for gradient descent; MAX, maximum iterations over all inputs
Output: $\{\mathcal{I}_{j\in\{j\}}\}$, imputation pattern.

```
Random initialization of the set {I_{j∈{j}}}
for i=1, ···, MAX do
    for each training example X do
        for i_G=1, ···, MAX_G do
            Compute the classifier's logits values: lgX = Logits(F_θ(X))
            Compute the classifier's logits values: lgI = Logits(F_θ(I_{j}))
            Estimate the loss L(lgX, lgI)
            Estimate the gradient ∇_{I_j} L for j ∈ {j}
            Perform gradient descent and update {I_j} for j ∈ {j}
        end for
    end for
end for
return imputation pattern {I_{j∈{j}}} as per the requirements of Eq.2
```

The loss function $\mathcal{L}$ over the logits outcome of the classifier is used for accuracy-preserving imputation pattern search. The logits of a classifier are interpreted as the unnormalized predictions for each candidate class label and input time-series signal pair. The role of this loss function is to compute the similarity between the prediction outcomes of the classifier F_θ for the original input example X (no missingness) and the input with the imputed pattern $\mathcal{I}_{\{j\}}$. For example, $\mathcal{L}$ can be the Mean Squared Error between the logits values of both predictions $F_\theta(X)$ and $F_\theta(\mathcal{I}_{\{j\}})$. *When $\mathcal{L} \to 0$, accuracy of the classifier over imputed and original inputs will be similar.*

Algorithm 1 shows the pseudo-code of proposed search approach to compute accuracy-preserving imputation pattern $\mathcal{I}_{\{j\}}$. We set $MAX_G = 100$ to ensure that AIM can find the optimal imputation pattern per example X. Additionally, we set $MAX = 50$ to ensure that AIM optimizes the imputation pattern across the training data. The output of this algorithm is used to populate a look-up table $\mathcal{I}$ that maps the set of missing channels $\{j\}$ to the corresponding imputation pattern $\{\mathcal{I}_j\}$. Therefore, given a missing sensor data configuration $\{j\}$ and any input X at test time, we construct $\mathcal{I}_{\{j\}}$ as shown in Eq. 2 and employ the classifier F_θ to predict the class label after using $\mathcal{I}_{\{j\}}$ to impute the missing data in input X.

B. Training Robust Classifiers for Improved Effectiveness

Recall that we store *one* imputation pattern for each missingness configuration to preserve the accuracy and AIM's imputation strategy does not depend on the available sensor data of the given input X. As a result, AIM has negligible overhead, but ML classifier may not make correct predictions with generic imputation patterns for a small fraction of the input examples. Therefore, we propose to train ML classifiers to be robust to small errors in the imputed data.

The motivation to train robust ML classifiers is two-fold. First, the ML classifier is less sensitive to the natural noise in the training data. As a result, we will be able to find more robust imputation patterns using our search algorithm and robust ML classifier. Second, the ML classifier will be more robust to small errors in the imputed data at the runtime.

We train robust a ML classifier using data augmentation and propose to apply the recent framework of generating augmented data for time-series signals based on their statistical features [22, 23]. The key idea is to generate small perturbations over the original time-series signals in the training data which preserve the statistical features. To instantiate this framework for our specific use-case, we employ the following statistical features: mean absolute error, statistical average, and root mean square. Additionally, for HAR applications, we include the body acceleration feature due to its high relevance.

V. Experiments and Results

This section analyzes the performance of the proposed data recovery approach on four datasets along different dimensions.

A. Experimental Setup

1) Wearable Device Setup: We employ the Odroid-XU3 board [16] for sensor data processing, while noting that any low-power processor can be used. Odroid-XU3 contains four high-performance ARM Cortex-A15 and four low-power Cortex-A7 cores. We use the Odroid-XU3 to store the imputation table and measure the overhead on A7 cores.

2) Datasets: AIM is validated using four datasets described below. To validate the proposed missing data recovery approach, we vary the number of missing sensors for each dataset with n sensors from one to $n-1$.

Shoaib et al. [8]: The Shoaib dataset includes three-axis accelerometer data for 10 users performing seven activities. The dataset has accelerometer sensors at five locations on the body: left pocket, right pocket, wrist, belt and upper arm.

PAMAP2 [13]: PAMAP2 is a HAR dataset that provides data from three accelerometers for five activities with nine users.

eRing [14]: eRing is a smart health dataset that uses a ring to capture data along four dimensions. The eRing dataset allows us to test the efficacy of AIM in gesture recognition settings.

SelfRegulationSCP1 (SR-SCP1) [15]: SR-SCP1 is a health monitoring dataset that includes EEG data from six channels. The data from EEG sensors is used to develop a control system to drive spelling devices for completely paralyzed patients.

3) Evaluation Metrics: We employ accuracy, memory, and energy consumption as evaluation metrics. Accuracy is used as a metric because accuracy is of utmost importance in health applications. Similarly, memory and energy are important for wearable devices due to resource constraints.

4) Classifier Representation: We use a 1-D convolutional neural network (CNN) as the classifier for all datasets. Specifically, we use a 1-D CNN with one conv. and max-pooling layers, and two fully connected layers with the ReLU activation and dropout value of 20%. We use the Adam optimizer [24] over 20 epochs for both standard and robust training.

B. Baseline Methods for Comparison

The proposed data recovery approach is compared against baseline approaches described below.

GAIN [12]: GAIN is a generative approach that recovers missing data as a function of the observed data. GAIN trains a deep neural network that takes the observed data and a mask specifying the missing time instances as input. The output of the generator is a data matrix that consists of imputed values.

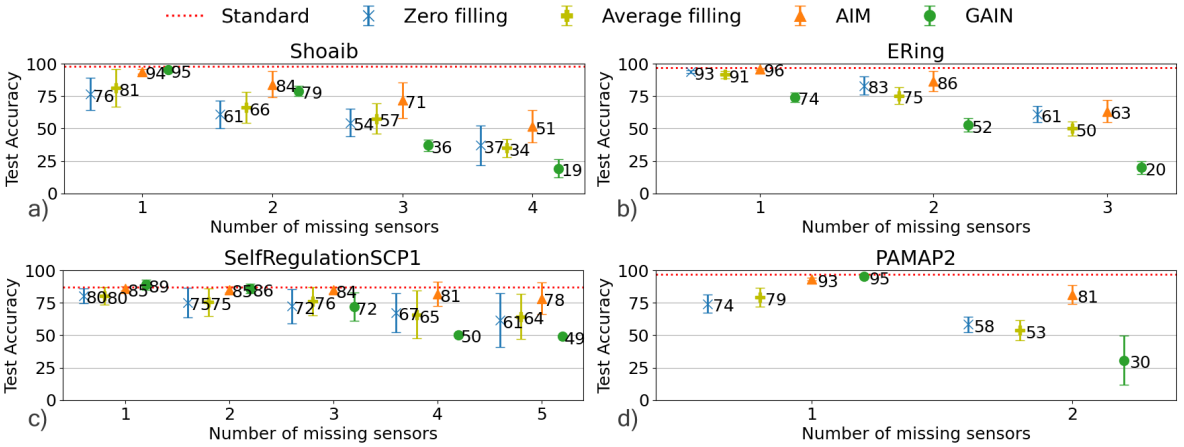

Fig. 3: Accuracy (Mean and standard deviation) of the robust-trained ML classifier via different imputation methods on all combinations of missing sensors.

One of the disadvantages of the GAIN approach is the high memory requirement for storage of the generator parameters and energy overhead for each imputation. Moreover, generative models for time-series data are challenging to train [25], which can affect their accuracy in complex tasks.

Zero Filling: In the absence of any data recovery algorithm, missing data will typically be filled with zeros. Therefore, we use it as one of the baselines for comparison. Filling missing values with previously observed data is not feasible since we assume the sensor is missing for the entire experiment.

Average Filling: Another realistic alternative for zero-filling is to fill the missing data with pre-determined values that represent the average case over the training data with no missingness. These values are determined by averaging sensor data across all training time-series signals.

C. Application Accuracy with Imputed Data

We start the experimental evaluation by analyzing the accuracy of the health applications under different missing data scenarios. For each dataset, we first train a classifier to perform the application tasks. Once the classifier is trained, we use it with the proposed search algorithm to find likely patterns of sensor data when one or more sensors are missing.

Figure 3 shows the comparison of accuracy for all four datasets. Each point on the figure shows the mean and standard deviation of the accuracy over all possible combinations of missing sensors. For example, in case of two missing sensors in the Shoaib dataset, we obtain the average and standard deviation over $\binom{5}{2}$ combinations of possible scenarios. We see that missing data with zero-filling or average-filling settings have a significant drop in accuracy. For example, for both HAR datasets, a single missing sensor results in more than 20% drop in average accuracy. In contrast, using the same classifier and missing data cases, AIM is able to efficiently recover the classification performance. Even when data from the entire window is missing, AIM is able to produce an average performance within 5% of the original accuracy for all datasets. AIM also succeeds in improving average performance of the classifiers on HAR datasets by 15% in the highly-unlikely case where almost all sensors are missing. Additionally, the confusion matrices in Figure 4 show that AIM

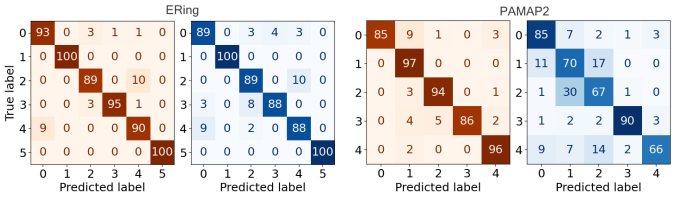

Fig. 4: Confusion matrix normalized over the true labels of the deployed classifier on ERing and PAMAP2 datasets using AIM (red) imputation methods in the event of a single missing sensor. The same performance using zero-filling (blue) is provided for reference.

improves the classification accuracy over different classes with equal importance in spite using a single imputation pattern.

Compared to the imputation provided by the baseline GAIN approach, AIM produces better results for most of the cases. Notably, GAIN fails to recover the original accuracy when more than one sensor is missing. GAIN has lower accuracy than zero-filling in some cases because GAIN is unable to follow real data accurately and incurs higher error. For PAMAP2, the average performance is reduced from 58% to 30% for the GAIN algorithm. In summary, the AIM approach is able to efficiently recover the data with low overhead, while the baseline GAIN approach is unable to recover the accuracy for more than one sensor missing and has a higher overhead.

D. Accuracy Improvement with Robust Classifiers

The AIM algorithm generates a pattern $\mathcal{I}_{\{j\}}$ to preserve the accuracy of the classifier in the case of $\{j\}$ missing input channels. Ideally, the generated pattern requires small adjustments to fit every input X to maintain $F_\theta(X) \approx F_\theta(\mathcal{I}_{\{j\}})$. To account for these adjustments, we use robust training for the ML classifier to overcome small errors in imputed data. Robust classifiers are also important because any health application must be able to handle small variations in data either due to natural disturbances or imputation. To this end, we compare the accuracy of the proposed robust classifiers with the standard classifier in Table I. Indeed, the table shows that robust training overcomes errors due to small imputation deviations by improving the average accuracy for majority of cases while reducing standard deviation. For example, SR-SCP1 has an increase of 6% in recovered accuracy with a

TABLE I: Classification accuracy of the imputed data of k missing sensors generated by AIM using different standard and robust training protocols. Table entries show mean with standard deviation in parantheses.

Dataset	k	Standard	Robust	Dataset	k	Standard	Robust
Shoaib	1	91 (2)	94 (1)	SR-SCP1	1	79 (8)	85 (1)
	2	80 (5)	84 (9)		2	79 (8)	85 (2)
	3	69 (13)	71 (13)		3	79 (8)	84 (2)
	4	50 (12)	51 (12)		4	81 (9)	81 (9)
ERing	1	95 (1)	96 (0)		5	73 (5)	78 (12)
	2	84 (8)	86 (7)	PAMAP2	1	92 (2)	93 (0)
	3	60 (2)	63 (8)		2	75 (3)	77 (11)

standard deviation of 1%. Overall, robust training is able to provide higher accuracy while reducing standard deviation.

E. Implementation Overhead

One of the primary advantages of AIM is low memory and energy overhead. Table II shows the memory overhead for AIM and GAIN, respectively. GAIN incurs high memory overhead to store the parameters of the generator network. In contrast, AIM has less than 1 MB memory overhead. AIM memory requirements are minimal even when the wearable device includes multiple health applications. The memory overhead for AIM can be further reduced by loading *only* the required imputation setting on detecting missing data.

Next, Figure 5(a) compares energy consumption of GAIN and AIM for all datasets. The energy consumption is obtained using power sensors on the Odroid-XU3 board. We see that AIM consumes less than 10 mJ per imputation while GAIN has significantly higher energy consumption. For instance, energy consumption for the SR-SCP1 dataset is close to 1 J for each imputation. Similarly, Figure 5(b) shows percentage energy savings achieved by AIM when compared to GAIN. The energy savings are close to 98% for all datasets except eRing. The eRing dataset has lower energy savings of about 74% since it has lower computation requirements for both GAIN and AIM, resulting in lower energy savings. In summary, the AIM approach provides superior performance over prior approaches while incurring significantly lower overhead.

TABLE II: Summary of memory overhead of AIM and GAIN approaches

Dataset	AIM Memory (MB)	GAIN Memory (MB)
Shoaib	0.180	25
PAMAP2	0.055	60
eRing	0.007	0.19
SR-SCP1	0.667	81

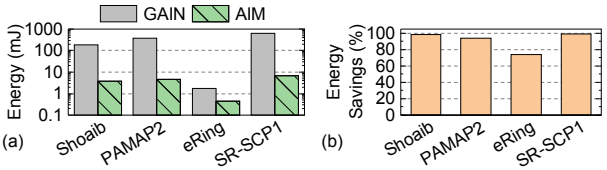

Fig. 5: a) Comparison of energy consumption for GAIN and AIM approach. The y-axis is shown in log scale to represent the large range of values. b) Energy savings achieved by AIM when compared to GAIN.

VI. Conclusion

Wearable devices are transforming a number of high-impact applications. However, they may suffer loss in the quality of service due to one or more sensors being unavailable at runtime. This paper presented a novel search-based algorithm that obtains most likely imputation patterns of sensor data for each missing data scenario via *offline* analytics. Experiments on four diverse wearable sensor based time-series benchmarks showed that the proposed approach is able to maintain accuracy within 5% of the ideal accuracy when the number of missing sensors is less than two, with negligible runtime overhead.

References

[1] A. Mosenia *et al.*, "Wearable Medical Sensor-Based System Design: A Survey," *IEEE TMSCS.*, vol. 3, no. 2, pp. 124–138, 2017.

[2] A. Limaye and T. Adegbija, "HERMIT: A Benchmark Suite for the Internet of Medical Things," *IEEE IoT J.*, vol. 5, no. 5, 2018.

[3] A. J. Espay *et al.*, "Technology in Parkinson's Disease: Challenges and Opportunities," *Movt. Disorders*, vol. 31, no. 9, pp. 1272–1282, 2016.

[4] P. Zappi *et al.*, "Activity Recognition from On-Body Sensors by Classifier Fusion: Sensor Scalability and Robustness," in *Proc. Int. Conf. on Intell. Sensors, Sensor Netw. and Info.*, 2007, pp. 281–286.

[5] H. Kim *et al.*, "Collaborative Classification for Daily Activity Recognition with a Smartwatch," in *Proc. SMC*, 2016, pp. 003 707–003 712.

[6] S. Liu *et al.*, "Handling Missing Sensors in Topology-Aware IoT Applications with Gated Graph Neural Network," *Proc. IMWUT*, vol. 4, no. 3, pp. 1–31, 2020.

[7] K. Kunze and P. Lukowicz, "Sensor Placement Variations in Wearable Activity Recognition," *IEEE Perv. Comput.*, vol. 13, no. 4, 2014.

[8] M. Shoaib *et al.*, "Fusion of Smartphone Motion Sensors for Physical Activity Recognition," *Sensors*, vol. 14, no. 6, pp. 10 146–10 176, 2014.

[9] I. M. Pires *et al.*, "Improving Human Activity Monitoring by Imputation of Missing Sensory Data: Experimental Study," *Future Internet*, vol. 12, no. 9, p. 155, 2020.

[10] T. De Waal, J. Pannekoek, and S. Scholtus, *Handbook of Statistical Data Editing and Imputation*. John Wiley & Sons, 2011, vol. 563.

[11] T. Hossain and S. Inoue, "A Comparative Study on Missing Data Handling Using Machine Learning for Human Activity Recognition," in *Proc. ICIEV and icIVPR*, 2019, pp. 124–129.

[12] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing Data Imputation Using Generative Adversarial Nets," in *ICML*, 2018, pp. 5689–5698.

[13] A. Reiss and D. Stricker, "Introducing a New Benchmarked Dataset for Activity Monitoring," in *ISWC*, 2012, pp. 108–109.

[14] M. Wilhelm *et al.*, "eRing: Multiple Finger Gesture Recognition with One Ring Using an Electric Field," in *Proc. Int. Work. on Sensor-based Activity Recognition and Interaction*, 2015, pp. 1–6.

[15] N. Birbaumer *et al.*, "A Brain-Controlled Spelling Device for the Completely Paralyzed," *Nature*, pp. 297–298, 2001.

[16] Hardkernel. (2014) Odroid-xu3. https://www.hardkernel.com/shop/odroid-xu3/ Accessed 11/20/2020.

[17] 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.

[18] Z. Guo *et al.*, "A Data Imputation Method for Multivariate Time Series Based on Generative Adversarial Network," *Neurocomputing*, vol. 360, pp. 185–197, 2019.

[19] D. Hussein, A. Jain, and G. Bhat, "Robust Human Activity Recognition Using Generative Adversarial Imputation Networks," in *Proc. DATE*, 2022, pp. 84–87.

[20] S. Talukder *et al.*, "Deep Neural Imputation: A Framework for Recovering Incomplete Brain Recordings," *arXiv:2206.08094*, 2022.

[21] T. Belkhouja and J. R. Doppa, "Analyzing Deep Learning for Time-Series Data Through Adversarial Lens in Mobile and IoT Applications," *IEEE TCAD*, vol. 39, no. 11, pp. 3190–3201, 2020.

[22] ——, "Adversarial Framework with Certified Robustness for Time-Series Domain via Statistical Features," *JAIR*, 2022.

[23] D. Hussein *et al.*, "Reliable Machine Learning for Wearable Activity Monitoring: Novel Algorithms and Theoretical Guarantees," in *Proc. ICCAD*, 2022, pp. 1–9.

[24] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *The Int. Conf. on Learning Representations (Poster)*, 2015.

[25] E. Brophy, Z. Wang, Q. She, and T. Ward, "Generative Adversarial Networks in Time Series: A Survey and Taxonomy," *arXiv preprint arXiv:2107.11098*, 2021.