A Survey of Embedded Machine Learning for Smart and Sustainable Healthcare Applications

Sizhe An, Yigit Tuncel, Toygun Basaklar, and Umit Y. Ogras

1 Introduction

Embedded machine learning has recently drawn significant attention due to the fast development of machine learning (ML) and embedded devices. It is an application of artificial intelligence (AI) to make decisions or predictions from the existing data at the edge without explicit programming. The success of embedded ML heavily relies on the recent improvement of computation power since ML algorithms are highly data-intensive. Machines need to launch complex linear algebraic computations, such as matrix and vector operations, to learn the non-trivial relationship between the inputs and outputs. To date, computing clusters using multiple high-frequency central processing units (CPU), graphics processing unit (GPU), and tensor processing unit (TPU) [79] are the most widely used resources to perform such operations. However, when the users want to enjoy the convenience of ML without transmitting their local data, computing clusters are certainly not a good choice due to the prices and large form factor.

An embedded device refers to a small computer system—a combination of computer processors, memory, and input/output devices [91]. Nowadays, people have access to multiple personal devices, such as smartphones, smartwatches, and autonomous cars. Embedded devices such as Raspberry Pi [66], Nvidia Jetson [52], and Arduino [11] are powerful yet affordable due to the recent emergence of hardware. For example, an Nvidia Jetson Nano developer kit [52] with 128-core 4GB memory GPU that can run most ML algorithms only costs less than $100. Embedded machine learning enables the deployment of ML algorithms on edge devices rather than the powerful computational cluster. It allows the end users to

S. An · Y. Tuncel · T. Basaklar · U. Y. Ogras (✉)
University of Wisconsin-Madison, Madison, WI, USA
e-mail: sizhe.an@wisc.edu; tuncel@wisc.edu; basaklar@wisc.edu; uogras@wisc.edu

S. Pasricha, M. Shafique (eds.), *Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing*, https://doi.org/10.1007/978-3-031-40677-5_6

perform machine learning directly on devices used in the field, thus leading to numerous novel applications.

Concurrent advances in machine learning and low-power computing areas pave the way for high-impact applications. These applications can attract social attention, promote industrial progress, and improve the quality of life. For example, computer vision (CV) and natural language processing (NLP) are major technical areas that widely apply machine learning since they are attached closely to the industry and commercial market. Specifically, computer vision helps machines understand images and videos, thus generating meaningful information and making decisions. Example applications include image classification, object detection, object tracking, and instance segmentation. The market for computer vision is expected to reach USD 48.6 billion by 2022 [38]. NLP refers to the technology that gives computers to understand text and language similar to human beings [37]. Application of NLP includes speech recognition, sentiment analysis, machine translation, and text summarization. Another popular machine learning application area is recommendation system (RS). Recommendation systems aim to recommend things to the users based on their previous interests and other factors. These systems predict the most likely content that will interest users. Many tech companies such as Google, Amazon, and Netflix rely on recommendation systems to enhance their customers' engagement.

As a particular subset of the previously mentioned ML applications trained solely with big data, embedded machine learning supports obtaining and processing new data locally. Embedded devices are usually equipped with different sensors to measure motion, biopotentials, and temperature. Embedded ML can directly use data from these sensors, thus enabling numerous novel applications. Target applications include smart healthcare, autonomous driving, professional sports, and power/energy management [23, 54, 87]. The rest of the chapter will first overview embedded machine learning frameworks and then offer examples of specific applications using embedded machine learning. This chapter focuses mainly on embedded machine learning applications with data obtained on-device. The concept of embedded machine learning and tinyML will be coherently interspersed and interact with the applications. Finally, we also introduce energy management as a service application since the deployment of ML applications on edge devices is limited by battery capacity.

2 Overview of Embedded Machine Learning Frameworks

Embedded machine learning frameworks typically consist of model training, model compression, and model inference, as illustrated in Fig. 1. Model training is the process of learning the non-trivial patterns or relationships between the inputs and outputs through an intensive search that includes trial and error. It usually needs a vast amount of data points to train to learn the hidden complex relationship between the inputs and outputs instead of memorizing from the existing data. Thus, model

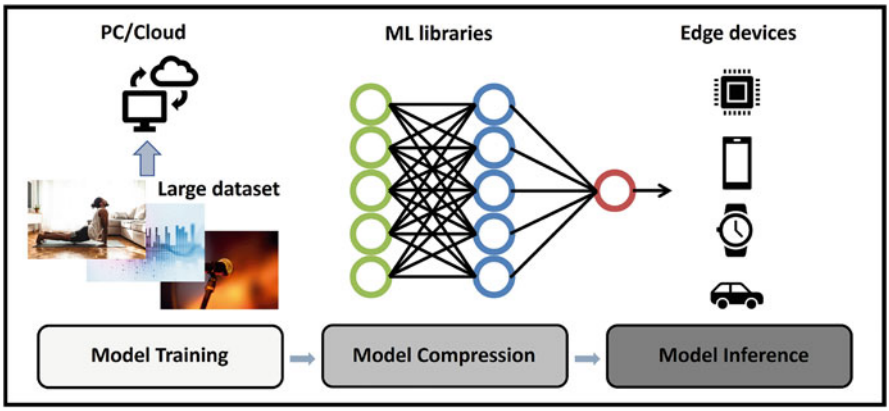

Fig. 1 Overview of embedding machine learning framework

training is usually performed in powerful computation processors, such as cloud servers, workstations, or personal computers (PC), as shown in Fig. 1.

The success of ML frameworks relies heavily on the improvement of computation power since ML algorithms need an intensive amount of data to train. However, users of ML applications do not participate in the model training part. Instead, they run the pre-trained ML models on their devices for inference in different applications. Therefore, it is crucial to ensure that the ML algorithms targeting embedded applications can run on edge devices with limited computational power and memory. To this end, popular ML libraries such as PyTorch [60] and TensorFlow [1] have been making significant efforts in compressing the size and accelerating the inference time of the model on embedded devices. For example, PyTorch Mobile [61] and TFLite [80] are the corresponding lightweight version of PyTorch and TensorFlow. Many recent studies take advantage of these lightweight ML frameworks that target embedded ML [26, 94, 95]. The lightweight version of TensorFlow, TFLite, can reduce the model size up to 75% with a minimal accuracy loss. For instance, a ResNet101 model, after optimized by TFLite, is only 44.9 MB, compared to 178.3 MB on TensorFlow with only 0.2% accuracy loss. Consequently, the tinyML concept is introduced with the fast development of lightweight ML libraries. TinyML refers to ML capable of performing on-device sensor data analytics at ultra-low power, thus enabling a variety of always-on applications and targeting battery-operated devices [82]. The middle part of Fig. 1 shows that model compression is bridging the model training and model inference using ML libraries.

Model inference refers to the process of inferring the most likely output of given inputs from the previously learned model. Hence, it does not require intensive computational power and can be easily deployed on edge devices. In the embedded machine learning flows, the model inference is the central part that runs on users' edge devices. For instance, users can access ML applications such as tracking

their vital signs using a smartphone or smartwatch [9] and setting up self-driving functions in their car. A recent study [94] designed and implemented convolution neural networks (CNNs) on smartphones using TFLite to estimate human activities in real time. Similarly, real-time human pose estimations for a single person and multi-person on mobile devices are proposed in [26] and [95], respectively. The right part of Fig. 1 shows that users can use the compressed ML algorithm for their customized applications using their edge devices.

3 Embedded Machine Learning Applications for Healthcare

The aging population has been becoming a serious concern all over the world. The consequent rise in health-related issues has drawn significant research attention from the industry and academic community. Technology companies have continuously increased their R&D expenditure for wearable devices that can be used for mobile activity and health monitoring. For instance, Apple utilizes its popular consumer-facing products, such as Apple Watch, to provide health-related features accessible on its watch to bridge wearables and clinical tools used in medical research [9]. In 2021, Google acquired the wearable giant Fitbit (smartwatch) to participate in the AI-enabled healthcare race among top-pitch technical companies [32]. Embedded machine learning enables various healthcare-related applications by feeding multi-modal sensing data obtained from humans into machine learning algorithms on devices.

Remote activity and healthy monitoring applications can provide valuable insights [12]. Hence, they can improve the quality of life in many healthcare applications, including but not limited to human activity recognition [4, 19], gait monitoring [5, 46], human pose estimation [2, 3, 92], and freezing of gait (FoG) [24, 55, 69]. For example, advanced ML algorithms can locally analyze the motion and physiological data from wearable sensors. This capability can enable many real-time applications such as irregular rhythm notification, early warning signs, and fall detection. These applications are also the first step toward diagnosis, prognosis, and rehabilitation of movement disorders similar to Parkinson's disease (PD) and stroke. The rest of this section discusses three illustrative examples as cases studies and summarizes the recent work in those areas.

3.1 Freezing-of-Gait Identification in PD Patients

Parkinson's disease (PD) is one of the most common age-related neurodegenerative diseases. It causes muscular rigidity, tremor, bradykinesia, slowness in movement, and postural instability [27, 45, 49, 83]. More than 50% of the PD patients develop freezing of gait (FoG) [44] in their advanced stages of the disease. Freezing of gait is a brief absence of the ability to walk despite the intention of moving the feet [59].

FoG episodes may include trembling of knees, short shuffling steps, or complete akinesia [71] and overall increase the risk of falling and deteriorate the patient's quality of life.

Clinical studies suggest that external stimuli, such as auditory, visual, or tactile cues, help patients to exit the FoG state and resume walking [59]. FoG identification is a challenging task since FoG episodes are rare events. Furthermore, clinical settings cannot provide the actual frequency of occurrence of FoG episodes. For example, FoG episodes mainly occur during turning, walking through doorways, or dual tasking, and 90% of them last less than 20 s [44]. Specific experimental sessions are designed to simulate the daily-life activities in clinical settings, such as walking and turning while dual tasking [44]. However, this practice is a challenging proposition due to fundamental limitations. On the one hand, the duration of these sessions in a clinical setting and the number of visits per patient are limited. On the other hand, the frequency of FoG occurrence and the duration of FoG episodes are short. Hence, the probability of observing FoG episodes during these simulations can be small.

FoG-related problems can be avoided by systems that can identify FoG episodes and provide an appropriate cueing mechanism such as audio. FoG identification can be divided into two subclasses: FoG detection and FoG prediction. FoG detection implies classifying FoG episodes *after patients start to experience them*. It has been heavily studied for over a decade. More than 50 studies related to FoG detection have been published by late 2019 [58]. In contrast, FoG prediction implies classifying FoG episodes before the occurrence of an FoG episode. The main goal of the FoG prediction studies is to *predict potential FoG episodes and prevent their onset* by providing preemptive cueing. Despite this promising potential, there are very few studies related to FoG prediction compared to FoG detection [58, 59, 73].

Both FoG detection and prediction studies use various wearable sensors placed on different parts of the patient's body. These sensors collect motion data using inertial measurement units, plantar pressure systems, and electromyography [44, 59]. Various machine learning approaches are utilized for FoG identification, including random forests, support vector machines (SVMs), nearest neighbor algorithms, and deep neural networks (DNNs) [57]. Classification algorithms, such as decision trees, SVMs, and *k*-nearest neighbor, require handcrafted spectral and statistical features extracted from the motion data of the PD patients, which needs substantial domain knowledge [57]. However, FoG identification approaches should require minimal preprocessing and manual effort to facilitate easy deployment on edge-AI devices. Approaches that employ DNNs do not require domain knowledge. They can directly utilize raw sensor data to identify FoG episodes. For example, convolutional neural networks (CNNs) are adopted widely among with long short-term memory (LSTM) networks to detect and predict FoG episodes [13, 53, 57, 73, 83]. Table 1 summarizes the recent FoG identification studies published since 2018. None of the above studies have embedded ML framework in an edge-AI device. These approaches require powerful computing resources that are hard to integrate on an edge device. Therefore, there is a critical need for lightweight FoG identification approaches

Table 1 A summary of recent FoG identification studies

	Sensor type and location	FoG identification approach
Sama et al. [69]	IMU placed on the waist	k-NN, random forest, logistic regression, Naïve Bayes, multilayer perceptron (MLP), SVM
Camps et al. [24]	IMU placed on the waist	CNN
Oung et al. [55]	3 accelerometers placed on the left shank, left thigh, and lower back	Probabilistic NN, SVM
Li et al. [41]	Accelerometer placed on the lower back	Mini-batch k-means clustering
Mikos et al. [47]	2 accelerometers placed on each ankle	MLP
Rad et al. [63]	3 accelerometers placed on the left shank, left thigh, and lower back	Denoising autoencoder
Handojoseno et al. [33]	EEG electrodes placed on the head	DNN
Torvi et al. [83]	3 accelerometers placed on the left shank, left thigh, and lower back	LSTM
El-Attar et al. [29]	Accelerometer placed on the left shank	DNN
Naghavi and Wade [49]	3 accelerometers placed on the left shank, left thigh, and lower back	Statistical analysis based on Kruskal–Wallis test
Naghavi et al. [50]	2 accelerometers placed on each ankle	k-NN, SVM, decision tree, MLP along with classifier bagging and synthetic minority over-sampling methods
Arami et al. [10]	3 accelerometers placed on the left shank, left thigh, and lower back	SVM
Demrozi et al. [28]	3 accelerometers placed on the left shank, left thigh, and lower back	k-NN
Reches et al. [67]	3 accelerometers placed on the left ankle, right ankle, and lower back	SVM
Shi et al. [75]	3 accelerometers placed on the left ankle, right ankle, and neck	CNN
Li et al. [42]	3 accelerometers placed on the left shank, left thigh, and lower back	CNN + LSTM
Sigcha et al. [77]	IMU placed on the waist	CNN + LSTM
Mancini et al. [44]	8 IMUs placed on the shins, feet, wrists, sternum, and lower back	Correlation and thresholding
Bikias et al. [20]	IMU placed on the wrist	CNN
Borzi et al. [21]	2 IMUs placed on the shins	k-NN, SVM, linear discriminant analysis, logistic regression

that leverage the wearable sensor data with minimal preprocessing of the data and activate an appropriate cueing mechanism locally on the edge device.

3.2 Human Activity Recognition

There has been growing interest in human activity recognition (HAR) due to its health monitoring and patient rehabilitation applications [4, 65, 76, 90]. Inertial measurement units (IMUs) are used to capture the body and joint movements to estimate or predict human activity. The recognized activities with time stamps are valuable insights for health monitoring and rehabilitation. The prevalence of low-cost motion sensors and embedded machine learning algorithms make it possible to perform human activity recognition on-device [18].

3.2.1 Processing Pipeline

The majority of HAR methods employ smartphones due to their popularity and easy access to integrated accelerometers and gyroscope sensors [6, 76, 90]. More recent work started using wearable devices for this purpose due to significant power consumption and form-factor benefits. A typical HAR framework consists of data preprocessing, feature extraction, and classifying algorithms, as shown in Fig. 2. Inertial measurement units, which typically include accelerometers and gyroscopes, are attached to different human body parts. They usually provide three-axis acceleration and angular velocity. If the activities are simple enough, a significant number of approaches use only the acceleration data since the gyroscope sensors have relatively larger power consumption [97]. The first step is to preprocess the raw sensor data to reduce measurement noise and construct activity windows. Then, the data in each activity window are used to produce features, such as the body acceleration and signal statistics (e.g., min, max, mean). Finally, these features are used by ML algorithms to recognize activities, such as standing, sitting, lying, walking, and jogging.

A range of methods is used during the preprocessing step. The most common preprocessing techniques include downsampling, low-pass filtering, and segmentation. The inertial sensors often sample at a high sampling rate (>50 Hz). However, most human activities are only at a few Hz range [18], making the sampled data redundant. Median and mean filters are two mainstream filters used for downsampling. Similarly, the sensor data, especially accelerometer data, are typically noisy. Therefore, most preprocessing techniques incorporate low-pass filtering and smoothing. Most ML algorithms require the inputs to have a fixed length. Therefore, preprocessing steps typically involve segmentation algorithms that divide the data into possibly overlapping windows. For example, segmentation algorithms can divide a long period (in the order of hours) of time series data into multiple fixed-length (e.g., one–ten seconds) windows. If the number of data samples in each window is uniform, the ML algorithms, such as DNNs, can conveniently process them.

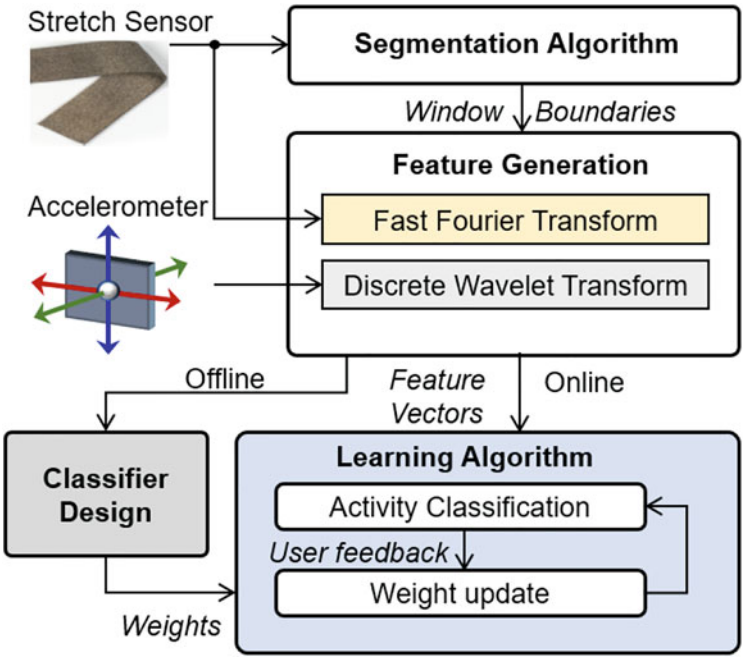

Fig. 2 Overview of human activity recognition. The figure is partially modified from [18]

3.2.2 Commonly Used ML Algorithms

Most of the HAR techniques employ supervised learning algorithms. Examples of supervised learning algorithms suitable for HAR include support vector machine (SVM), random forests, decision trees, k-nearest neighbors (k-NN), and neural network (NN). These algorithms take the labeled data to train the classifier, as outlined below:

- *Support vector machine*: SVM techniques try to find a hyperplane in high-dimensional space that separates two output classes [34]. If they cannot find the separating hyperplane in a lower dimension, it keeps mapping the data into higher dimensions until the separating hyperplane is found. HAR problem is essentially a multi-class classifying problem. Multiple classifiers need to be performed to apply SVM to HAR for differentiating more than two output classes since SVM is a binary classifier.
- *Random forests and decision trees*: Decision tree classifiers are commonly used for classification problems since it is intuitive and explainable. They use a series of rules to make decisions, just like how humans make decisions [34]. Decision trees take the dataset features to create binary questions and continually separate the dataset until all data samples are isolated to different classes. An ensemble

of tree-structured classifiers is employed for random forests. The most predicted classes among all trees are chosen as the final predicted class.

- *k-Nearest Neighbors*: This approach is one of the most traditional and popular techniques in classifying problems [34]. K-NN first computes k-nearest neighbors in the training set. Then, it chooses the most common class among k neighbors. That class is then the final estimated class. K-NN technique requires all training data to be stored locally. Many of the HAR techniques used k-NN for the offline training [70].
- *Neural networks*: Neural networks are widely used in classification and regression problems. Multilayer perceptron (MLP) classifier is one of the primary neural networks used for human activity recognition. It consists of three layers at least: one input layer, one hidden layer, and one output layer. Each layer of the MLP has multiple neurons/nodes with non-linear activation functions. In the HAR context, the number of hidden neurons in the hidden layers is crucial for obtaining good accuracy while retaining low computational complexity. The number of neurons in the output layer corresponds to the output classes. Convolution neural networks (CNNs) are popular for processing 2D images. Convolutional kernels can convolve with the input image layers by layers to extract the helpful feature maps instead of choosing the features manually in other techniques [93]. Researchers recently applied CNN in HAR by reshaping the 1D HAR features to a 2D feature map input [16].

3.2.3 Offline vs. Online Learning

Currently, there are two mainstream HAR training paradigms: offline training and online training. The training is performed on dedicated computing resources such as power CPU and GPU for offline training. Machine learning algorithms require a large amount of data to capture different patterns' behavior and learn how to classify them. Thus, dedicated computing resources are used since they can process a vast amount of data. The offline pre-trained models then are deployed to edge devices, for example, smartphones and smart wearables, to perform the inference for new users. Offline training is the fundamentals of all supervised training algorithms. The downside of this approach is that the performance of inferring on new users that have not appeared in the training data is inevitably worse than inferring on the trained users. Online training tackles this issue by continually training with new user data on the edge devices. The pre-trained models are also deployed to edge devices, but the new users' data obtained on the field are fed into the machine learning algorithm running on the edge device to train online. Since the amount of data is small compared to the offline training data, the edge devices can train the models in real time. For HAR on-device, the neural network is the most popular method since it supports online training [18].

3.3 Human Pose Estimation

Human pose estimation aims to detect and track key joints, such as wrists, elbows, and knees. It has rapidly growing applications areas, including rehabilitation, professional sports, and autonomous driving [23, 54, 87]. For instance, one of the leading causes of autonomous car accidents is "robotic" driving, where the self-driver makes a legal but unexpected stop and causes other drivers to crash into it [54]. Real-time human pose estimation can help computers understand and predict human states, thus leading to more natural driving. Likewise, remote rehabilitation applications, which are currently not feasible, can be enabled by human pose estimation.

Human pose estimation can be performed by processing image, video, LiDAR (light detection and ranging), inertial sensors (IMUs), or mmWave radar data. RGB image and video frames are the most common input types since they offer accurate real-world representations with true color. However, the RGB frame quality depends heavily on the environmental setting, such as light conditions and visibility. Alternatively, the LiDAR point cloud obtained by laser scanning overcomes these challenges. However, it has high-cost and significant processing requirements, making them unsuitable for indoor applications such as rehabilitation. mmWave radar can generate high-resolution 3D point clouds while maintaining low-cost, price, and power advantages. Inertial sensors (IMUs) can also reconstruct human pose using the sensing accelerations and gyroscopes [88, 89]. This section discusses these three broad approaches and recent literature.

3.3.1 Human Pose Estimation Using RGB Camera

In the computer vision field, human pose estimation has drawn attention after a seminal study in 2005 [64]. This study presents a framework to detect ten distinct body parts using rectangular templates from RGB images. He et al. [35] propose Mask R-CNN, which can reconstruct skeleton from RGB images using K masks by leveraging ResNet neural network architecture. It first detects K different key points and then connects them. Mask R-CNN has become popular due to its fast processing time and accurate estimation. Similarly, Cao et al. proposed OpenPose [25], a real-time human pose estimation technique that can detect human body, face, and foot key points together for the first time. OpenPose has become one of the popular benchmarks due to its decent performance and the easy-to-use open-source package. Besides the RGB video-based approach, Microsoft Kinect and Kinect V2 [74] provide depth cameras to extract the human joints representation. Both Kinect and Kinect V2 use an RGB camera and a depth sensor consisting of an infra-red camera and projector as sensing units to capture the information. The Kinect family has become one of the popular methods to obtain the ground truth label for training due to its convenience, low cost, and nice performance [7, 72, 96].

From an RGB image containing a person, the human pose estimation model typically consists of the cropping bounding box, extracting features, and predicting the joint coordinates. The first step is to crop the bounding box containing the human. The area of a person can be only 1/5 of the image or even smaller. For the human pose estimation task, the region of interest (ROI) is only the area related to humans. Hence, an efficient human bounding box detection algorithm is crucial. After obtaining the human bounding box, the next step is to extract useful features from the area. Most of the techniques use deep CNNs since many of them have been proved to be suitable feature extractors in the image processing field [35]. Finally, the model needs to output the human joints coordinates. In estimating the joint coordinates, heatmap-based and regression-based are two mainstream methods.

Heatmap-based models learn each joint point's position through the Gaussian distribution graphs. The method first renders Gaussian probability distribution heatmaps for every joint point and then applies argmax or soft-argmax operation to the heatmaps, thus obtaining the final estimation results. Since the maximum value of every heatmap corresponds to a joint's coordinates, the resolution of the output heatmap needs to be relatively high (64×64 usually). Thus, this method's computation and memory overhead are high since multiple high-resolution Gaussian heatmaps need to be rendered (one output heatmap for each joint).

Regression-based models represent another alternative that is simpler and more intuitive. It directly learns the joints' coordinates values using L1 or L2 loss. Since the regression-based method does not require rendering the heatmap and maintaining the high resolution, the output feature map can be small compared to the heatmap-based model. Thus, the computation and memory requirements of the regression-based model are significantly lower than the heatmap-based model. For instance, using Resnet-50, the floating-point operation per second (FLOPs) of the regression-based model is 1/20000 of the heatmap-based model [43]. This result shows that the regression model is friendly to the edge devices. Regression-based methods are widely applied in industry [43] since it is computationally efficient and straightforward. However, the heatmap-based method is generally more robust for occlusion and blur. In addition, the heatmap-based model has better explainability than the regression-based model. Recently, researchers have started to combine two methods to keep advantages of both of them [43].

3.3.2 Human Pose Estimation Using mmWave Radar

The human pose can also be reconstructed from mmWave signals. Compared to the RGB image source, mmWave signals preserve user privacy well since the mmWave signal does not reveal salient and rich information such as true-color images. At the same time, the sparse input source makes human pose estimation a more challenging task. Almost all mmWave human pose estimation methods use a regression-based model. In 2018, researchers proposed RF-Pose3D [96], a technique that reconstructs up to 14 body parts, including the head, neck, shoulders, elbows, wrists, hip, knees, and feet. This work first uses 12 camera nodes to record RGB-based video and

then obtain label key points from OpenPose. At the same time, radar signals at a few GHz are used to generate the RF heatmap. They then train a region proposal network (RPN) to zoom in on RF data and a CNN with ResNet architecture to extract the 3D skeleton from the region of interest. For keypoint localization, the average errors in the x, y, z axes are 4.2, 4.0, and 4.9 cm, respectively.

Besides being limited to 14 joints, this work does not leverage the mmWave radar's ability to obtain a high-quality point cloud. Thus, it requires a much more complex NN architecture with high computation cost. Moreover, multiple cameras and bulky radar signal generating systems hinder the practicality of the approach. The most recent mmWave radar-based pose estimation techniques use point cloud representation from the commercial radar device Texas Instrument (TI) xWR1x43 [81]. Sengupta et al. [72] propose mmPose, a human pose estimation technique that constructs the skeleton by using mmWave point cloud and a forked-CNN architecture. They use two radar devices and sum up the point values in the feature map level to overcome the sparse representation of the point cloud. An et al. [3] present a meta-learning and frame aggregation framework to help mmWave-based human pose estimation model converge faster for unseen scenarios. Xue et al. [92] propose the mmMesh technique to construct human mesh using mmWave point cloud.

Finally, another recent study proposes a mmWave-based assistive rehabilitation system (MARS) [2] using human pose estimation. It sorts the mmWave point cloud and performs matrix transformations before feeding them to a CNN model. MARS can reconstruct up to 19 human joints and human skeleton in 3D space using mmWave radar without raising privacy concerns and requiring strict lighting settings. Moreover, MARS provides the users with 19 joints velocity estimations, four critical angle estimations, and ten commonly used rehabilitation posture correction feedback. It incorporates point cloud preprocessing, a CNN that outputs joint positions, and rehabilitation movement feedback to the user. It first maps the 5D time series mmWave point cloud to a 5-channel feature map and then outputs 3D joint positions. It finally provides joint velocity, angle estimations, and posture correction feedback. The overview of MARS is shown in Fig. 3. An example of human pose estimation using mmWave radar point cloud is shown in Fig. 4.

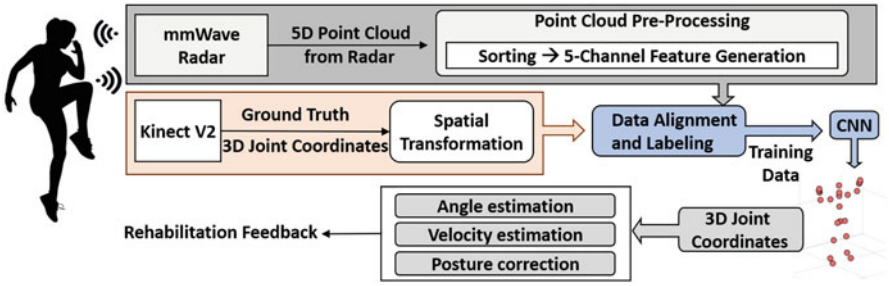

Fig. 3 Overview of human pose estimation using mmWave point cloud and its downstream healthcare-related tasks. The figure is partially modified from [2]

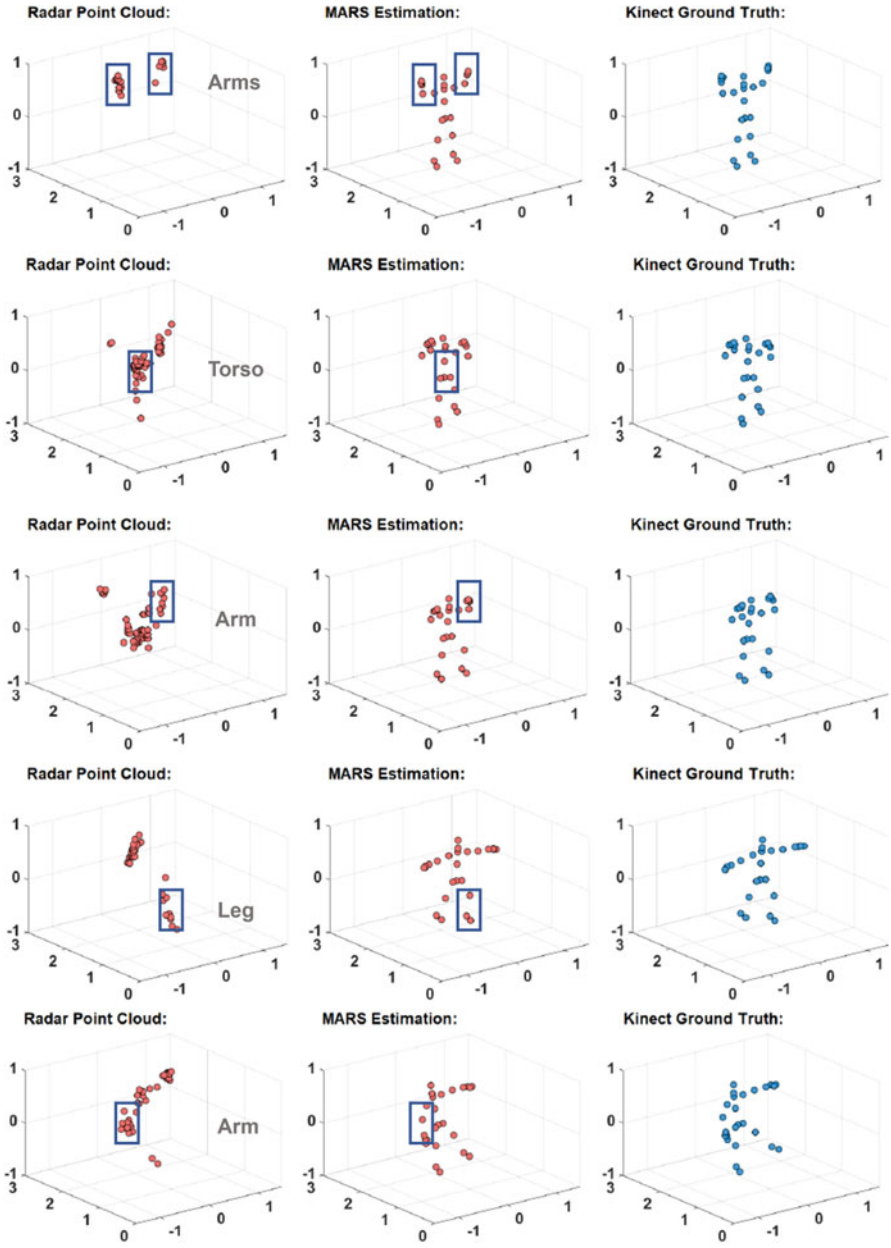

Fig. 4 Example of human pose estimation using mmWave radar point cloud. From left to right, it shows radar point cloud, MARS estimation, and the ground truth [2]. Some of the human parts are highlighted by the bounding boxes in the figure. The figure is partially modified from [2]

Model inference of MARS takes only 64 μs and consumes 442 μJ energy on the Nvidia Jetson Xavier-NX board. These results show the practicality of the proposed technique running real time on low-power edge devices. The accuracy of human pose estimation using mmWave is comparable to that of an RGB image. However, the explainability of the model and solution for free-form human pose estimation is still challenging. There are several future challenges for mmWave human pose estimation to be widely applied in real-life applications.

3.3.3 Human Pose Estimation Using Inertial Sensors

Besides cameras and radars, wearables such as inertial sensors (IMUs) also play an essential role in human pose estimation. IMU-based human pose estimation is relatively robust to different environmental settings since sensing is not interfered with by light conditions or visibility. Thus, it is more practical for occlusions or baggy clothing scenarios. In addition, the explainability of IMUs-based human pose estimation is pretty good since every IMU is placed in a specific position of a person. As one of the earliest studies in this field, [68] estimate human pose using 17 IMUs, and a Kalman filter is employed for all the measurements. It comprehensively defined 17 IMUs on a person, thus achieving accurate human pose estimation. However, the large number of IMUs requires long setup times and makes it uncomfortable for users. Marcard et al. proposed sparse inertial poser (SIP) [88]: automatic 3D human pose estimation from sparse IMUs. This work provides a new method to estimate the human pose using only six IMUs. By exploiting a statistical body model and jointly optimizing posture over continuous time frames to fit both orientation and acceleration data, SIP achieves positional errors of 3.9 cm. A follow-up work [89] combines IMUs and a moving camera to estimate multiple human poses in challenging outdoor scenes robustly.

In summary, human pose estimation can be performed using different input sources. Table 2 compares different input sources in terms of accuracy, privacy concern, price, and the anti-interference ability. Lightweight embedded machine learning algorithms enable running human pose estimation models on edge devices. In real-life applications, it is crucial to choose proper input sources of the human pose estimation model according to different requirements such as accuracy, privacy, and robustness.

Table 2 Comparison between different input sources for human pose estimation. Anti-interference here represents the robustness of the algorithm. Specifically, different input sources are affected by environmental conditions such as light and smoke to varying degrees

	Data form	Accuracy	Privacy	Price	Anti-interference
Camera	Image/video	⋆⋆⋆	⋆	⋆⋆	⋆
LiDAR	Point cloud	⋆⋆⋆	⋆⋆⋆	⋆	⋆⋆⋆
Radar	Point cloud/heatmap	⋆⋆	⋆⋆⋆	⋆⋆	⋆⋆⋆
IMUs	Accelerations	⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆

4 Energy Management

The commonly used edge devices include smartphones, smartwatches, and other wearable devices. Small form-factor devices, in particular, have severely limited battery capacity due to form-factor requirements, such as small size, lightweight, and flexibility. For example, the Oura Ring 3 incorporates a 22 mAh–3.7 V battery and advertises a battery life of 4 to 7 days [56]. Despite this limitation, these devices collect significant amounts of data to enable sophisticated health monitoring applications discussed in previous sections. The energy consumption soars if they transmit these data to a mobile device or to the cloud since wireless streaming of information is prohibitively power-consuming. Consequently, the recharge interval is short, which causes the users to stop using these devices. Therefore, significant research effort focuses on reducing the dependency of wearable devices on batteries.

Reducing the dependency on batteries critically depends on the developments in three main research areas: (1) energy harvesting (EH), (2) energy management, and (3) low-power design:

(1) **Energy harvesting** refers to techniques that generate power from ambient resources, such as light and motion. It provides a complementary energy source to batteries.
(2) **Energy management** encompasses the techniques that aim to optimally utilize the available energy from batteries and energy harvesting sources. It regulates the energy consumption to maximize the user experience within the constraints set by the energy source and the device.
(3) **Low-power design** refers to a broad class of design styles that aim to minimize the power consumption of the devices, while they meet the processing requirements. Popular techniques include clock and power gating, leakage power minimization, and heterogeneous computing, such as using domain-specific hardware accelerators to boost energy efficiency.

This chapter overviews energy management, a key enabler, and service application for edge devices running target ML applications. To this end, it first summarizes the wearable energy harvesting modalities and provides a general idea of the energy budget for wearable devices. Then, it presents an overview of optimal energy management techniques. We leave the low-power design practices and optimizations on energy consumption out of this chapter. We refer the interested readers to other surveys and books on low-power design techniques [15, 62].

4.1 Energy Sources and Budget

Wearable EH techniques generate usable electrical energy from various sources in a user's environment while conforming to the physical and comfort constraints associated with the wearable form factor [48]. The most common energy sources

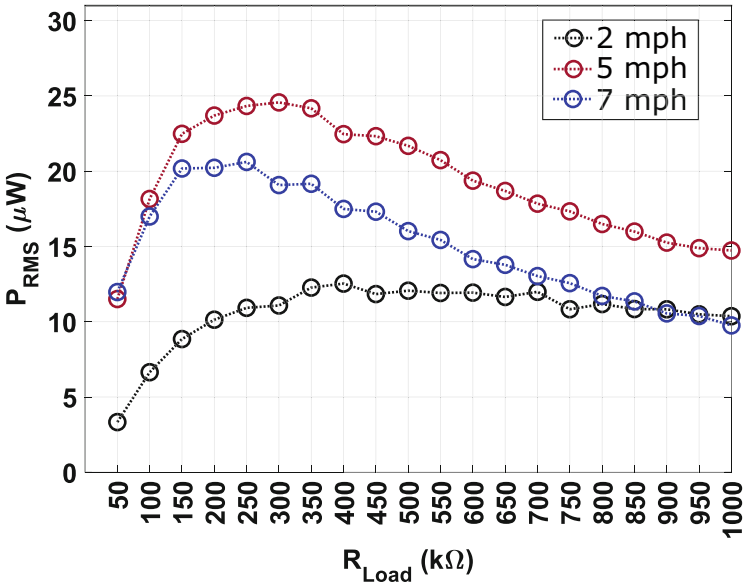

Fig. 5 Power vs. load resistance curve of a wearable piezoelectric energy harvester at different gait speeds

are light, motion, electromagnetic waves, and heat [78]. Ambient light has the highest potential for wearable EH devices. For instance, with an 8.1 cm^2 flexible PV cell, ambient light EH offers a capacity of over 1 mW outdoors (5000 lux) and close to 100 μW indoors (500 lux) [39]. Similarly, radio-frequency (RF) EH can harvest 10 μW with an 18.4 cm^2 flexible antenna with a signal strength of −10 dBm at 915 MHz [51]. Body-heat EH has power levels of about 3 μW with a 1 cm^2 flexible harvester at an ambient temperature of 15 °C (i.e., a temperature difference of 22 °C) [36]. Human-motion EH is particularly interesting for wearable applications because the energy is available on demand. For example, energy due to human activity is at hand when required by an activity-monitoring application. In addition, human-motion EH can harvest about 25 μW with a 23.8 cm^2 piezoelectric transducer, while the wearer is jogging (i.e., 5 mph gait speed) [84, 85], as illustrated in Fig. 5.

4.2 Optimal Energy Management

We can enable a long-term recharge-free operation if the edge devices have an energy-neutral operation. Energy neutrality means that the energy consumed over a time period (e.g., one day) is less than or equal to the energy produced during the same period. When the device has a battery, the energy stored in the battery can

temporarily power the device if the harvested energy is insufficient. However, the battery will be recharged back to its original level if the energy-neutral operation is guaranteed. Hence, the energy-neutral operation can automate battery charging through harvested energy such that the battery level is restored at the end of each day. In the absence of a battery, this system reduces to intermittent computing where the operating halts when no energy is available. In summary, optimal energy management techniques can maximize the device utility (e.g., the amount of time it remains active) under the available energy budget, whether from a battery or energy harvesting source.

Achieving energy-neutral operation is challenging due to the conflict between the uncertainty in harvested energy and the target application's quality-of-service (QoS) requirements. The application performance and utilization of the device can diminish when the harvested energy is limited [30]. For example, consider a wearable health application where the target device must collect the vital signals and process them locally to detect abnormalities. On the one hand, the device needs a steady and sufficient amount of energy to perform its intended operation, e.g., analyzing the collected signals within a deadline. On the other hand, the harvested energy may fluctuate widely and even vanish entirely during the same period. Therefore, the limited and highly varying nature of the harvested energy necessitates deliberate planning and management.

Energy management algorithms use the available energy judiciously to maximize the application performance while minimizing manual recharge interventions to achieve energy-neutral operation [86]. These algorithms should satisfy the following conditions to be deployed on a wearable resource-constrained device:

- Incurring low execution time and power consumption overhead
- Having a small memory footprint
- Being responsive to the changes in the environment
- Learning to adopt such changes

Kansal et al. [40] present the general framework of energy-neutral operation for energy harvesting devices. The authors propose a linear programming approach to maximize the duty cycle of a sensor node and a lightweight heuristic to help solve the linear programming with ease. Similarly, the work in [22] proposes a long-term energy management algorithm, referred to as long-term ENO, which aims to achieve energy neutrality for one year or more. As complementary to this work, going with a more fundamental control-theory approach, Geissdoerfer et al. [31] propose a feedback controller to achieve long-term ENO. To account for the application requirements when deciding the duty cycle of the nodes, Bhat et al. use a generalized utility function that defines the application characteristics [17]. They present a lightweight framework based on the closed-form solution of the optimization problem that maximizes the utility while maintaining energy-neutral operation. Although these are essential studies in this field, none of them consider user activity patterns, hence the stochastic nature of energy harvesting, which is at the core of wearable energy harvesting techniques.

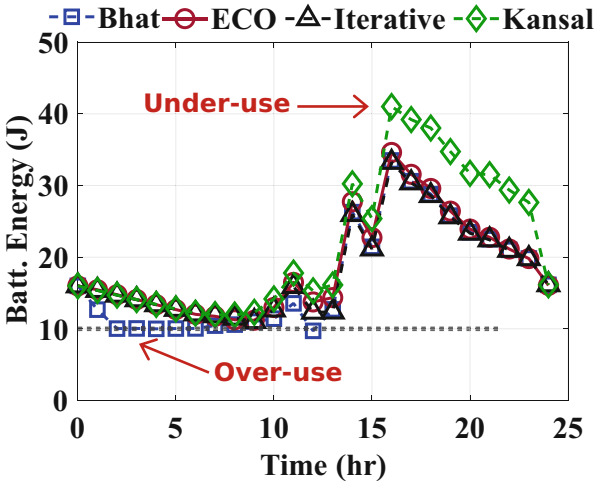

Fig. 6 Comparison of ECO to other approaches and an optimal iterative algorithm

A recently proposed energy management framework, ECO, is tailored for wearable use cases by incorporating user activity and energy harvesting uncertainty into energy management [86]. The ECO framework maximizes a utility function under specific battery energy constraints. The utility function can model any arbitrary metric, such as device throughput and classification accuracy. The framework takes the initial battery energy at the beginning of each day and the expected energy harvesting profile for a finite horizon (e.g., 24 h) as inputs. The expected energy harvesting profile is obtained from a novel EH forecasting model, which considers user patterns. At the beginning of the day, using the energy harvesting profile, ECO first finds the energy that the device can consume during each hour that maximizes the utility function under the battery energy constraints. As the day progresses, this solution is not optimum due to the variations in harvested energy. Using the actual harvested energy, ECO corrects the initial allocations using a lightweight runtime optimization algorithm after an hour. As a result, the ECO framework adapts to the deviations from the expected EH values with negligible runtime overhead. Figure 6 illustrates how ECO addresses the over-utilization and under-utilization of the energy seen in two prior approaches, which result in higher application utility. Moreover, measurements on a wearable device prototype show that ECO has 1000× smaller energy overhead than iterative optimal approaches with a negligible loss in utility.

ECO and other prior work are highly dependent upon the accuracy of the energy forecasts. They can compensate for deviations in user patterns *after the fact*, which causes a deviation from the optimal trajectory. As a remedy, reinforcement learning (RL)-based resource management algorithms are employed for energy management. RL-based approaches benefit from not relying on forecasts of the harvested energy, in contrast to the prediction-based techniques presented above.

These techniques implicitly learn user patterns, and they can proactively perturb the allocations before a significant deviation in usage pattern happens. RLMan is a recent prediction-free energy management approach based on RL [8]. It aims to maximize packet generation rate while avoiding power failures. tinyMAN is another RL-based approach that takes the battery level and the previous harvested energy values as inputs (states). It maximizes the utility of the device by judiciously allocating the harvested energy throughout the day (action). [14]. Over time, by interacting with the environment, the tinyMAN agent learns to manage the available energy on the device according to the harvested energy.

In conclusion, energy management remains a fundamental research field essential for the success of the embedded AI field. Energy management techniques must run as background service applications to ensure the successful operation of embedded AI applications under the available energy budget. Significant challenges include developing low-overhead techniques that maximize energy efficiency under uncertainty and scarce energy resources. Developing novel energy forecasting models is an important research direction for prediction-based approaches. Similarly, efficient runtime learning of user patterns is critical for prediction-free approaches. Finally, the developments in this field can transfer to other resource allocation problems in many different areas, including telecommunications to space-based systems.

5 Conclusions

Embedded machine learning enables numerous novel applications since low-power edge devices allow cutting-edge machine learning algorithms to obtain data from multiple sensors and run locally. This chapter overviewed the opportunities and challenges in the embedded machine learning applications context. It presented a survey of edge-AI application use cases for embedded machine learning. First, it overviewed embedded machine learning frameworks, consisting of model training, model compression, and model inference. Then, it presented several edge-AI applications for healthcare, such as freezing-of-gait identification for Parkinson's disease patients, human activity recognition, and human pose estimation. Finally, we discussed energy management as a fundamental enabler for wearable devices since battery shortage is one of the leading factors that limit embedded machine learning on wearable devices. Lightweight machine algorithms for these high-impact applications and other novel applications offer unique research opportunities.

References

1. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software available from https://tensorflow.org

2. An, S., Ogras, U.Y.: MARS: mmWave-based assistive rehabilitation system for smart healthcare. ACM Trans. Embed. Comput. Syst. **20**(5s), 1–22 (2021)
3. An, S., Ogras, U.Y.: Fast and scalable human pose estimation using mmWave point cloud (2022). Preprint. arXiv:2205.00097
4. An, S., Bhat, G., Gumussoy, S., Ogras, U.: Transfer learning for human activity recognition using representational analysis of neural networks (2020). Preprint. arXiv:2012.04479
5. An, S., Tuncel, Y., Basaklar, T., Krishnakumar, G.K., Bhat, G., Ogras, U.Y.: Mgait: model-based gait analysis using wearable bend and inertial sensors. ACM Trans. Internet Things **3**(1), 1–24 (2021)
6. Anguita, D., Ghio, A., Oneto, L., Parra, F.X.L., Ortiz, J.L.R.: Energy efficient smartphone-based activity recognition using fixed-point arithmetic. J. Univ. Comput. Sci. **19**(9), 1295–1314 (2013)
7. Antunes, J., Bernardino, A., Smailagic, A., Siewiorek, D.P.: AHA-3D: a labelled dataset for senior fitness exercise recognition and segmentation from 3D skeletal data. In: Prof. of the British Machine Vision Conference (BMVC), p. 332 (2018)
8. Aoudia, F.A., Gautier, M., Berder, O.: RLMan: an energy manager based on reinforcement learning for energy harvesting wireless sensor networks. IEEE Trans. Green Commun. Netw. **2**(2), 408–417 (2018)
9. Apple: Apple Watch. Helping your patients identify early warning signs. https://www.apple.com/healthcare/apple-watch/ (2021). Accessed 8 Jul 2021
10. Arami, A., Poulakakis-Daktylidis, A., Tai, Y.F., Burdet, E.: Prediction of gait freezing in Parkinsonian patients: a binary classification augmented with time series prediction. IEEE Trans. Neural Syst. Rehabil. Eng. **27**(9), 1909–1919 (2019)
11. Arduino: Arduino. https://www.arduino.cc/ (2021). Accessed 8 Jul 2021
12. Basaklar, T., Tuncel, Y., An, S., Ogras, U.: Wearable devices and low-power design for smart health applications: challenges and opportunities. In: 2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), pp. 1–1. IEEE, Piscataway (2021)
13. Basaklar, T., Tuncel, Y., Ogras, U.Y.: Subject-independent freezing of gait (FoG) prediction in Parkinson's disease patients. In: 2021 IEEE Biomedical Circuits and Systems Conference (BioCAS), pp. 1–6. IEEE, Piscataway (2021)
14. Basaklar, T., Tuncel, Y., Ogras, U.Y.: tinyMAN: lightweight energy manager using reinforcement learning for energy harvesting wearable IoT devices (2022). Preprint. arXiv:2202.09297
15. Bellaouar, A., Elmasry, M.: Low-power digital VLSI design: circuits and systems. Springer Science & Business Media (2012)
16. Bevilacqua, A., MacDonald, K., Rangarej, A., Widjaya, V., Caulfield, B., Kechadi, T.: Human activity recognition with convolutional neural networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 541–552. Springer, Berlin (2018)
17. Bhat, G., Park, J., Ogras, U.Y.: Near-optimal energy allocation for self-powered wearable systems. In: Proceedings of International Conference on Computer-Aided Design (ICCAD), pp. 368–375 (2017)
18. Bhat, G., Deb, R., Chaurasia, V.V., Shill, H., Ogras, U.Y.: Online human activity recognition using low-power wearable devices. In: 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–8. IEEE, Piscataway (2018)
19. Bhat, G., Tuncel, Y., An, S., Ogras, U.Y.: Wearable IoT devices for health monitoring. TechConnect Briefs **2019**, 357–360 (2019)
20. Bikias, T., Iakovakis, D., Hadjidimitriou, S., Charisis, V., Hadjileontiadis, L.J.: DeepFog: an IMU-based detection of freezing of gait episodes in Parkinson's disease patients via deep learning. Front. Robot. AI **8** (2021)
21. Borzì, L., Mazzetta, I., Zampogna, A., Suppa, A., Olmo, G., Irrera, F.: Prediction of freezing of gait in Parkinson's disease using wearables and machine learning. Sensors **21**(2), 614 (2021)
22. Buchli, B., Sutton, F., Beutel, J., Thiele, L.: Dynamic power management for long-term energy neutral operation of solar energy harvesting systems. In: Proceedings of the Conference on Embedded Network Sensor Systems, pp. 31–45 (2014)

23. Camille Simon-Al-Araji. Bringing AI to the NBA (2019)
24. Camps, J., Sama, A., Martin, M., Rodriguez-Martin, D., Perez-Lopez, C., Arostegui, J.M.M., Cabestany, J., Catala, A., Alcaine, S., Mestre, B., et al.: Deep learning for freezing of gait detection in Parkinson's disease patients in their homes using a waist-worn inertial measurement unit. Knowl. Based Syst. **139**, 119–131 (2018)
25. Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7291–7299 (2017)
26. Choi, S., Choi, S., Kim, C.: MobileHumanPose: toward real-time 3D human pose estimation in mobile devices. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2328–2338 (2021)
27. Deb, R., Bhat, G., An, S., Ogras, U., Shill, H.: Trends in technology usage for Parkinson's disease assessment: a systematic review. medRxiv (2021)
28. Demrozi, F., Bacchin, R., Tamburin, S., Cristani, M., Pravadelli, G.: Toward a wearable system for predicting freezing of gait in people affected by Parkinson's disease. IEEE J. Biomed. Health Inform. **24**(9), 2444–2451 (2019)
29. El-Attar, A., Ashour, A.S., Dey, N., El-Kader, H.A., El-Naby, M.M.A., Shi, F.: Hybrid DWT-FFT features for detecting freezing of gait in Parkinson's disease. In: Information Technology and Intelligent Transportation Systems, pp. 117–126. IOS Press, Amsterdam (2019)
30. Fraternali, F., Balaji, B., Sengupta, D., Hong, D., Gupta, R.K.: Ember: energy management of batteryless event detection sensors with deep reinforcement learning. In: Proceedings of the 18th Conference on Embedded Networked Sensor Systems, pp. 503–516 (2020)
31. Geissdoerfer, K., Jurdak, R., Kusy, B., Zimmerling, M.: Getting more out of energy-harvesting systems: energy management under time-varying utility with PREAcT. In: Proceedings of the 18th International Conference on Information Processing in Sensor Networks, pp. 109–120 (2019)
32. Google: Google completes Fitbit acquisition. https://blog.google/products/devicesservices/fitbit-acquisition/ (2021). Accessed 8 Jul 2021
33. Handojoseno, A.M.A., Shine, J.M., Nguyen, T.N., Tran, Y., Lewis, S.J.G., Nguyen, H.T.: Analysis and prediction of the freezing of gait using EEG brain dynamics. IEEE Trans. Neural Syst. Rehabil. Eng. **23**(5), 887–896 (2014)
34. Hastie, T., Tibshirani, R., Friedman, J.H., Friedman, J.H.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, vol. 2. Springer, Berlin (2009)
35. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
36. Huu, T.N., Van, T.N., Takahito, O.: Flexible thermoelectric power generator with Y-type structure using electrochemical deposition process. Appl. Energy **210**, 467–476 (2018)
37. IBM: Natural Language Processing (NLP). https://www.ibm.com/cloud/learn/naturallanguage-processing (2021). Accessed 8 Jul 2021
38. IBM: What is computer vision? https://www.ibm.com/topics/computer-vision (2021). Accessed 8 Jul 2021
39. Jokic, P., Magno, M.: Powering smart wearable systems with flexible solar energy harvesting. In: IEEE International Symposium on Circuits and Systems, pp. 1–4 (2017)
40. Kansal, A., Hsu, J., Zahedi, S., Srivastava, M.B.: Power management in energy harvesting sensor networks. ACM Trans. Embedd. Comput. Syst. **6**(4), 32 (2007)
41. Li, B., Zhang, Y., Tang, L., Gao, C., Gu, D.: Automatic detection system for freezing of gait in Parkinson's disease based on the clustering algorithm. In: 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp. 1640–1649. IEEE, Piscataway (2018)
42. Li, B., Yao, Z., Wang, J., Wang, S., Yang, X., Sun, Y.: Improved deep learning technique to detect freezing of gait in Parkinson's disease based on wearable sensors. Electronics **9**(11), 1919 (2020)
43. Li, J., Bian, S., Zeng, A., Wang, C., Pang, B.,Liu, W., Lu, C.: Human pose regression with residual log-likelihood estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 11025–11034 (2021)

44. Mancini, M., et al.: Measuring freezing of gait during daily-life: an open-source, wearable sensors approach. J. Neuroeng. Rehabil. **18**(1), 1–13 (2021)
45. Masiala, S., Huijbers, W., Atzmueller, M.: Feature-set-engineering for detecting freezing of gait in Parkinson's disease using deep recurrent neural networks (2019). Preprint. arXiv:1909.03428
46. Meng, Z., et al.: Gait recognition for co-existing multiple people using millimeter wave sensing. In: Proceedings of AAAI Conference on Artificial Intelligence, vol. 34, pp. 849–856 (2020)
47. Mikos, V., Heng, C.-H., Tay, A., Yen, S.-C., Chia, N.S.Y., Koh, K.M.L., Tan, D.M.L., Au, W.L.: A neural network accelerator with integrated feature extraction processor for a freezing of gait detection system. In: 2018 IEEE Asian Solid-State Circuits Conference (A-SSCC), pp. 59–62. IEEE, Piscataway (2018)
48. Mitcheson, P.D., Yeatman, E.M., Rao, G.K., Holmes, A.S., Green, T.C.: Energy harvesting from human and machine motion for wireless electronic devices. Proc. IEEE **96**(9), 1457–1486 (2008)
49. Naghavi, N., Wade, E.: Prediction of freezing of gait in Parkinson's disease using statistical inference and lower–limb acceleration data. IEEE Trans. Neural Syst. Rehabil. Eng. **27**(5), 947–955 (2019)
50. Naghavi, N., Miller, A., Wade, E.: Towards real-time prediction of freezing of gait in patients with Parkinson's disease: addressing the class imbalance problem. Sensors **19**(18), 3898 (2019)
51. Nguyen, S., Amirtharajah, R.: A hybrid RF and vibration energy harvester for wearable devices. In: IEEE Applied Power Electronics Conference, pp. 1060–1064 (2018)
52. Nvidia. Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nanodeveloper-kit (2021). Accessed 8 Jul 2021
53. O'Day, J., Lee, M., Seagers, K., Hoffman, S., Jih-Schiff, A., Kidziński, Ł., Delp, S., Bronte-Stewart, H.: Assessing inertial measurement unit locations for freezing of gait detection and patient preference. J. Neuroeng. Rehabil. **19**(1), 1–15 (2022)
54. Odemakinde, E.: Human pose estimation with deep learning – ultimate overview in 2021 (2021)
55. Oung, Q.W., Basah, S.N., Muthusamy, H., Vijean, V., Lee, H., Khairunizam, W., Bakar, S.A., Razlan, Z.M., Ibrahim, Z.: Objective evaluation of freezing of gait in patients with Parkinson's disease through machine learning approaches. In: 2018 International Conference on Computational Approach in Smart Systems Design and Applications (ICASSDA), pp. 1–7. IEEE, Piscataway (2018)
56. Oura. OURA – The most accurate guide on Sleep, Readiness, and Activity [Online] https://ouraring.com/. Accessed 1 Oct 2021
57. Pardoel, S.: Detection and prediction of freezing of gait in Parkinson's disease using wearable sensors and machine learning (2021)
58. Pardoel, S., Kofman, J., Nantel, J., Lemaire, E.D.: Wearable-sensor-based detection and prediction of freezing of gait in Parkinson's disease: a review. Sensors **19**(23), 5141 (2019)
59. Pardoel, S., Shalin, G., Nantel, J., Lemaire, E.D., Kofman, J.: Early detection of freezing of gait during walking using inertial measurement unit and plantar pressure distribution data. Sensors **21**(6), 2246 (2021)
60. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Steiner, A.B., Fang, L., Bai, J., Chintala, S.: PyTorch: an imperative style, high performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates Inc., Red Hook (2019)
61. PyTorch: PyTorch Mobile. https://pytorch.org/mobile/home/ (2022). Accessed 8 Jul 2021
62. Rabaey, J.M., Pedram, M.: Low Power Design Methodologies, vol. 336. Springer Science & Business Media, Berlin (2012)
63. Rad, N.M., Laarhoven, T.V., Furlanello, C., Marchiori, E.: Novelty detection using deep normative modeling for IMU-based abnormal movement monitoring in Parkinson's disease and autism spectrum disorders. Sensors **18**(10), 3533 (2018)

64. Ramanan, D., Forsyth, D.A., Zisserman, A.: Strike a pose: tracking people by finding stylized poses. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 271–278. IEEE, Piscataway (2005)
65. Rashid, N., Demirel, B.U., Al Faruque, M.A.: AHAR: adaptive CNN for energy-efficient human activity recognition in low-power edge devices. IEEE Internet Things J. **9**(15), 13041–13051 (2022)
66. Raspberry Pi: Raspberry Pi. https://www.raspberrypi.com/documentation/ (2021). Accessed 8 Jul 2021
67. Reches, T., Dagan, M., Herman, T., Gazit, E., Gouskova, N.A., Giladi, N., Manor, B., Hausdorff, J.M.: Using wearable sensors and machine learning to automatically detect freezing of gait during a fog-provoking test. Sensors **20**(16), 4474 (2020)
68. Roetenberg, D., Luinge, H., Slycke, P.: Xsens MVN: full 6DOF human motion tracking using miniature inertial sensors. Xsens Motion Technol. BV. Tech. Rep. **1**, 1–7 (2009)
69. Samà, A., Rodríguez-Martín, D., Pérez-López, C., Català, A., Alcaine, S., Mestre, B., Prats, A., Crespo, M.C., Bayés, À.: Determining the optimal features in freezing of gait detection through a single waist accelerometer in home environments. Pattern Recogn. Lett. **105**, 135–143 (2018)
70. Sani, S., Wiratunga, N., Massie, S.: Learning deep features for KNN-based human activity recognition. In: CEUR Workshop Proceedings (2017)
71. Schaafsma, J.D., Balash, Y., Gurevich, T., Bartels, A.L., Hausdorff, J.M., Giladi, N.: Characterization of freezing of gait subtypes and the response of each to levodopa in Parkinson's disease. Eur. J. Neurol. **10**(4), 391–398 (2003)
72. Sengupta, A., Jin, F., Zhang, R., Cao, S.: Mm-pose: real-time human skeletal posture estimation using mmWave radars and CNNs. IEEE Sensors J. **20**(17), 10032–10044 (2020)
73. Shalin, G., Pardoel, S., Lemaire, E.D., Nantel, J., Kofman, J.: Prediction and detection of freezing of gait in Parkinson's disease from plantar pressure data using long short-term memory neural-networks. J. Neuroeng. Rehabil. **18**(1), 1–15 (2021)
74. Shao, L., Han, J., Xu, D., Shotton, J.: Computer vision for RGB-D sensors: kinect and its applications [special issue intro]. IEEE Trans. Cybern. **43**(5), 1314–1317 (2013)
75. Shi, B., Yen, S.C., Tay, A., Tan, D.M.L., Chia, N.S.Y., Au, W.L.: Convolutional neural network for freezing of gait detection leveraging the continuous wavelet transform on lower extremities wearable sensors data. In: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), pp. 5410–5415. IEEE, Piscataway (2020)
76. Shoaib, M., Bosch, S., Incel, O.D., Scholten, H., Havinga, P.J.M.: A survey of online activity recognition using mobile phones. Sensors **15**(1), 2059–2085 (2015)
77. Sigcha, L., Costa, N., Pavón, I., Costa, S., Arezes, P., López, J.M., De Arcas, G.: Deep learning approaches for detecting freezing of gait in Parkinson's disease patients through on-body acceleration sensors. Sensors **20**(7), 1895 (2020)
78. Sudevalayam, S., Kulkarni, P.: Energy harvesting sensor nodes: survey and implications. IEEE Commun. Surv. Tutorials **13**(3), 443–461 (2010)
79. Svitla: CPU, GPU, and TPU for fast computing. https://svitla.com/blog/cpu-gpu-and-tpu-forfast-computing-in-machine-learning-and-neural-networks (2021). Accessed 8 Jul 2021
80. TensorFlow: TensorFlow Lite: ML for mobile and edge devices. https://www.tensorflow.org/lite (2022). Accessed 8 Jul 2021
81. Texas Instruments: IWR1443BOOST. https://www.ti.com/tool/IWR1443BOOST (2014). Accessed 29 Sep 2020
82. tinyML: tinyML Summit ahead! https://www.tinyml.org/ (2021). Accessed 8 Jul. 2021
83. Torvi, V.G., Bhattacharya, A., Chakraborty, S.G.: Deep domain adaptation to predict freezing of gait in patients with Parkinson's disease. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1001–1006. IEEE, Piscataway (2018)
84. Tuncel, Y., Bandyopadhyay, S., Kulshrestha, S.V., Mendez, A., Ogras, U.Y.: Towards wearable piezoelectric energy harvesting: modeling and experimental validation. In: Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design, pp. 55–60 (2020)

85. Tuncel, Y., Basaklar, T., Ogras, U.: How much energy can we harvest daily for wearable applications? In: 2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), pp. 1–6. IEEE, Piscataway (2021)
86. Tuncel, Y., Bhat, G., Park, J., Ogras, U.: ECO: enabling energy-neutral IoT devices through runtime allocation of harvested energy. IEEE Internet Things J. **9**(7), 4833–4848 (2022) https://doi.org/10.1109/JIOT.2021.3106283
87. Vakanski, A., Jun, H.-p., Paul, D., Baker, R.: A data set of human body movements for physical rehabilitation exercises. Data **3**(1), 2 (2018)
88. Von Marcard, T., Rosenhahn, B., Black, M.J., Pons-Moll, G.: Sparse inertial poser: automatic 3D human pose estimation from sparse IMUs. In: Computer Graphics Forum, vol. 36, pp. 349–360. Wiley Online Library (2017)
89. von Marcard, T., Henschel, R., Black, M.J., Rosenhahn, B., Pons-Moll, G.: Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 601–617 (2018)
90. Wang, A., Chen, G., Yang, J., Zhao, S., Chang, C.-Y.: A comparative study on human activity recognition using inertial sensors in a smartphone. IEEE Sensors J. **16**(11), 4566–4578 (2016)
91. Wikipedia: Embedded system. https://en.wikipedia.org/wiki/Embedded_system (2021). Accessed 8 Jul 2021
92. Xue, H., Ju, Y., Miao, C.,Wang, Y., Wang, S., Zhang, A., Su, L.: mmMesh: towards 3D real-time dynamic human mesh construction using millimeter-wave. In: Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services, pp. 269–282 (2021)
93. Yamashita, R., Nishio, M., Do, R.K.G., Togashi, K.: Convolutional neural networks: an overview and application in radiology. Insights Imaging **9**(4), 611–629 (2018)
94. Zebin, T., Scully, P.J., Peek, N., Casson, A.J., Ozanyan, K.B.: Design and implementation of a convolutional neural network on an edge computing smartphone for human activity recognition. IEEE Access **7**, 133509–133520 (2019)
95. Zhang, J., Zhang, D., Xu, X., Jia, F., Liu, Y., Liu, X., Ren, J., Zhang, Y.: MobiPose: real-time multi-person pose estimation on mobile devices. In: Proceedings of the 18th Conference on Embedded Networked Sensor Systems, pp. 136–149 (2020)
96. Zhao, M., et al.: RF-based 3D skeletons. In: Proceedings of Conference of the ACM Special Interest Group on Data Communication, pp. 267–281 (2018)
97. Zhu, S., Anderson, H., Wang, Y.: Reducing the power consumption of an IMU based gait measurement system. In: Pacific-Rim Conference on Multimedia, pp. 105–116. Springer, Berlin (2012)