ARTICLE IN PRESS
YDSPR:103146
Digital Signal Processing ••• (••••) ••••••
Contents lists available at ScienceDirect
Digital Signal Processing
www.elsevier.com/locate/dsp

A survey on edge computing for wearable technology

Xinqi Jin[a], Lingkun Li[b], Fan Dang[a,*], Xinlei Chen[a], Yunhao Liu[a]

[a] Tsinghua University, Beijing, China
[b] Michigan State University, East Lansing, MI, USA

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Wearable technology
Edge computing
Computation scheduling
Energy-saving

ABSTRACT

Smart wearable devices have become more and more popular in our daily life due to their unique power of "wearing-while-using." However, requirements of light weight and compact size lead to limited on-device resources in most wearable products, which hamper the development of wearable technology. Edge computing provides an opportunity for wearable devices to access more resources without violating the constraints on weight and size.

In this article, we first investigate the drawbacks of wearable devices and explore the potential of addressing such drawbacks by edge computing. Then we conduct a comprehensive survey on existing works from four aspects, *i.e.*, computation scheduling, information perception, energy-saving, and security. Finally, we point out several future research directions worth our attention. We believe that wearable devices enhanced with edge computing technologies would bring more benefits and convenience to our life shortly.

© 2021 Elsevier Inc. All rights reserved.

1. Introduction

Wearable technology is developing fast in recent years. According to the International Data Corporation, the wearable shipment volume is expected to have a five-year compound annual growth rate (CAGR) as high as 12.4% and reach from 396.0 million units in 2020 to 637.1 million units in 2024 [1]. Unlike smartphones, many wearable devices continuously stay close to human bodies, sensing from or interacting with the human body directly and automatically without much manual intervention. Therefore, wearable devices like smartwatches, wrist bands, and smart glasses can profoundly improve the quality of life in a manner that smartphones cannot achieve. For example, Mi Smart Band 5 [2] can sense the heart rate around the clock to justify the health status of the user and vibrate to remind the user in case of an unusually high heart rate. Apple Watch [3] records motion data collected by the accelerometer sensor to justify if the user suffers a hard fall. An emergency call will be initiated automatically if the user keeps immobile for about a minute after the fall is detected. In summary, the source of wearables' unique power is their "wearing-while-using" feature.

However, this "wearing-while-using" feature also brings disadvantages for wearables at the same time. Wearable devices need to be lightweight and well-fitting enough to make users feel comfortable while using them. Due to these requirements on size and weight, wearables may have to adopt low-end hardware. For example, batteries on wearables are relatively small and thus require a high recharging frequency for continuous service (*e.g.*, Apple Watch Series 6 [3] requires recharging every 18 hours). Delay-sensitive wearable applications, such as wearable-based fall detection [4,5], and Simultaneous Localization and Mapping (SLAM) in headsets [6] may not be feasible on many commercial off-the-shelf (COTS) wearables due to their limited computing capabilities.

To meet the aforementioned challenges for wearables, one promising method is to provide external resources to constrained wearables through communication technologies. For example, mobile cloud computing (MCC), which provides data storage and processing functionalities at remote cloud servers to mobile devices [7], could be utilized to enhance resource-limited mobile devices without any increase of the device size or weight. Although many works aim at enhancing various kinds of mobile devices via MCC (*e.g.*, MCC-powered searching service [8], payment service [9], *etc*), we argue that MCC is not suitable for many wearable applications due to several special characteristics of wearable devices and wearable applications. *First*, many wearable applications, such as those monitoring the health emergency of users [4,10], need to run in real-time. However, in the MCC paradigm, data is uploaded to the cloud via an unstable Internet connection, incurring a high and fluctuant latency. *Second*, many wearable applications take data continuously generated by sensors as input [10–12]. Uploading

* Corresponding author.
E-mail address: dangf09@gmail.com (F. Dang).

https://doi.org/10.1016/j.dsp.2021.103146
1051-2004/© 2021 Elsevier Inc. All rights reserved.

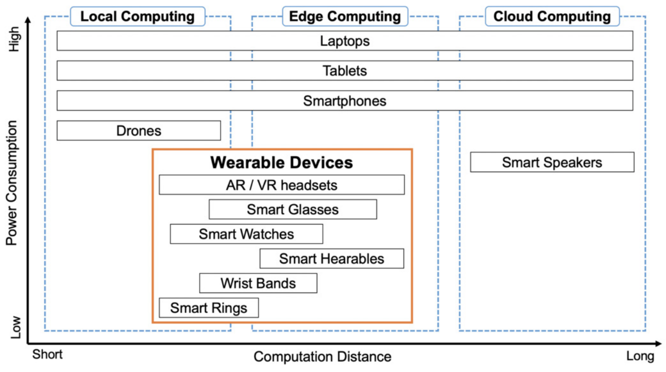


Fig. 1. The relationship between computation distance and power consumption for mobile devices. The x-axis denotes the distance between the device and the location where computation occurs.

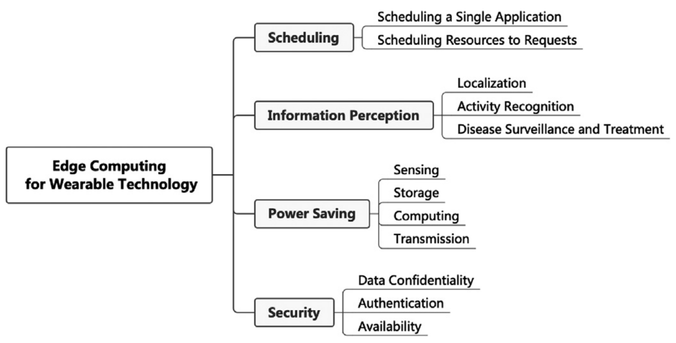


Fig. 2. Overview of existing research efforts devoted to edge computing for wearable technology.

such vast amounts of data to the mobile cloud would impose a great overhead on the Internet.

As an emerging computing paradigm, edge computing provides an opportunity to overcome these flaws of MCC while still providing resources to wearables. In this survey, the terminology "edge computing" specifically refers to multi-access edge computing (MEC). According to the European Telecommunications Standards Institute (ETSI) [13], multi-access edge computing is defined as a system that provides an IT service environment and cloud-computing capabilities through deploying devices at the edge of an access network. For convenience, we will refer to devices providing edge computing capabilities as edge gateways in this article. With a decentralized architecture, edge gateways in MEC are much closer to wearables than centralized cloud servers, leading to a lower latency, a smaller jitter, and less overhead on the Internet. A more detailed introduction to edge computing is in Section 2.2.

From the above discussion, we make such arguments: (a) wearables mainly rely on local (*i.e.*, on-device) computing [14], and edge computing [10–12]; and (b) other types of mobile devices could exploit local computing, edge computing [15,16], and cloud computing [8,9]. We summarize such views in Fig. 1. Indeed, there exist many works aiming at performing complex perception algorithms in wearable systems with the aid of edge gateways. For example, Edge-SLAM [6] offloads two computing-intensive components, the local mapping, and the loop closing, from wearables to edge gateways. Edge computing could also be utilized to enhance wearables' battery life, such as achieving energy-efficient persistent storage with encryption [17], and running optimization algorithms on edge gateways to regulate the status of wearable sensors [18,19]. Besides, due to the limited storage space of wearables, it is natural to store data generated by wearables on edge gateways — many wearables [2,3] support syncing generated data to edge gateways such as paired smartphones.

Several surveys on wearable devices or edge computing have been published in recent years. Seneviratne et al. [20] conducted a comprehensive survey on wearables in terms of communication security, energy efficiency, and computing. However, the authors only mentioned several works that combine MCC with wearables and ignored the usage of MEC in wearable applications. Ghamari et al. [21] conducted a survey on wireless body area networks for healthcare. The authors considered the wearable-based healthcare system as a four-layer model, which is essentially an edge computing system. Nevertheless, their survey mainly focuses on investigating and comparing existing low-power communication technologies suitable for wearables in a residential environment. Mach et al. [22] surveyed architecture and offloading techniques of edge computing. Abbas et al. [23] investigated edge computing in terms of architectures, advantages, applications, state-of-the-art (SOTA) research, security, *etc.* All aforementioned papers, however, focus on either wearable devices or edge computing, rather than the combination of these two technologies. Considering the great potential to integrate wearables and edge computing, we think a survey on works integrating these two technologies could be of interest to many researchers.

To the best of our knowledge, this paper is the first to present a detailed survey of edge computing for wearable technology. As shown in Fig. 2, we organize existing works on edge computing for wearables into four research topics, *i.e.*, scheduling, information perception, power saving, and security.

The remainder of the paper is organized as follows. In Section 2, we first give a brief introduction to wearable technology and edge computing. Then we discuss the potential to integrate the two technologies. In Section 3, we review each of the four critical issues presented in Fig. 2 separately. In Section 4, we discuss future research directions in edge computing for wearable technology. This paper is concluded in Section 5.

2. Wearable devices and edge computing

2.1. Wearable devices

As we have discussed in Section 1, wearables are typically made tiny in size and light in weight for user convenience. To validate this viewpoint, we have investigated the technical specifications of some popular wearable products and summarized the results in Table 1. From Table 1 we can find that most smartwatches and wrist bands are lighter than 50 grams.

We have also pointed out in Section 1 that wearables usually adopt low-end hardware due to the constraint on size and weight. To show the resource constraint of wearables more intuitively, we compare their computation capacity and battery life with other more powerful mobile devices including smartphones, tablets, and laptops in Fig. 3. As shown in Fig. 3, today's wearables either sacrifice battery life or computation capacity. In contrast, mobile devices like smartphones are not as constrained on size and weight as wearables, so they could possess a long battery life and a strong computation capacity at the same time.

Generally, the resource limitation causes three drawbacks among wearable devices:

- **Long-range communications are restricted.** As shown in Table 1, many wearable devices merely support the Bluetooth protocol, and only two of all products listed in Table 1 support Cellular networks or Wi-Fi connections. Therefore, wearable devices should typically rely on nearby smartphones or laptops to connect to the Internet.

Table 1
Technical specifications of some COTS wearable devices.

Product name	Product type	Weight (g)	Battery	Connectivity			Data source
				Cellular	Wi-Fi	Bluetooth	
Apple Watch Series 6	Smart Watch	30.5–47.1	18 hours	√	√	√	[3]
Fitbit Charge 4	Wristband	30.1	7 days			√	[24] [25]
Microsoft HoloLens 2	Mixed Reality Headset	566	2–3 hours		√	√	[26]
Bose Frames Tempo	Smart Glasses	371.4	8 hours			√	[27]
Amazon Echo Loop	Smart Ring	9–12	1 day			√	[28]
Mi Smart Band 5	Wristband	11.9	>14 days			√	[2]
Apple AirPods Max	Smart Hearable	384.8	20 hours			√	[29]
Huawei X Gentle Monster Eyewear II	Smart Glasses	44.2–48.1	5 hours			√	[30]

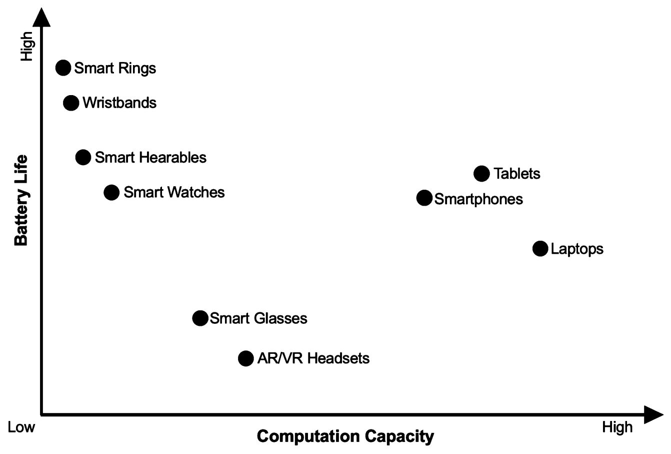


Fig. 3. Battery life and computation capacity of mobile devices. We use data in Table 1 for wearables, while the data sources for other mobile devices are [31,32]. We only use the technical specifications of today's top products to make a fair comparison.

- **Perception capabilities are impaired.** Wearable devices may not be equipped with high-end sensors which provide advanced functions with high precision at the cost of high energy consumption [33,34]. Moreover, it is difficult for wearables to continuously process the sensed data with complex algorithms to accurately obtain high-level information.
- **More vulnerable to attacks.** Wearable devices are more vulnerable to attacks than general-purpose computers for three main reasons. *First,* wearables may not have enough resources to execute antivirus software continuously for malware detection. *Second,* wearables could not afford to encrypt every message using more secure yet computing-intensive asymmetric encryption. Actually, even the commonly used and less computing-expensive methodology of exchanging a symmetric key with the assistance of asymmetric encryption, *e.g.,* the key agreement mechanism in Transport Layer Security (TLS), is not prevalent in COTS wearables. As we will discuss in Section 3.4, many research efforts are aimed at improving the security level of key exchange without the assistance of asymmetric encryption. Nevertheless, the existing key exchange methods for wearables are still less reliable than asymmetric encryption-aided methods. *Third,* since some wearables, *e.g.,* Amazon Echo Loop [28], and Hexoskin Smart Kit [35] do not have user interfaces, users may not notice any anomaly for a relatively long time since these wearables are compromised.

2.2. Edge computing

Edge computing has been widely used in multiple fields, such as video analysis, content delivery, and smart city. For instance, Ananthanarayanan et al. [36] listed five potential applications based on video analysis, including traffic management, smart cars, personal digital assistants, surveillance, and augmented reality. The data can be processed on edge gateways to achieve a low latency. Content Delivery Network (CDN) is another example. Static resources are deployed on the intermediate server close to clients to reduce the overall network latency. In a smart city, there are many sensors generating large amounts of data every day. To reduce the transmission overhead, the majority of raw data is processed at the edge [37].

Different from cloud computing, which typically only contains an end device layer and a central cloud layer, edge computing, as shown in Fig. 4, also contains an edge gateway layer. We will next briefly introduce these three layers.

The end device layer consists of various end devices, including but not limited to wearable devices, smart city devices, and smart home devices. These end devices have certain features in common. *First,* many end devices are equipped with sensors that constantly generate measurements of the physical world. The generated data is the input of many real-world applications [5,6,11,36]. For example, many COTS wearables have accelerometer embedded [2,3,26,29], and many smart speakers [38,39] are equipped with microphones to listen to voice commands from users. *Second,* these end devices are typically made with fewer resources than general-purpose computers for wide deployment at low cost. For example, widely deployed smart cameras typically have "a single CPU core, CPU speeds of 1–1.4 GHz, and 64–256 MB of RAM." [40]

As for the edge gateway layer, it consists of edge gateways that are in the vicinity of end devices and have relatively rich resources. Since resource-limited end devices may not be able to process continuously generated sensor data in real-time, many works aim at offloading computation from end devices to edge gateways to achieve a lower processing latency [41,42]. Edge gateways in the edge gateway layer can be further divided into two categories: (a) public MEC servers such as network routers, base stations, and servers customized for MEC; and (b) powerful private devices such as smartphones, tablets, and laptops. Some works target public MEC servers [6,43–45], while others focus on powerful private devices [5,10,17,46].

The central cloud layer aims at providing "a shared pool of configurable computing resources (*e.g.,* networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [47] The amount of resources that could be rapidly provisioned in today's large cloud service providers is really incredible, *e.g.,* hundreds or even thousands of cloud servers could be deployed in minutes when the users of Amazon Web Services (AWS) need to [48]. Despite much more abundant resources than edge gateways, cloud servers are typically far away from end gateways and thus not suitable for many delay-sensitive applications. As discussed in Section 3.2 and summarized in Table 4, in existing edge computing systems for wearables, the central cloud layer mainly supports large-scale data processing [11,49] and re-

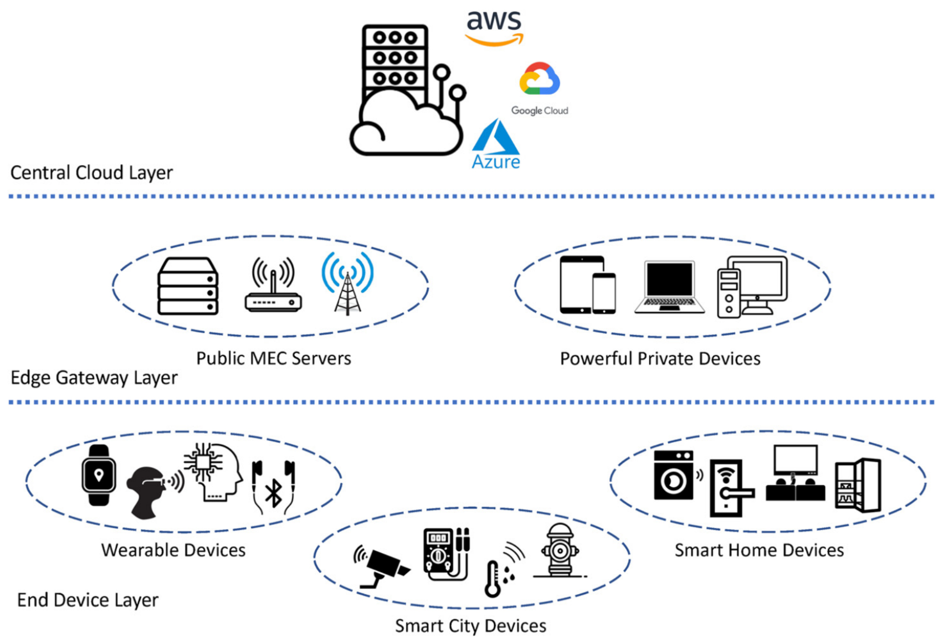


Fig. 4. Common architecture of edge computing.

Table 2
Comparison of three layers in edge computing.

Layer	Deployment	Transmission delay	Computing capacity	Storage capacity
End Device Layer	Distributed	No	Low	Low
Edge Gateway Layer	Distributed	Low	Moderate	Moderate
Central Cloud Layer	Centralized	High	High	High

mote engagement [50], rather than real-time processing of sensor data.

From the top central cloud layer to the bottom end device layer, computing and storage resources of the individual device get weaker, but the device locations are more distributed and the number of devices is growing bigger. A comparison of these three layers is shown in Table 2. End devices can offload some complex computing tasks to edge gateways, after which edge gateways can filter, process, and aggregate data generated by end devices. Then edge gateways can transmit the intermediate processing result to the central cloud for further analysis or further use. Other resources of edge gateways and cloud servers (*i.e.*, storage space) can also be utilized by end devices.

2.3. Integration of wearable technology and edge computing

As we have discussed in Section 1, edge computing is better suited to assisting wearables than MCC, due to its advantages on latency and bandwidth requirement. Enhanced by edge computing, wearables could be more powerful in the following aspects. *First,* wearables' perception ability can be further explored in combination with edge computing. The main reason is that wearables alone may not be able to perform an in-depth analysis of raw data due to limited on-device resources. By transmitting data collected by sensors to edge gateways, or even cloud servers in certain situations, more complex pattern recognition methods can be utilized, and thus more meaningful and accurate information can be recognized. *Second,* wearable devices' battery life can be greatly extended by reducing the energy consumption of sensing, storage, and computation via edge computing. This effect can be further amplified by using more energy-efficient communication protocols to transmit data between wearables and edge gateways. These two aspects will be discussed in more detail in Section 3.2 and Section 3.3 respectively.

Despite all these benefits, the adoption of edge computing in wearable applications however also brings many security concerns. In addition to subjective factors, such as insufficient attention to security when designing such systems, there are many objective reasons leading to the current fragile security situation:

- **Heterogeneous Operating Systems and Communication Protocols.** Wearables and edge gateways may run diverse operating systems, such as Linux, Real-Time Operating System (RTOS), *etc.* Some wearables even run programs directly without installing any operating systems on the device. Besides, different wearables and edge gateways may support different communication protocols, such as Narrowband Internet of Things (NB-IoT) [51], ZigBee [52], LoRa [53], Bluetooth Low Energy (BLE) [54], *etc.* Such heterogeneity introduces much difficulty in designing a unified security mechanism, since it may take time and efforts to migrate security frameworks from one wearable-oriented edge computing system to others [55].
- **Limited Resources in Wearables.** As we have discussed above, wearables only have limited resources. Therefore, they just cannot adopt complex defensive measures as general-purpose computers do and are thus more vulnerable to many kinds of attacks.

3. Existing research efforts

As shown in Fig. 2, existing research efforts devoted to edge computing for wearable technology can be grouped into four research topics, including scheduling, information perception, power

Table 3
Comparison of scheduling technologies.

Reference	Resource competition considered	Working stage		Computing nodes			Solution
		Initial server deployment	Application runtime	End device	Edge gateway	Cloud server	
[56]	No		√	√	√		Using Bandwidth Prediction to Decide Offloading Strategy
[41]	No		√	√	√	√	A Delay-minimizing Offloading Policy Considering End-to-edge, Edge-to-cloud, and Edge-to-edge Interaction
[42]	No		√	√	√		An Online Learning Based Offloading Algorithm with Adaption to Dynamics
[57]	No		√	√	√	√	A Heuristic Algorithm Giving Priority to Interactive Sub-tasks
[45]	Yes	√		√	√		Adoption of Delay Ranking for Searching Ideal Deployment Locations
[58]	Yes	√		√	√		Formulation of a Multi-objective Server Deployment Problem using Mixed Integral Programming
[59]	Yes		√	√	√		Formulating the Allocation of CPU Cycles in Edge Gateways as a Multiple Knapsack Problem
[60]	Yes		√	√	√		A Distributed Offloading Algorithm Based on Game Theory to Coordinate Usage of Multiple Wireless Channels
[46]	Yes		√	√	√	√	A Context-aware Offloading Scheme to Balance User Experience of Wearables and Energy Saving of Edge Gateways

saving, and security. Each research topic is investigated in one of the four subsections of this section.

3.1. Scheduling

In this paper, scheduling refers to the scheme used for allocating resources of local devices, edge gateways, and cloud servers to the execution of applications. Some works only consider a single application, while others take resource competition among multiple applications into consideration. We investigate these two kinds of works in Section 3.1.1, and Section 3.1.2 respectively. All surveyed works are listed in Table 3.

3.1.1. Scheduling a single application among three layers of edge computing

Many works aim at scheduling a single application, with various goals such as improving the user experience with lower execution delay, reducing the energy consumption of wearables, *etc*. Since energy saving is one of the most important research issues in wearable technology, we will discuss it as an independent topic in Section 3.3. Here we focus on works related to reducing the execution delay through computation offloading.

The total delay of a computing task is the sum of transmission delay and computing delay. On-device computing incurs the highest computing delay and no transmission delay; cloud computing incurs the lowest computing delay but the highest transmission delay. Edge computing is a trade-off between on-device computing and cloud computing since it incurs intermediate-level transmission delay as well as intermediate-level computing delay. Many works aim to divide a task into sub-tasks and assign them to different layers to achieve the optimal total delay. Multiple aspects are considered when designing the task partition and assignment strategies, including the complexity of sub-tasks, the amount of intermediate data between sub-tasks, the current status of network and execution environments (*e.g.*, available CPU capacity on each available execution environment, network bandwidth, *etc*). Wolski et al. [56] designed a scheduler to decide whether a task needs to be offloaded for better performance based on the prediction value of bandwidth between the local device and the remote server. Yousefpour et al. [41] proposed a delay-minimizing offloading scheme for edge gateways, with the goal of reducing the service delay for IoT nodes. Since the proposed framework does not restrict the "number, type, or topology" of Internet of Things (IoT) nodes, edge gateways, and cloud servers, it could be utilized to make decisions on computation offloading in edge computing systems for wearables. The authors considered the case where multiple edge gateways are available for a single application. IoT-to-cloud interaction, edge-to-cloud interaction, and communication between edge gateways are adopted to reduce the service delay through sharing the load. Wang et al. [42] also focused on the case where multiple edge gateways are available for a single application and targeted optimizing the application latency. The novelty mainly lies in considering the mobility of both IoT nodes and edge gateways, which could be very common in the case of wearables, and the temporal variation of shareable resources. To adapt to the dynamics, the authors proposed an online learning-based offloading strategy, in combination with the Combinatorial Multi-Armed Bandits (CMAB) framework.

In addition to the aforementioned works that treat the delay of each task as equally important, some works rethink the offloading problem by assigning different weights to the delay of different tasks belonging to a single wearable application. These works think that tasks involving user interaction have a more significant influence on the overall user experience and give priority to these tasks when designing the offloading scheme. Cheng et al. [57] referred to tasks that should be executed only on wearables as *w-tasks* (*e.g.*, sensing, display, *etc*), and categorized all tasks into two classes – *w-tasks* and *non-w-tasks*. They believed that the response time of *w-tasks* has a greater impact on the user experience than that of *non-w-tasks*. Therefore, to enhance the user experience, they proposed a heuristic algorithm aiming at enabling as many *w-tasks* as possible to execute within a guaranteed delay from the last executed *w-task*.

3.1.2. Scheduling resources to user requests

Resource allocation is an important research issue in the context of edge computing. As shown in Fig. 5, the resource allocation problem can be further divided into two main sub-problems based

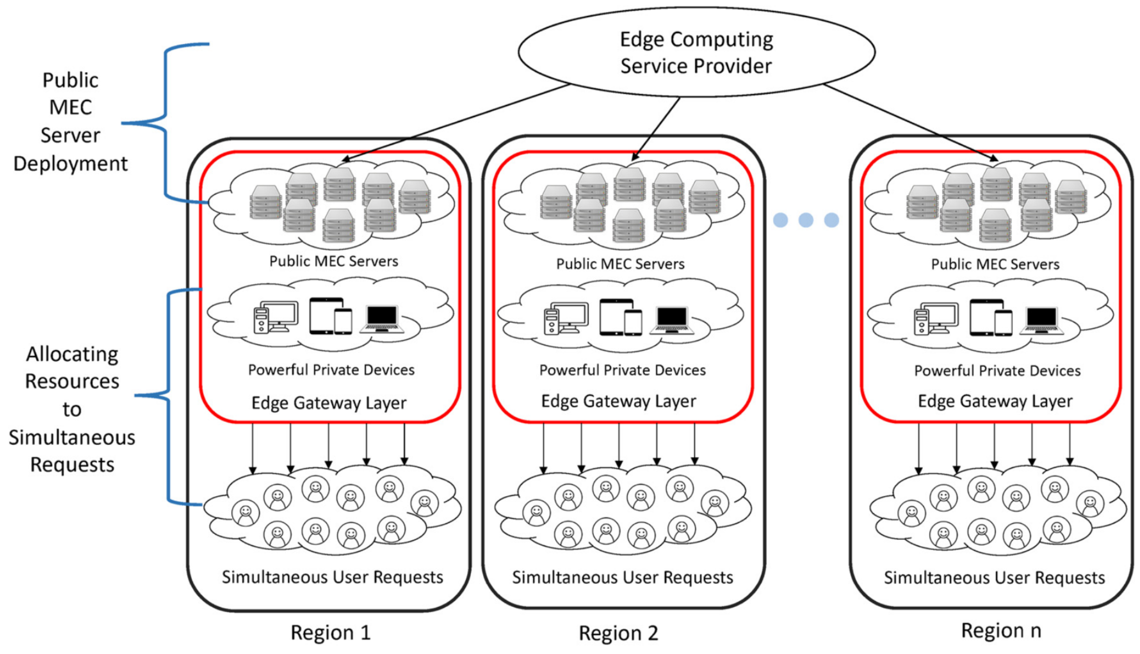

Edge Computing Service Provider
Public MEC Server Deployment
Allocating Resources to Simultaneous Requests
Public MEC Servers
Powerful Private Devices
Edge Gateway Layer
Simultaneous User Requests
Region 1
Region 2
Region n

Fig. 5. Resource competition between users. Resource allocation consists of two phases and is represented by arrows.

on the timescale of the problems [59]. One is the initial deployment of public MEC servers which occurs on a long timescale of several months, and the other is the allocation of currently available resources to multiple simultaneous user requests which happens on a short timescale of minutes or seconds. The scheme adopted to allocate finite resources to numerous users can have a great impact on the overall service quality.

As for the initial placement of public MEC servers, it is mainly involved with edge computing service provided at a large scale (*i.e.,* country-wide networks or global networks) by large companies and institutions, rather than powerful private devices owned by individual users. An inappropriate initial deployment of resources may assign too much computing resources to regions with little demand for edge computing service and insufficient resources to regions where many computing-intensive edge computing applications are running, thus wasting costly computing resources deployed at the edge. Many researchers try to improve the overall Quality of Service (QoS) of edge computing through the proper MEC server placement. Yin et al. [45] presented a decision support framework called Tentacle to determine server placement policies. Tentacle takes many factors (including monetary budgets, QoS requirements, traffic constraints, *etc*) into consideration and can identify unforeseen better edge locations. A user performance improvement as high as 10–45% is observed at global-wide networks in the simulation experiments. Wang et al. [58] formulated the server deployment problem as an optimization problem, with the goals of load balancing of public MEC servers and minimum end-to-end delay between end-users and public MEC servers.

As for the allocation of deployed resources to multiple simultaneous user requests within a region, it involves both public MEC servers and powerful private devices. An inappropriate allocation scheme may cause network congestion, *e.g.,* many users may choose the same wireless communication channel, or allocate excessive resources of edge gateways to less urgent wearables applications but only leave insufficient resources to those urgent ones, thus ruining the overall benefit of edge computing. Ketykó et al. [59] focused on competition for CPU cycles of edge gateways and reduced the multi-user resource allocation problem in edge computing to a Multiple Knapsack Problem. Chen et al. [60] noticed that in a multi-channel wireless network, users will interfere with each other if too many of them choose the same channel for data transmission. They put forward a distributed offloading mechanism based on the game theory to decide: (a) whether a task should be offloaded or executed locally; and (b) which wireless channel should be chosen to alleviate interference and achieve lower transmission delay. Yang et al. [46] proposed a context-aware task offloading (CATO) mechanism that applies to Android-based edge gateways. Specifically, in existing Android systems, there are two kinds of processes – foreground processes and background processes. Foreground processes use all CPU resources and execute faster, but they consume more energy on gateways. In contrast, background processes are more energy-saving, at the cost of executing slower with limited CPU resources. Hardware-level techniques are exploited by many Android smartphones to distinguish foreground and background processes, *e.g.,* the ARM big.LITTLE architecture consists of big CPU cores suitable for foreground processes and little CPU cores tailored for background processes. In CATO, offloaded tasks related to user interactions are considered urgent and executed in foreground processes, while other offloaded tasks are executed in background processes or even further offloaded to cloud servers. In this way, CATO achieves a balance between user experience on wearables and energy saving on gateways.

3.2. Information perception

As discussed in Section 2.1, wearable applications typically have limited perception capabilities. By partitioning and offloading tasks to edge gateways or even the central cloud, wearable devices can conduct complex analyses on raw data, thus having a better perception of the ambient environment as well as the user status, and greatly improving the quality of human life. Existing works related to extending wearables' perception ability via edge computing can be roughly grouped into three categories: (a) localization; (b) human activity recognition (HAR); and (c) disease surveillance and treatment. All investigated works are summarized in Table 4. Besides, we also compare the data quality requirement as well the computation complexity of these works. The results are shown

Table 4
Comparison of works exploiting edge computing for information perception of wearable applications.

Reference	Target	Required hardware	Core algorithm	Computation within each layer		
				End device layer	Edge gateway layer	Central cloud layer
[50]	L	GPS Sensor; RFID Tag	-	Location Estimation based on GPS	RFID Reading	Store the Location Data for Remote Access
[6]	L	Camera	A Distributed Visual-SLAM Algorithm	Tracking	Local Mapping; Loop Closing	-
[5]	A	Accelerometer	Machine Learning Algorithms for Fall Detection	–	Execution of Machine Learning Models	Model Updating; Archiving; Visualization
[12]	A	Accelerometer	A Hierarchical HAR Framework	Execution of a Lightweight Classifier and an Offload Controller	Execution of a Complex Classifier	-
[11]	A	Accelerometer; Gyroscope	A HAR Algorithm Incorporating K-means Clustering, Support Vector Machines, and Hidden Markov Models	Pre-processing	Performing Data Analysis	Processing of Data from the Whole Community
[61]	D	rs-fMRI Sensor; EEG Sensor	A Pipeline for Localization of Epileptogenicity using Multimodal Data	–	Execution of the Pipeline	–
[49]	D	Health Sensor; Environmental Sensor;Location Sensor	A Framework Incorporating Individual Infection Detection and Large-scale Outbreak Analysis	-	Individual Infection Classification	Large-scale Outbreak Analysis
[10]	D	ECG Sensor	A Heart Attack Alerting Pipeline based on the QRS Signals	-	Data Pre-processing	Performing Diagnosis
[66]	D	pH Sensor; Heater; Temperature Sensor	–	–	Not Reported	Supporting Remote Engagement

***Abbreviations used in the 2nd column**: L – Localization; A – Activity Recognition; D – Disease Surveillance and Treatment.*

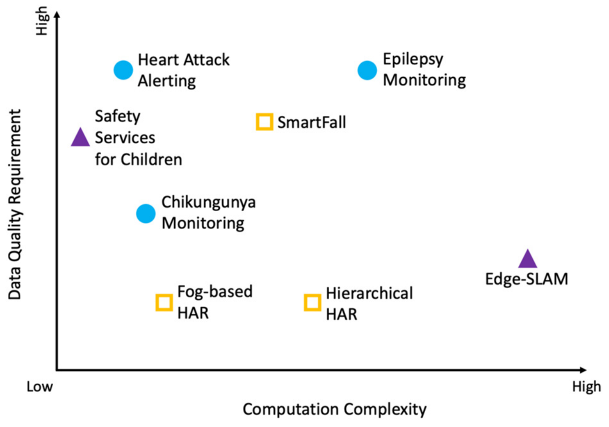


Fig. 6. The research status of enhancing wearables' perception ability via edge computing.

in Fig. 6, where works focusing on localization (Safety Services for Children [50], and Edge-SLAM [6]), activity recognition (SmartFall [5], Hierarchical HAR [12], and Fog-based HAR [11]), and disease surveillance and treatment (Epilepsy Monitoring [61], Chikungunya Monitoring [49], and Heart Attack Alerting [10]) are marked by triangles, squares, and circles respectively.

3.2.1. *Localization*

Wearable devices with localization ability are of great practical value since they can support many services such as tracking children/elderly for safety purposes, recommending nearby stores, and navigation to other places. In many cases, edge gateways are required to implement a wearable localization system. We argue that there are three main reasons for the need of edge computing in such systems. *First,* some localization methods (such as methods based on radio-frequency identification (RFID) and received signal strength indicator (RSSI)) require the participation of other devices, which are usually installed on an edge gateway in the context of wearable-based localization. *Second,* some wearable localization applications rely on an edge gateway with Internet access to upload their location to servers for further use. *Finally,* some localization methods (*e.g.,* SLAM) are computing-expensive and it is necessary to offload some tasks to edge gateways to perform such methods in a real-time manner.

Jutila et al. [50] proposed a safety service that incorporates wearable sensor technology with edge computing. Specifically, students are required to carry a device that could upload position calculated from the Global Positioning System (GPS) to servers, and RFID tags that can interact with RFID readers deployed at school. Locations, where children occur, are recorded on cloud servers and shown to teachers and parents, which can help to prevent children from being lost. Since RFID tags alone cannot infer whether a pupil appears in a specific area or upload its data to cloud servers through the Internet, edge computing plays a key role in such a system. Another example is SLAM, which consists of the simultaneous estimation of the motion state and location of a device equipped with some sensors, and the construction of a map of the ambient environment [62]. By running SLAM algorithms in headsets, users can localize themselves in a complex environment and enjoy various location-aware services such as peer-to-peer indoor navigation [63]. Nevertheless, as Xu et al. [64] have demonstrated, even the latest smartphone (*e.g.,* Samsung Galaxy S10 and Google Pixel 2) cannot execute computing-intensive ORB-SLAM [65] at 15 frames per second (FPS) alone, not to mention resource-limited wearable devices. Ali et al. [6] presented an end-edge collaborative SLAM system called Edge-SLAM to solve this problem. They demonstrated that through offloading local mapping and loop closure tasks from mobile devices with wearable-grade computing ability to the edge, the latency of these two tasks and overall memory usage in mobile devices could be greatly reduced.

3.2.2. *Activity recognition*

Activity recognition is the cornerstone of many practical applications, such as human-computer interaction, health surveillance, and lifelogging. Wearables are suited for collecting data to support this task. Since activity recognition may involve some complex algorithms to achieve high detection accuracies, edge computing is often employed. Samie et al. [12] proposed a hierarchical method for activity recognition, which consists of a binary classifier to decide whether to offload computation to the edge gateway or not, a lightweight local classifier recognizing only a subset of all activities, and a powerful edge-based classifier recognizing all possible activities (or remaining categories). The authors demonstrated that this hierarchical model can achieve 3x energy saving and slightly higher accuracy than the baseline strategy to offload everything. Mauldin et al. [5] proposed a three-layer fall detection system called SmartFall. In SmartFall, data collected by a smartwatch at the end device layer is transferred to a paired smartphone located at the edge gateway layer. After receiving the raw data, the paired smartphone will run deep learning models on the data for fall detection, send alert messages to care providers, and finally upload data to the central cloud layer for the model updating, archiving, and visualization. Similarly, Concone et al. [11] proposed a three-tier architecture for human activity recognition, in which the resource-constrained wearable devices are only used to sense and preprocess raw data, and the smartphones are responsible for executing computing-intensive activity recognition algorithms.

3.2.3. *Disease surveillance and treatment*

Wearable sensors can also collect various kinds of physiological data, opening the door for human disease surveillance and treatment. Researchers have combined these dedicated wearable sensors with edge computing to create better patient monitoring services with long battery life and short response time. Hosseini et al. [61] pointed out that the brain activity of epilepsy patients consists of four states, *i.e.*, the preictal state, ictal state, postictal state, and interictal state. When a seizure is about to occur, the brain activity is in the preictal state. Then follows the ictal state identifying "the interval during which activity manifests as a seizure." The postictal state indicates the end of the seizure. As for the interictal state, it occurs between seizures. They designed a system to monitor and treat epilepsy patients. Specifically, the system uses wearable sensors to collect electroencephalography (EEG) and resting-state functional magnetic resonance imaging (rs-fMRI) data, and then sends collected data to edge gateways to identify the preictal state as well as localize the seizure focus. Once the preictal state is identified, the system will make an alert or stimulate localized focus to prevent seizure. Sood et al. [49] targeted using edge computing for wearable technology to diagnose and prevent the outbreak of the Chikungunya virus. The proposed system has a three-tier architecture, *i.e.*, the data accumulation layer including wearable sensors as well as ambient environmental/location sensors, the edge gateway layer responsible for executing classification algorithms to identify virus infection and making alerts when necessary, and the cloud layer responsible for monitoring and controlling the outbreak of the Chikungunya virus. Gusev et al. [10] developed a heart attack alerting system consisting of a wearable electrocardiogram (ECG) biosensor, a smartphone, and a cloud server, based on the fact that "a heart attack may be predicted a couple of hours before its onset by detecting changes in the Electrocardiogram (ECG) of the patient." [67] ECG data is collected by sensors and transmitted to smartphones for preliminary processing, after which it is transmitted to cloud servers for final diagnosis. In case of an identified impending heart attack, cloud servers will send an alert message to doctors and 24h healthcare service centers together with the location of the patient's smartphone. Mostafalu et al. [66] developed a smart bandage consisting of a re-usable electronics module and a disposable patch. The patch is equipped with sensors, a microheater, and thermo-responsive drug carriers. Data sampled by sensors, which could reflect the status of chronic wounds, is transmitted to the smartphone via the Bluetooth module of the electronics module. The smartphone could be used to make decisions on drug release with the involvement of patients, caregivers, *etc.* Decisions could be transmitted to the electronics module through Bluetooth, after which the electronics module could regulate the power applied to the microheater and thus influence the amount of thermo-responsive drug carriers' emission.

3.3. *Power saving*

As shown in Table 1, many wearable products use small-sized batteries and are limited to relatively short battery life. As wearable manufacturers tend to integrate more sensors and thus leave less space for batteries, the situation is easy to deteriorate in the future.

Wearable devices' energy consumption is mainly caused by four functionalities, including sensing, storage, computation, and communication. Edge computing can reduce wearables' energy consumption caused by sensing, storage, and computation. Since adoption of edge computing may incur additional energy consumption of communication modules on wearables, we also introduce some energy-efficient communication protocols at last.

3.3.1. *Saving energy consumption of sensing*

One way to enhance wearables' battery life is to dynamically adjust their onboard sensors' working states by running optimization algorithms in edge gateways. Amiri et al. [18] observed different activities of patients result in a different degree of signal quality degeneration in the context of wearable sensors. Therefore, different sensing power is required to achieve the same Signal-to-Noise Ratio (SNR) for different activity states, *e.g.*, a higher sensing power is required for a running patient to achieve the same SNR as a sitting patient. They proposed an adaptive optimization algorithm running on the edge gateway layer. The algorithm is based on the patient's contextual information, *e.g.*, health status, activity status, *etc*, and decides the most energy-efficient working states of wearable sensors which could maintain an acceptable signal quality in all possible activity states. The edge gateway will instruct the sensors to change their working states after the optimization algorithm gives a suggestion. Similarly, Anzanpour et al. [19] designed an edge-assisted control engine to configure wearable sensors' working state adaptively, *e.g.*, switching between low power and high power mode, or choosing between periodic recording and continuous sensing, by incorporating multiple goals such as the health status of the user, the continuity of monitoring, and the sensing accuracy. The energy consumption would decrease by 44% at most from the observation.

An alternative method is to use the rate of kinetic energy harvesting (KEH) rather than specialized and energy-hungry sensors for sensing. In KEH-enabled wearables, user mobility-induced kinetic energy is exploited to charge wearables. Since the rate of KEH reflects the status of user motion and movement to some extent, KEH also provides a new perspective for perception. Moreover, some researchers pointed out that EH-based perception could be more energy-efficient than traditional sensor-based perception.

For example, Lan et al. [68] noticed that the frequency of data sampling has a significant impact on the overall energy consumption of wearables, as frequent sampling requires frequent wake-up of the microcontroller. However, existing activity recognition systems typically rely on instant acceleration data which could provide only instantaneous information, and thus require a high sampling rate. Therefore, they utilized the capacitor of the kinetic energy harvester in KEH-powered smart shoes for sensing. Since the capacitor value could provide accumulated information of the user's mobility status, the system requires a relatively low sampling rate for detecting activities of daily living (ADL). The authors demonstrated that the proposed method could detect 5 daily activities with 92% accuracy as well as reduce the overall system power by 75%. Similarly, Xu et al. [69] utilized the output voltage signal from KEH for gait recognition in wearables, which could reduce the energy consumption by 82.15% as well as maintain an equal error rate (EEH) of approximately 10%.

3.3.2. *Saving energy consumption of storage*

One surprising fact about mobile storage devices is that the software storage stack consumes up to 200 times more energy than the low-power flash storage hardware [70]. Huang et al. [17] argued that it is because of three reasons, including extended CPU idle time due to waiting for I/O of slow mobile flash [71], runtime OS overhead, and data encryption for secure storage on flash. They proposed the concept of *BB-RAM* (i.e., battery-backed RAM, which is considered to be non-volatile) and designed the WearDrive storage system, which only uses *BB-RAM* on wearables but uses both *BB-RAM* and flashes on smartphones. In the WearDrive system, data generated by wearables are transmitted to smartphones via the low-power network connection, and then smartphones are responsible for encryption and durable storage of received data. Observing that BLE is more energy-efficient for transferring small amounts of data and WiFi-Direct is better in other cases, the WearDrive system flexibly selects between BLE and WiFi-Direct to further save the energy consumption of wearables. Experiment results showed that, compared to the traditional storage process which performs the encryption locally and then writes data to the local flash, the WearDrive system achieves an up to 8.85x better performance from the aspect of I/O throughput, and an up to 3.69x battery life improvement of wearables with negligible impact on the phone.

3.3.3. *Saving energy consumption of computing*

Another way to address the energy constraint is to offload power-hungry computing tasks to edge gateways. Early to 1998, Rudenko et al. [72] thought of computation offloading as a potential method to improve battery life. They showed that a significant amount of overall energy consumption could be saved through such a computing model, in spite of additional energy cost caused by data transmission. Chun et al. [73] designed a system called CloneCloud which enables unmodified mobile applications to offload some computing tasks to remote servers automatically and seamlessly. Experiment results showed that CloneCloud can achieve a 20x speed-up as well as a 20x energy consumption saving. Zhang et al. [44] proposed an iterative search algorithm to jointly optimize energy efficiency and transmission latency in the edge computing model, in which the remaining energy of smart devices is used to calculate the weighting factor of energy consumption and latency. Specifically, the optimization algorithm assigns a large weight to energy consumption if the remaining energy of smart devices is low; otherwise, more emphasis will be put on latency. Numerical results showed that a better life of end devices is enabled by the proposed method.

3.3.4. *Saving energy consumption of communication*

To further extend battery life via edge computing, it is of great significance to reduce the energy consumption of inter-device communication. As shown in Table 5, many works focus on designing low-power communication techniques to enhance the battery life of end devices. Some of the listed works target optimizing traditional communication technologies [74–76] and still use the radio module in the sender to generate the carrier signal, which could be energy-intense for resource-limited wearables. In contrast, the latest backscatter techniques listed here [77–79] directly modulate ambient carriers instead and thus save much power.

First we introduce works that optimize the energy efficiency of traditional communication protocols. Buettner et al. [74] proposed a Media Access Control (MAC) protocol called X-MAC for wireless sensor networks (WSNs). In previous asynchronous protocols for WSNs, the sender sends the target address only after transmitting a preamble at least as long as the sleep time of the target receiver. This simple design guarantees that a synchronous receiver could always receive the packet, but it presents three serious problems: (a) the target receiver must wait for the end of the preamble even if it wakes up much earlier, resulting in low energy efficiency for the target receiver as well as the sender; (b) all non-target receivers must keep on listening until the sender transmits the target device address, leading to energy waste of these non-target devices; and (c) the sender keeps on transmitting the full-length preamble even after the target receiver wakes up, incurring a high latency. Since X-MAC substitutes the original long preamble packet with a series of intermittent short preambles, each of which contains the target address, and an awake target device could send an early acknowledgment (ACK) during the interval between two adjacent short preambles. As for non-target devices, they could immediately go back to sleep when it obtains the target address from a short preamble. And the sender could start transmitting payload data rather than remaining short preambles after it detects the early ACK. In this way, the aforementioned three problems could be effectively solved, and the energy efficiency of all nearby WSN nodes could be greatly improved. Suarez et al. [80] evaluated the effectiveness of X-MAC in a multi-hop scenario. They successfully achieved a 10x ZigBee lifetime by substituting the default MAC protocol with the X-MAC protocol. The Bluetooth Special Interest Group introduced a new feature known as Bluetooth Low Energy (BLE) in Bluetooth Core Specification v4.0 Release [54], with the goal of running on products that “require lower current consumption, lower complexity, and lower cost” than Bluetooth Basic Rate / Bluetooth Enhanced Data Rate (BR/EDR). Gomez et al. [75] demonstrated that a BLE device equipped with a coin cell battery can work for at most 14.1 years theoretically, proving that BLE is well suited to energy-limited wearables. Li et al. [76] proposed a new LoRa control system known as *DyLoRa*, which can optimize the energy efficiency of the LoRa protocol by dynamically adjusting parameters such as spreading factor and transmission power. Experiment results showed that *DyLoRa* achieved a 41.2% lower energy consumption than the state-of-the-art approach.

All the efforts mentioned above, however, require wearables to modulate data into a self-generated carrier signal. In contrast, in backscatter communication, the end device could modulate its reflections of an existing radio frequency (RF) signal which is generated by other devices. In traditional backscatter communication (*e.g.*, radio-frequency identification (RFID)), a special-purpose controller (*e.g.*, an RFID reader) transmits an RF signal to the end device (typically known as a backscatter tag), and the end device simply “modulates the reflection coefficient of its antenna.” [81] Liu et al. [77] proposed to use ambient RF signals instead, *e.g.*, TV signals, cellular signals, *etc.* In this manner, two backscatter-

Table 5
Comparison of low-power communication technologies.

Technology	Source of the carrier signal	Power consumption		Reference
		Results	Evaluation setup	
X-MAC	Sender	1–2 mA	One Receiver & At Most 9 Senders; Under no Contention	[74]
Bluetooth Low Energy	Sender	0.01–10 mA	Under the Attribute Protocol (ATT) One-way Communication; *connSlaveLatency* = 0; *connInterval* Ranging from 7.5 to 4000 ms	[75]
DyLoRa	Sender	200–500 mW	One LoRa Gateway & 11 LoRa nodes	[76]
Ambient Backscatter	Ambient Carriers	Not Reported (Battery-free Prototype Integrating a Radio Frequency Energy Harvester)	–	[77]
Wi-Fi Backscatter	Ambient Wi-Fi Carriers	0.65 μW for the Transmit Circuit; 9.0 μW for the Receiver Circuit	–	[78]
Frequency-shifted Backscatter	Ambient Wi-Fi or Bluetooth Carriers	45 μW; 1100 bits/μJ	–	[79]

based devices could communicate with each other without using a specialized controller. Huang et al. [82] proposed a battery-free wearable system in the form-factor of shoes. Energy harvesters are placed inside both the left shoe and the right shoe. Due to the limited amount of energy that could be collected by wearables, it is infeasible to transmit harvested power from one shoe to another. Therefore, it is not trivial to utilize the energy from both shoes. The authors instead assigned different tasks to different shoes, *i.e.,* one shoe for energy-intensive sensing, and the other shoe for energy-intensive Bluetooth communication with the smartphone. They chose the energy-saving ambient backscatter for data synchronization across two shoes. Kellogg et al. [78], however, noticed that since ambient backscatter could only enable communication among backscatter-based devices, it actually creates an isolated network that does not belong to the Internet. They solved this problem by proposing the *WiFi Backscatter* model, where Wi-Fi signals rather than other ambient signals are used for modulation. A WiFi Backscatter tag could transmit a “1” bit or a “0” bit to a Wi-Fi device, through “either reflecting or absorbing the Wi-Fi packets” received by the Wi-Fi device. Zhang et al. [79] pointed out that both ambient backscatter and *WiFi Backscatter* have some severe drawbacks. The TV carrier signal used in ambient backscatter is not always available, and *WiFi Backscatter* presents low resistance to noise and mobility-induced dynamics. The authors designed a backscatter tag that shifts an existing Wi-Fi or Bluetooth carrier to a clean frequency band. Information is transmitted by the tag through on-off keying (OOK). The advantages of this frequency shifting mechanism are two-fold. *First,* the receiver only needs to focus on the shifted and relatively clean band. In contrast, in many previous backscatter technologies, the backscattered signal and the carrier signal reside in the same channel, leading to a high SNR while extracting the backscattered signal. *Second,* different from ambient backscatter whose primary carrier presents an unknown structure, frequency-shifted backscatter could utilize the structure of the primary carrier, such as the fixed preamble of a Wi-Fi or Bluetooth packet, to which modern radio chipsets are designed with high sensitivity. The authors demonstrated that frequency-shifted backscatter could work within a longer range of 4.8 m distance with up to 48.7 kbps throughput, at a power budget as low as 45 μW.

3.4. Security

In this section, we review the security issues by discussing several common security threats as well as corresponding defensive mechanisms. Here we focus on three kinds of security threats: (a) threats to data confidentiality; (b) threats to authentication; and (c) threats to availability. We list surveyed works in Table 6.

3.4.1. Threats to data confidentiality

Data confidentiality means “the protection of data from unauthorized access and disclosure.” [83] Since wearables are widely used to collect private physiological data for health monitoring and run privacy-sensitive applications like e-payment, it is of high interest for attackers to destroy data confidentiality.

1) Attacks on Data Confidentiality

   Data confidentiality can be compromised on the local device, on the edge gateway, on the central cloud, or during the transmission. We cover two kinds of attacks as follows:

   a) Eavesdropping Attacks

      Eavesdropping attacks occur during transmission and aim to eavesdrop on encrypted data that is transmitted between wearables and edge gateways. Since the secret key is the foundation of data encryption, one intuitive method of ruining data confidentiality during transmission is to break the key.

      Some researchers tried to exploit protocol-level vulnerabilities in the key establishment phase to perform an attack. Many wearables use the BLE protocol as their main communication protocol to transmit data to ambient devices. Unfortunately, the old versions of the BLE protocol have several well-known protocol-level flaws that provide the entry point for eavesdropping attacks. Many existing devices are using these old versions rather than the newer ones that already fix these vulnerabilities, *e.g.,* over 4 billion devices are estimated to use outdated and insecure BLE 4.0 or 4.1 in 2018 [116]. Filizolla et al. [84] noticed that some old versions of the BLE protocol (*e.g.,* the BLE 4.0, and the BLE 4.1) adopt the insecure LE Legacy Pairing protocol to exchange a secret Long Term Key (LTK) for data encryption with the help of a Temporary key (TK). Through a brute-force enumeration, attackers can easily obtain the TK, then use the TK to further break the LTK and get all encrypted data exchanged along the channel. Furthermore, some open-source tools, *e.g.,* Crackle [85], are available to utilize such vulnerability and sniff BLE connections.

   b) Inference Attacks Targeting Machine Learning Models

      Many wearable applications now use machine learning to analyze the sensed data [5,12,61]. However, the wide use of

Table 6
Papers related to security issues in edge computing for wearable technology.

Attack type	Threat type			Research content		Data source
	Threat to confidentiality	Threat to authentication	Threat to availability	Attack method	Defensive measure	
Eavesdropping Attacks	√			√		[84,85]
					√	[86–89]
Inference Attacks Targeting Machine Learning Models	√			√		[90–94]
					√	[95–97]
Dictionary Attacks		√		√		[98]
					√	[99–103]
Stealing Authentication Credentials		√		√		[104,105]
					√	[99–103,106,107]
Breaking the Authentication Key		√		√		[108]
Distributed Denial-of-Service (DDoS) Attacks			√	√		[109]
					√	[110–114]
Interference on the Communication Link			√	√		[115]

machine learning also brings much worry about data confidentiality. There exist many works focusing on inference attacks that seek to infer sensitive information related to the training samples of deep learning models. Two typical kinds of such attacks are Membership Inference Attacks (MIAs) and property inference attacks [117].

MIAs seek to infer if a specific data record belongs to the training set of a trained model, which could compromise the confidentiality of membership information. Shokri et al. [91] performed MIAs based on the black-box setting, where only the prediction value from the target machine learning model and the true class label are required to infer if the target data record is a member of the training set. Nasr et al. [90] further extended black-box MIAs to the white-box setting, where they used the gradient vector with respect to the target data point as the main input to the attack model. In addition to the stand-alone learning setting, they also adapted their attack method to the federated learning setting. Experiment results demonstrated that both an adversarial participant and a malicious central parameter server could achieve an impressive accuracy of membership inference using the proposed method.

As for property inference attacks, they try to infer characteristics of the data records in the training set. Ganju et al. [94] tried to compromise the global properties of the training set for white-box Fully Connected Neural Networks (FCNNs), such as the relative attractiveness of the individuals in the training set for a smiley face classifier, or the fraction of training records from a certain class. Melis et al. [93] proposed an attack method which could be used by adversarial participants in the federated learning setting to "infer properties of other participants' training data that are not true of the class as a whole, or even independent of the features that characterize the classes of the joint model." Wang et al. [92] incorporated a Generative Adversarial Network (GAN) with a multi-task discriminator to reconstruct training samples of the victim user based on a malicious central parameter server in the federated learning setting.

The above works demonstrated that private user data of wearable applications could be compromised by malicious edge gateways (or cloud servers) if it is used to train or fine-tune the machine learning model deployed on the edge gateway layer (or on the central cloud layer). Besides, other end devices could also infer such private user data in the federated learning setting through various attack methods.

2) Effects of Compromised Data Confidentiality

Even some attackers may only seek access to sensitive data out of curiosity, the compromised data confidentiality still undoubtedly undermines customers' trust in wearable devices. Actually, many attackers do not stop at satisfying their curiosity. They may use leaked private data to make profits, *e.g.*, selling private health data to insurance companies. They can even further analyze leaked data to obtain some more sensitive information and thus cause greater loss of victims. For example, if data sensed by a wearable accelerometer, magnetometer, or gyroscope sensors are exposed to attackers, then attackers can exploit it to infer the keystroke on mobile devices. Some researchers [118–120] exploited motion data captured by wearable devices to infer the keystroke on mobile devices with prior knowledge about some context information, *e.g.*, keyboard size, and posture of keypad plane. Liu et al. [121] took one step further by performing such inference without any context-aware requirements. A top-5 successful attacking rate as high as 94% is observed in their experiment. In addition to the keystroke on mobile devices, other private user information is also put at risk by this kind of attack. Maiti et al. [122] demonstrated that mechanical lock combinations could be inferred by analyzing smartwatches' gyroscope sensor data.

3) Possible Solutions

Due to the lack of the public key infrastructure (PKI) in typical usage scenarios of wearables, and the high computation overhead of public-key cryptography (PKC), it is infeasible for many wearables to utilize PKC against eavesdropping attacks. An alternative approach is to utilize a signal commonly accessible to two communication parties for a key generation or key exchange.

In key generation technologies, two communicating devices could independently sample from the commonly accessible signal, and then use the recorded data as materials to generate keys. Since the measurement values from the two sides may have some difference, technologies such as error correction codes are required to bridge this gap and guarantee the same key. Xu et al. [88] gave a comprehensive survey of existing key generation systems for IoT and classified them based on the required hardware. Typically, there are six categories of required hardware, *i.e.*, radio module, microphone, inertial measurement unit (IMU) sensor, miscellaneous hardware, camera, and hybrid hardware. We encourage interested readers to read this survey [88] for further understanding.

A key exchange based on commonly accessible signals is another promising way to defend against eavesdropping attacks. It could be enforced by hiding the exchanged secret key with the aid of the fuzzy vault technique or using out of band channels. In the fuzzy vault, the sender mixes the exchanged secret key S together with a set of values A. To obtain S from the mixture of S and A, the receiver needs to use another set of values B that share many common values with A. To utilize fuzzy vault for communication between wearables and edge gateways, A and B should be obtained from the measurement of a bilaterally accessible signal such as the accelerometer signal [87]. As for out-of-band key exchange, private edge gateways and wearables could use out-of-band signals, *e.g.*, the vibration signal [86], to exchange the shared secret key.

To avoid the overhead of encrypting every message after agreeing on the same key, but still, be able to defend against eavesdroppers, one alternative solution is to use jamming techniques. Shen et al. [89] proposed a jamming-based method known as *Ally Friendly Jamming*, which can disable the wireless communication channels of unauthorized devices but still enable wireless connectivity of authorized devices. *Ally Friendly Jamming* generates jamming signals with a shared secret key and emits them at the same time as original signals, so that unauthorized receivers, who are agnostic of the shared key, could not recover original signals from the mixture of two signals. As for authorized receivers, they can regenerate the jamming signals and then subtract them from the received signals to recover original signals.

To defend against inference attacks targeting machine learning models, we suggest two defensive measures as follows.

First, we can restrict access to machine learning models. Shokri et al. [91] demonstrated that by only exposing the label of the most likely class, rather than the confidence values for all classes, the black-box MIA accuracy could be decreased by 10%–26%. As for the white-box MIA, hardware isolation techniques could be utilized to reduce the amount of information available to attackers. Tramèr et al. [95] noticed that for machine learning models deployed in the central cloud layer, Trusted Execution Environments (TEEs), such as Intel SGX [123], ARM TrustZone [124], *etc*, could be exploited to isolate sensitive computations of machine learning from the untrusted part of the operating system. Nevertheless, these isolation techniques are currently limited to CPUs and do not adapt to Graphics Processing Units (GPUs) which gain much popularity in machine learning-based services these days due to their strong parallel-processing capability. To utilize available GPU resources, as well as guarantee the computation integrity and user privacy, they proposed to outsource all linear layers, *e.g.*, convolution layers, dense layers, *etc*, from the TEE-based CPU to the untrusted GPU with the help of cryptographic protocols. Mo et al. [96] also utilized hardware isolation techniques to improve security. They proposed to partition a Deep Neural Network (DNN) model into two parts, *i.e.*, the "shallow" part containing the first few layers, and the "deep" part following the "shallow" part. The "shallow" part executes inside the TEE device to improve security, while the "deep" part executes in the untrusted part of the operating system to reduce overhead. Experiment results demonstrated that white-box MIAs can be defended effectively even if the "shallow" part contains only the first layer of the DNN model, with negligible (only 3%) overhead of CPU time and energy consumption.

Second, we can adjust the training process to defend against inference attacks. A trained model may remember some unique characteristics of training samples that are not generic among other samples from the same class. Considering this overfitting problem being widely exploited to perform various inference attacks, a promising way to defend against these attacks is to alleviate such overfitting. Nasr et al. [97] adapted the training process of machine learning models to a min-max game, which seeks to train the model to be accurate as well as have the strongest resistance to the strongest black-box MIAs. Experiment results demonstrated that the adversarial training process could effectively defend against black-box MIAs by alleviating overfitting, and significantly reducing the gap between training and testing accuracies.

3.4.2. *Threats to authentication*

Authentication means that "both the sender and receiver should be able to confirm the identity of the other party involved in the communication to confirm that the other party is indeed who or what they claim to be." [125] Authentication mechanisms are widely used to determine the identities of connected devices. Through bypassing the authentication mechanism, attackers can disguise themselves as licensed wearables or as licensed edge gateways and perform some over-privileged operations.

1) Attacks on Authentication

a) Dictionary Attacks

The most intuitive attack on authentication mechanisms is a dictionary attack, in which an attacker tries to match the weak password used for identity authentication in a brute-force manner. The terminology "dictionary" refers to a set of commonly used passwords owned by the attacker. Many dictionaries are publicly available now [98], lowering the barrier for launching such an attack.

b) Stealing Authentication Credentials

Attackers can also exploit protocol-level flaws to obtain authentication credentials by deception. Cassola et al. [104] presented a variant of an evil twin attack on WPA Enterprise networks. By exploiting flaws in the authentication phase, it compromised the authentication credentials used by the victim, with which an authenticated connection to the legitimate network using the victim's identity could be established. Wang et al. [105] performed a man-in-the-middle attack against the BLE protocol. In detail, they deployed a malicious device M that interrupted an already established connection between a peripheral BLE device P and a central BLE device C. The device M then connected to device P by the *JustWorks* pairing method. To obtain the authentication credentials required by the device P, the device M disguised itself as the device P and accepted the connection from device C. The deceived device C would send the device P's authentication credentials to the device M. Finally, device M could send the authentication credentials to the device P and successfully bypassed the authentication mechanism of BLE.

c) Breaking the Authentication Key

Some end devices only use a hard-coded key to authenticate the other party involved in the communication. For example, Ronen et al. [108] exploited reverse engineering to analyze the security mechanisms of the Phillips Hue smart lamps. They discovered that these lamps simply use a hard-coded key to authenticate the received firmware updates. By breaking the same global key used for all the lamps of these product types via side-channel attacks, they bypassed the authentication mechanisms of these lamps successfully.

2) Effects of Compromised Authentication

Through breaking the authentication mechanism, attackers can disguise themselves and perform a series of follow-up attacks. A typical example of these follow-up attacks is the firmware modification attack. A wearable device without direct access to the Internet may rely on edge gateways to first download the latest firmware from the Internet and then send the firmware to it via other supported communication protocols such as BLE. This process has a severe vulnerability: if a malicious firmware update application is installed on an infected gateway and successfully bypasses the authentication mechanism somehow, the wearable device will be totally destroyed. Shim et al. [126] exploited vulnerabilities in a fitness tracker's firmware modification process to perform such an attack. They reverse-engineered the gateway app and the target tracker's firmware, finding out that no authentication is required after the provider is paired to the wearable device. Then they built a fake gateway on a PC and successfully used the fake gateway to trigger a firmware update with a tampered firmware. They used code obfuscation and integrity verification of firmware to mitigate such vulnerabilities. Ronen et al. [108] reverse engineered the process used to trigger a firmware update on the Phillips Hue smart lamps and discovered that the lamps utilized AES-CCM to encrypt/decrypt and authenticate every firmware update. By conducting side-channel attacks exploiting the power consumption, they successfully compromised the AES-CCM master key and could trigger arbitrary firmware updates with the broken key.

3) Possible Solutions

To defend against dictionary attacks, some unconventional authentication credentials could be used in combination with traditional token-based authentication credentials, to form the so-called two-factor authentication. Besides, manufacturers can also substitute traditional authentication credentials with unconventional ones directly, which could possibly improve the convenience or lower the overhead of authentication. In general, there are three kinds of unconventional authentication techniques, *i.e.,* per-person credentials, per-device credentials, and environment-based credentials. Per-person credentials authenticate the identity of users by checking biometrics such as human fingerprint [99], face images [100], physical characteristics of touching fingers [102], *etc.* Per-device credentials instead authenticate the identity of the device, regardless of the identity of the user who is operating the device, or the environment where the device is placed. The basic principle behind these techniques is that different devices could be distinguished by device-specific characteristics, such as software-based characteristics (*e.g.,* due to the complexity of IEEE 802.11 standards, different software-level implementations differ in characteristics including MAC layer behavior, frame sequence numbers, and traffic pattern), hardware-based characteristics (*e.g.,* clocks on different devices have different offset values with respect to the reference clock, so the offset values can be used to identify different devices), *etc.* For a more detailed introduction to software-based and hardware-based per-device authentication, interested readers could refer to [103]. As for environment-based credentials, they could be exploited to compare the ambient environment perceived by wearables and edge gateways. This kind of credentials applies to usage scenarios where a proximate device is highly likely to be an authorized device. As we have discussed earlier in Section 3.4.1, there are various sensors capable of capturing the characteristics of the ambient environment that could be used for key generation, such as radio sensors, audio sensors, *etc.* Obviously, information collected by these sensors could also be used to generate authentication credentials. For example, Karapanos et al. [101] proposed to compare ambient noise perceived by two devices. Two devices sharing similar ambient noise are considered to be in proximity to each other. The original usage scenario of this noise-based authentication is to support safe account login without the need of copying any Short Messaging Service (SMS) verification code from the phone to other devices, but we can adapt this method to those wearable devices that only expect connections from nearby edge gateways.

For attacks that steal authentication credentials by deception, we suggest revising authentication protocols [106,107]. Device manufacturers can also use the aforementioned unconventional authentication credentials, *e.g.,* use fingerprint data [99] that is hard to steal, or use two-factor authentication to enhance security.

For attacks that break the authentication key, we advise using dynamic software-based authentication keys rather than fixed hard-coded keys.

3.4.3. Threats to availability

1) Attacks on Availability

a) Distributed Denial-of-service (DDoS) Attacks

According to the definitions given by the U.S. Cybersecurity and Infrastructure Security Agency (CISA), "a denial-of-service (DoS) attack occurs when legitimate users are unable to access information systems, devices, or other network resources due to the actions of a malicious cyber threat actor," and "a distributed denial-of-service (DDoS) attack occurs when multiple machines are operating together to attack one target." [127]

IoT devices are usually weaker than general-purpose computers and thus easier to be compromised and turned into "bots," *i.e.,* compromised devices used for launching DDoS attacks. A large network of "bots" is known as a "botnet," which may overwhelm victim servers if its scale is large enough. A famous example of IoT-based botnets is the "Mirai" botnet, whose initial attack on KrebsOnSecurity.com reached up to 665 Gbps, being one of the biggest assaults the Internet has ever witnessed [128]. According to Antonakakis [129], Mirai infections peaked at 60k in their seven-month analysis. All these surprising facts demonstrated the weakness of IoT devices as well as the necessity to pay attention to IoT-based DDoS.

According to Vishwakarma [109], DDoS attacks in the IoT network can be divided into three types, *i.e.,* application-layer attacks, infrastructure layer attacks, and zero-day DDoS attacks. Application layer attacks refer to attacks occurring in the application layer, such as HTTP flooding that exhausts victim servers' processing capability via a flood of HTTP requests. In contrast, infrastructure layer attacks tend to exhaust victim servers' outgoing and incoming bandwidth as well as internal resources by exploiting weakness at the transport or network layer. For example, SYN flooding initiates a TCP connection to a target server by sending an SYN packet with a spoofed IP address, making the server wait for a confirmation ACK that will never occur. As for zero-day attacks, attackers need to find a zero-day vulnerability in the code running on the victim server. The modifier "zero-day" means that the attack is performed at the "zero" day since the discovery of the vulnerability when the vendor or developer may be unaware of the bug or do not have enough time to release a patch. A famous example of zero-day attacks is the ZDI-20-709 vulnerability which "allows network-adjacent attackers to execute arbitrary code on affected installations of NETGEAR R6700 routers." [130] Specifically, the vulnerability was reported to the vendor by

Zero Day Initiative (ZDI) on February 5th, 2020. The vendor requested an extension of public release until the end of June to ZDI, but the request was rejected and the vulnerability was published online as a zero-day vulnerability on June 15th, 2020.

b) Interference on the Communication Link

In addition to DDoS attacks, another kind of threat to availability is interference on the communication link. Different from DDoS attacks, attacks based on interference on the communication link do not require the presence of a distributed cluster of compromised IoT devices and is easier to be launched. However, targets of DDoS attacks range from edge gateways to any other accessible servers, while interference on the communication link can only injure devices in the vicinity of the attacker.

Most wearables utilize vulnerable BLE as their primary communication protocol, and it is easier to congest a BLE channel than to create a "botnet" for launching DDoS attacks. Goyal et al. [115] successfully launched such a kind of attack on the Fitbit Charge tracker, which uses BLE to sync its sensor data with the corresponding mobile application on the smartphone. They observed that the Fitbit tracker keeps advertising its custom characteristics even when it is already paired, and do not follow the BLE guidelines to use dynamic device address. Then they used GattTool to connect to the paired Fitbit tracker, keeping it too busy to sync its sensor data with the paired device by continually asking its characteristics. They also launched a similar attack on an unpaired Fitbit tracker and prevented legitimate devices from pairing with the tracker.

2) Effects of Compromised Availability

It is straightforward that user experience will be disrupted when a DDoS attack occurs. When a service provider becomes the target of a DDoS attack, it may not be able to distinguish normal user requests from malicious DDoS requests. As malicious traffic exceeds the service provider's maximum load, normal user requests will not be handled by the paralyzed service provider as expected, leading to the damage of user experience.

From the perspective of service providers, compromised availability may do harm to their profits, since consumers may turn to other service providers for a better user experience. Besides, it will increase the cost of providing services, since more resources are required to provide high-quality services to benign users while serving undetected malicious requests at the same time. According to a recent report [131], about 5% of monthly gaming-related traffic is due to DDoS attacks. This proportion can get as high as 30% when a DDoS attack is being performed, greatly increasing the operating costs of gaming service providers. Although wearable devices may occupy only a small proportion of bots today, it is still very important to keep our eye on wearable-based DDoS considering the potential popularity of more powerful wearables in the future.

3) Possible Solutions

Since DDoS attacks involve a large collection of compromised "bots," we need to capture enough real-world attacks for a better understanding of the characteristics of DDoS attacks (*e.g.,* the most frequently used methods to turn IoT devices into "bots") before adopting preventive measures. Dang et al. [114] deployed the HoneyCloud system consisting of hundreds of IoT honeypots, *i.e.,* "security resource whose value lies in being probed, attacked, or compromised," [132] across the world for a whole year to capture real-world IoT attacks. HoneyCloud consists of both hardware IoT honeypots and cloud-based virtual honeypots. After recording and analyzing 264 million suspicious connections and 28 million effective attacks, they concluded that a typical attack against IoT devices consists of three successive phases – intrusion, infection, and typical monetization. In the intrusion phase, most attackers launch a brute-force dictionary attack, while few choose to exploit security flaws to bypass the authentication mechanism instead. Only after finishing the intrusion phase successfully can attackers control the compromised device and launch various follow-up attacks including denial-of-service, Telnet/SSH scan, *etc*. From the findings of HoneyCloud, we argue that developers should enhance wearables' authentication mechanism to prevent them from being turned into "bots."

As for service providers, per-packet DDoS detection [111] and statistics-based DDoS detection [110] should be exploited to defend against DDoS attacks. For zero-day vulnerabilities, static code analysis [112] and firmware-based memory checking [113] can be utilized to identify them.

To avoid interference on the communication link exploiting static BLE address, we advise paying more attention to system security and follow the security guidelines of related protocols.

4. Future research directions

To make wearable devices benefit more from edge computing, there are some research directions worth exploring. Here we list four potential research directions, each of which corresponds to one of the four research issues discussed in Fig. 2.

4.1. Adapting scheduling schemes to usage scenarios of wearables

Existing works related to scheduling in edge computing tend to assume simple scenarios and ignore many realistic factors, *e.g.,* coexistence of offloaded data and conventional data, hierarchical placement of edge gateways, real-time variations of available edge computing resources, evaluation in real networks, *etc*, which could lead to sub-optimal offloading decisions in actual use [22]. This problem could be further exacerbated in edge computing for wearable technology due to several reasons. *First,* many works do not consider user mobility when designing scheduling schemes, which is an unavoidable problem with wearables moving with users. User mobility could cause fluctuation of the network condition, which may affect the benefit of different scheduling schemes. In some cases, handoffs, *i.e.,* the processes of migrating offloaded tasks from one edge gateway to another, could occur with moving users. The handoff process could not only affect the network condition and available resources provided by edge gateways but also incur a high handoff delay. New technologies are desired to tackle all these challenges caused by user mobility. For example, pre-migration techniques based on user mobility prediction could be adopted to alleviate the effect of the handoff delay. *Second,* different wearable applications on a single wearable device may rely on the same sensor data, *e.g.,* both fall detection and activity recognition may require data collected by the accelerometer sensor. Existing scheduling schemes designed for a single application simply instruct these correlative applications to offload tasks separately, ignoring that the same information may be transferred to edge gateways by these applications repeatedly. Such strategies could waste network bandwidth and extend the overall transmission delay. Nevertheless, scheduling these applications in a unified manner faces several challenges: (a) different applications may do different local preprocessing on the same raw data before transferring it to edge gateways; and (b) each application may only transfer data challenging to process for this specific application. More research efforts are required to tackle these challenges.

Besides, edge computing for wearable technology differs from other kinds of edge computing systems in many aspects. Unlike other less constrained mobile devices, most COTS wearable devices now only support the short-range Bluetooth protocol and mainly rely on powerful private edge gateways rather than public edge servers. By exploiting the unique features of private edge gateways compared to public edge servers, and the characteristics of those communication protocols widely adopted by wearables, researchers could further optimize scheduling schemes for wearables-oriented edge computing. Examples include the aforementioned CATO mechanism [46] optimized for the ARM big.LITTLE architecture which is popular in modern smartphones, and the WearDrive system [17] combining BLE and WiFi-Direct for energy saving.

4.2. Enhancing personalization of wearable applications

Many wearable applications use machine learning models to extract high-level information from sensed raw data. Since different users have different characteristics in their physiological signals, the optimal model parameters for different users may differ. Therefore, the common strategy of using the same model for all users may not achieve the best accuracy for each user. Personalized models are desired by wearable applications to provide services with higher quality.

Recent works [133,134] focus on personalization. Despite their excellent performance, all these works solicit some labels from users, which however limits the usage of these personalization techniques in usage scenarios that lack enough labeled data. For example, this problem could become very severe for wearable applications predicting the seizure of fatal diseases [10,61] for two reasons. *First,* since the number of seizures for an individual user is typically very limited, only insufficient data could be provided to existing supervised or semi-supervised personalization techniques, which could degrade the performance of these methods. *Second,* since the accuracies of such applications can have a great impact on users' safety, personalization techniques are desired to achieve higher accuracy. In summary, it is necessary to propose new personalization techniques that can work well for applications that lack sufficient labeled data from the individual user.

4.3. Improving emerging energy sources for wearables

Orthogonal to optimizing wearables' energy consumption, exploiting diverse energy sources for wearables is another promising way to extend the battery life of wearables. There are typically three energy sources for wearables.

- ***First,*** users can carry a charger with them, and they need to find a socket whenever they want to charge their wearables. This common method is unquestionably inconvenient for users.
- ***Second,*** some high-end edge gateways now support reverse wireless charging, *i.e.*, they can charge other electronic devices, including wearables that support wireless charging [135,136]. This method has the drawbacks of low energy conversion efficiency (*e.g.*, a full charge for the Samsung Galaxy Watch Active (230 mAh) could gobble 29% of the Galaxy S10e battery (3100 mAh) [137]), low charging speed (*e.g.*, the power of reverse wireless charging provided by many high-end smartphones is typically less than 5W [138]), and the need for extra components (*e.g.*, the inductive coil, the controller of the coil [135]) in compact wearables.
- ***Third,*** energy harvesting technologies can be exploited to collect power from the ambient environment. There exist various sources of energy that could be harvested by wearables, including thermal energy [139], kinetic energy [68,69,82], solar energy [140], RF energy [141], *etc*. The drawback of energy harvesting technologies is that the harvested energy could be unstable (*e.g.*, the solar power, a common source of wearable energy harvesting, could be 100 times slower on a cloudy day than on a sunny day) and weak (typically in the order of a few μW to mW) [142].

We think that more works need to be done to overcome the latter two energy sources' drawbacks.

4.4. Promoting non-cryptographic security techniques for wearables

Security mechanisms based on cryptography may incur a great overhead on resource-limited wearables. Non-cryptographic security techniques are desired to reduce such overhead as well as maintain security. We have discussed some non-cryptographic security techniques in Section 3.4, including novel authentication credentials, jamming techniques, *etc*. Nevertheless, despite their applicability to wearables, in theory, they may not be practical in today's COTS wearables for several reasons.

First, as Zeng et al. [103] have pointed out, per-device authentication credentials are limited by the "tradeoff between the detection rate and false alarm rate." Besides, channel/location-based fingerprinting, which is superior in terms of uniqueness, adaptiveness, and mimicking resistance, may not apply well to wearables, since user mobility could greatly change channel/location-based characteristics.

Second, existing per-person authentication credentials are mainly designed for those more powerful mobile devices rather than resource-limited wearables. Some of them require specialized sensors, limiting their usage in compact wearables. For example, fingerprint-based authentication [99] requires onboard fingerprint scanners. And face recognition-based authentication [100] requires onboard cameras to capture face images. We think more practical per-person authentication credentials should be further considered to achieve authentication on wearables.

Third, jamming techniques, which eliminate the need of encrypting every message after pairing, still have some severe drawbacks. For example, *Ally Friendly Jamming*, which is introduced in Section 3.4.1, could not resist adversarial jammers.

In summary, despite the advantage of lower computation complexity, existing non-cryptographic security techniques still have some problems that need to be solved. Some drawbacks are derived from the methods themselves, while others are caused by resource limitations of wearables, *e.g.*, per-person credentials are mainly designed for more powerful mobile devices rather than wearables. We call on researchers to aim more research efforts towards promoting non-cryptographic security techniques that are suitable to wearables.

5. Conclusion

Wearable devices are gaining popularity in the market with their distinctive ability to sense and interact with the human body directly. Nevertheless, they typically have only limited local resources due to the constraints on size and weight. As an emerging computing model, edge computing has a great potential to enhance wearable devices by providing sufficient resources to wearables with little transmission delay.

Considering the key role of edge computing in next-generation wearable technology, we conduct a comprehensive survey of recent works on edge computing for wearable technology. We group the surveyed works into four research issues, *i.e.*, scheduling, information perception, power saving, and security. After investigating existing works, we propose several open research issues in edge computing for wearable technology, including wearable-oriented scheduling, personalization, emerging energy sources, and non-cryptographic security techniques. In summary, this article provides a detailed reference for wearable device manufacturers and researchers on how to maximize the practical values and alleviate security issues of edge computing for wearable technology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study is supported in part by the National Science Foundation of the US under Grant NSF1919154, and the China Postdoctoral Science Foundation under Grant 2019M650685.

References

[1] Worldwide Wearables Market Forecast to Maintain Double-Digit Growth in 2020 and Through 2024, According to IDC, https://www.idc.com/getdoc.jsp?containerId=prUS46885820. (Accessed 30 January 2021).
[2] Mi Smart Band 5, https://www.mi.com/global/mi-smart-band-5/specs/. (Accessed 30 January 2021).
[3] Apple Watch Series 6 – Technical Specifications, https://support.apple.com/kb/SP826. (Accessed 30 January 2021).
[4] T. de Quadros, A.E. Lazzaretti, F.K. Schneider, A movement decomposition and machine learning-based fall detection system using wrist wearable device, IEEE Sens. J. 18 (12) (2018) 5082–5089.
[5] T.R. Mauldin, M.E. Canby, V. Metsis, A.H.H. Ngu, C.C. Rivera, SmartFall: a smartwatch-based fall detection system using Deep Learning, Sensors 18 (10) (2018) 3363.
[6] A.J. Ben Ali, Z.S. Hashemifar, K. Dantu, Edge-SLAM: edge-assisted visual simultaneous localization and mapping, in: Proceedings of the 2020 ACM MobiSys, ACM, New York, NY, USA, 2020, pp. 325–337.
[7] H.T. Dinh, C. Lee, D. Niyato, P. Wang, A survey of mobile cloud computing: architecture, applications, and approaches, Wirel. Commun. Mob. Comput. 13 (18) (2013) 1587–1611.
[8] Y. Kim, Mobile terminal and photo searching method thereof, https://patents.google.com/patent/US9043318B2, 2015. (Accessed 3 May 2021).
[9] A. Belpaire, A. Natu, Cloud based payment method, https://patents.google.com/patent/US9672513B2, 2017. (Accessed 3 May 2021).
[10] M. Gusev, A. Stojmenski, A. Guseva, ECGalert: a heart attack alerting system, in: D. Trajanov, V. Bakeva (Eds.), Proceedings of the 2017 International Conference on ICT Innovations, Springer, Cham, Switzerland, 2017, pp. 27–36.
[11] F. Concone, G.L. Re, M. Morana, A fog-based application for human activity recognition using personal smart devices, ACM Trans. Internet Technol. 19 (2) (2019) 1–20.
[12] F. Samie, L. Bauer, J. Henkel, Hierarchical classification for constrained IoT devices: a case study on human activity recognition, IEEE Int. Things J. 7 (9) (2020) 8287–8295.
[13] ETSI GS MEC 001 – V2.1.1, https://www.etsi.org/deliver/etsi_gs/MEC/001_099/001/02.01.01_60/gs_MEC001v020101p.pdf. (Accessed 30 January 2021).
[14] A. Mathur, N.D. Lane, S. Bhattacharya, A. Boran, C. Forlivesi, F. Kawsar, DeepEye: resource efficient local execution of multiple deep vision models using wearable commodity hardware, in: Proceedings of the 2017 ACM MobiSys, ACM, New York, NY, USA, 2017, pp. 68–81.
[15] M. Whaiduzzaman, M.R. Hossain, A.R. Shovon, S. Roy, A. Laszka, R. Buyya, A. Barros, A Privacy-Preserving Mobile and Fog computing framework to trace and prevent COVID-19 community transmission, IEEE J. Biomed. Health Inform. 24 (12) (2020) 3564–3575.
[16] Q. Yuan, H. Zhou, J. Li, Z. Liu, F. Yang, X.S. Shen, Toward efficient content delivery for automated driving services: an edge computing solution, IEEE Netw. 32 (1) (2018) 80–86.
[17] J. Huang, A. Badam, R. Chandra, E.B. Nightingale, WearDrive: fast and energy-efficient storage for wearables, in: Proceedings of the 2015 USENIX ATC, USENIX Association, Santa Clara, CA, USA, 2015, pp. 613–625.
[18] D. Amiri, A. Anzanpour, I. Azimi, A.-M. Rahmani, P. Liljeberg, N.D. Dutt, M. Levorato, Optimizing energy efficiency of wearable sensors using fog-assisted control, in: Fog Computing, John Wiley & Sons, Ltd, 2020, pp. 245–268.
[19] A. Anzanpour, H. Rashid, A.M. Rahmani, A. Jantsch, N.D. Dutt, P. Liljeberg, Energy-efficient and reliable wearable Internet-of-Things through fog-assisted dynamic goal management, Proc. Comput. Sci. 151 (2019) 493–500.
[20] S. Seneviratne, Y. Hu, T. Nguyen, G. Lan, S. Khalifa, K. Thilakarathna, M. Hassan, A. Seneviratne, A survey of wearable devices and challenges, IEEE Commun. Surv. Tutor. 19 (4) (2017) 2573–2620.
[21] M. Ghamari, B. Janko, R.S. Sherratt, W. Harwin, R. Piechockic, C. Soltanpur, A survey on wireless body area networks for eHealthcare systems in residential environments, Sensors 16 (6) (2016) 831.
[22] P. Mach, Z. Becvar, Mobile edge computing: a survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1628–1656.
[23] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile edge computing: a survey, IEEE Int. Things J. 5 (1) (2018) 450–465.
[24] Fitbit Charge 4 Advanced Fitness Tracker | Shop, https://www.fitbit.com/global/us/products/trackers/charge4. (Accessed 30 January 2021).
[25] Amazon.com: Fitbit Charge 4 Fitness and Activity Tracker, https://www.amazon.com/Fitbit-Fitness-Activity-Tracking-Included/dp/B084CQ41M2/. (Accessed 30 January 2021).
[26] Microsoft Hololens, https://www.microsoft.com/en-us/hololens. (Accessed 30 January 2021).
[27] Bose Frames Tempo, https://www.bose.com/en_us/products/frames/bose-frames-tempo.html. (Accessed 30 January 2021).
[28] Amazon.com: Echo Loop – Smart ring with Alexa, https://www.amazon.com/dp/B07JPK4XJ6. (Accessed 30 January 2021).
[29] AirPods Max – Technical Specifications – Apple, https://www.apple.com/airpods-max/specs/. (Accessed 30 January 2021).
[30] HUAWEI X GENTLE MONSTER Eyewear II Specifications, https://consumer.huawei.com/en/wearables/gentle-monster-eyewear2/specs/. (Accessed 30 January 2021).
[31] iPhone 12 Pro and 12 Pro Max – Technical Specifications – Apple, https://www.apple.com/iphone-12-pro/specs/. (Accessed 30 January 2021).
[32] MacBook Pro 13-inch – Technical Specifications – Apple, https://www.apple.com/macbook-pro-13/specs/. (Accessed 30 January 2021).
[33] J.A. Stankovic, T. He, Energy management in sensor networks, Philos. Trans. R. Soc. A 370 (2012) 52–67.
[34] The cost of low power, https://senseair.com/knowledge/sensor-technology/technology/the-cost-of-low-power/. (Accessed 30 January 2021).
[35] Hexoskin Smart Kit – Men's, https://www.hexoskin.com/collections/all/products/hexoskin-smart-kit-mens. (Accessed 4 February 2021).
[36] G. Ananthanarayanan, P. Bahl, P. Bodík, K. Chintalapudi, M. Philipose, L. Ravindranath, S. Sinha, Real-time video analytics: the Killer App for edge computing, Computer 50 (10) (2017) 58–67.
[37] H. Arasteh, V. Hosseinnezhad, V. Loia, A. Tommasetti, O. Troisi, M. Shafie-khah, P. Siano, Iot-based smart cities: a survey, in: Proceedings of the 2016 IEEE EEEIC, IEEE, New York, NY, USA, 2016, pp. 1–6.
[38] Homepod Support, https://support.apple.com/homepod, 2021. (Accessed 2 May 2021).
[39] Amazon.com: All-new Echo (4th Gen) | With premium sound, smart home hub, and Alexa | Charcoal, https://www.amazon.com/All-New-Echo-4th-Gen/dp/B07XKF5RM3. (Accessed 2 May 2021).
[40] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G.H. Xu, R. Netravali, Reducto: on-camera filtering for resource-efficient real-time video analytics, in: Proceedings of the 2020 ACM SIGCOMM, ACM, New York, NY, USA, 2020, pp. 359–376.
[41] A. Yousefpour, G. Ishigaki, R. Gour, J.P. Jue, On reducing IoT service delay via fog offloading, IEEE Int. Things J. 5 (2) (2018) 998–1010.
[42] K. Wang, Y. Tan, Z. Shao, S. Ci, Y. Yang, Learning-based task offloading for delay-sensitive applications in dynamic fog networks, IEEE Trans. Veh. Technol. 68 (11) (2019) 11399–11403.
[43] M. Al-Zinati, T. Almasri, M. Alsmirat, Y. Jararweh, Enabling multiple health security threats detection using mobile edge computing, Simul. Model. Pract. Theory 101 (2020) 101957.
[44] J. Zhang, X. Hu, Z. Ning, E.C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, B. Hu, Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks, IEEE Int. Things J. 5 (4) (2018) 2633–2645.
[45] H. Yin, X. Zhang, H.H. Liu, Y. Luo, C. Tian, S. Zhao, F. Li, Edge provisioning with flexible server placement, IEEE Trans. Parallel Distrib. Syst. 28 (4) (2017) 1031–1045.
[46] Y. Yang, Y. Geng, L. Qiu, W. Hu, G. Cao, Context-aware task offloading for wearable devices, in: Proceedings of the 2017 IEEE ICCCN, IEEE, New York, NY, USA, 2017, pp. 1–9.
[47] The NIST Definition of Cloud Computing, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf. (Accessed 30 January 2021).
[48] Global Infrastructure, https://aws.amazon.com/about-aws/global-infrastructure/. (Accessed 2 May 2021).
[49] S.K. Sood, I. Mahajan, A fog-based healthcare framework for Chikungunya, IEEE Int. Things J. 5 (2) (2018) 794–801.
[50] M. Jutila, E. Strömmer, M. Ervasti, M. Hillukkala, P. Karhula, J. Laitakari, Safety services for children: a wearable sensor vest with wireless charging, Pers. Ubiquitous Comput. 19 (5–6) (2015) 915–927.
[51] M. Chen, Y. Miao, Y. Hao, K. Hwang, Narrow band Internet of Things, IEEE Access 5 (2017) 20557–20577.
[52] S. Safaric, K. Malaric, ZigBee wireless standard, in: Proceedings of the 2006 IEEE ELMAR, IEEE, New York, NY, USA, 2006, pp. 259–262.
[53] S. Devalal, A. Karthikeyan, Lora technology – an overview, in: Proceedings of the 2018 IEEE ICECA, IEEE, New York, NY, USA, 2018, pp. 284–290.
[54] BLUETOOTH SPECIFICATION Version 4.0, https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=456433. (Accessed 30 January 2021).

[55] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu, W. Lv, Edge computing security: state of the art and challenges, Proc. IEEE 107 (8) (2019) 1608–1631.
[56] R. Wolski, S. Gurun, C. Krintz, D. Nurmi, Using bandwidth data to make computation offloading decisions, in: Proceedings of the 2008 IEEE IPDPS, IEEE, New York, NY, USA, 2008, pp. 1–8.
[57] Z. Cheng, P. Li, J. Wang, S. Guo, Just-in-time code offloading for wearable computing, IEEE Trans. Emerg. Top. Comput. 3 (1) (2015) 74–83.
[58] S. Wang, Y. Zhao, J. Xu, J. Yuan, C. Hsu, Edge server placement in mobile edge computing, J. Parallel Distrib. Comput. 127 (2019) 160–168.
[59] I. Ketykó, L. Kecskés, C. Nemes, L. Farkas, Multi-user computation offloading as Multiple Knapsack Problem for 5G Mobile Edge Computing, in: Proceedings of the 2016 IEEE EuCNC, IEEE, New York, NY, USA, 2016, pp. 225–229.
[60] X. Chen, L. Jiao, W. Li, X. Fu, Efficient multi-user computation offloading for mobile-edge cloud computing, IEEE/ACM Trans. Netw. 24 (5) (2016) 2795–2808.
[61] M. Hosseini, T.X. Tran, D. Pompili, K. Elisevich, H. Soltanian-Zadeh, Deep learning with edge computing for localization of epileptogenicity using multimodal rs-fMRI and EEG big data, in: Proceedings of the 2017 IEEE ICAC, IEEE, New York, NY, USA, 2017, pp. 83–92.
[62] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, J.J. Leonard, Past, present, and future of simultaneous localization and mapping: toward the robust-perception age, IEEE Trans. Robot. 32 (6) (2016) 1309–1332.
[63] E. Dong, J. Xu, C. Wu, Y. Liu, Z. Yang, Pair-Navi: Peer-to-Peer indoor navigation with mobile visual SLAM, in: Proceedings of the 2019 IEEE INFOCOM, IEEE, New York, NY, USA, 2019, pp. 1189–1197.
[64] J. Xu, H. Cao, D. Li, K. Huang, C. Qian, L. Shangguan, Z. Yang, Edge assisted mobile semantic visual SLAM, in: Proceedings of the 2020 IEEE INFOCOM, IEEE, New York, NY, USA, 2020, pp. 1828–1837.
[65] R. Mur-Artal, J.M.M. Montiel, J.D. Tardós, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Trans. Robot. 31 (5) (2015) 1147–1163.
[66] P. Mostafalu, A. Tamayol, R. Rahimi, M. Ochoa, A. Khalilpour, G. Kiaee, I.K. Yazdi, S. Bagherifard, M.R. Dokmeci, B. Ziaie, S.R. Sonkusale, A. Khademhosseini, Smart bandage for monitoring and treatment of chronic wounds, Small 14 (33) (2018).
[67] P. Leijdekkers, V. Gay, A self-test to detect a heart attack using a mobile phone and wearable sensors, in: Proceedings of the 2008 IEEE CBMS, IEEE, New York, NY, USA, 2008, pp. 93–98.
[68] G. Lan, D. Ma, W. Xu, M. Hassan, W. Hu, CapSense: capacitor-based activity sensing for kinetic energy harvesting powered wearable devices, in: Proceedings of the 2017 ACM MobiQuitous, ACM, New York, NY, USA, 2017, pp. 106–115.
[69] W. Xu, G. Lan, Q. Lin, S. Khalifa, M. Hassan, N. Bergmann, W. Hu, KEH-Gait: using kinetic energy harvesting for Gait-based user authentication systems, IEEE Trans. Mob. Comput. 18 (1) (2019) 139–152.
[70] J. Li, A. Badam, R. Chandra, S. Swanson, B. Worthington, Q. Zhang, On the energy overhead of mobile storage systems, in: Proceedings of the 2014 USENIX FAST, USENIX Association, Santa Clara, CA, 2014, pp. 105–118.
[71] H. Kim, N. Agrawal, C. Ungureanu, Revisiting storage for smartphones, ACM Trans. Storage 8 (4) (2012) 1–25.
[72] A. Rudenko, P. Reiher, G.J. Popek, G.H. Kuenning, Saving portable computer battery power through remote process execution, ACM SIGMOBILE Mob. Comput. Commun. Rev. 2 (1) (1998) 19–26.
[73] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, CloneCloud: elastic execution between mobile device and cloud, in: Proceedings of the 2011 ACM EuroSys, ACM, New York, NY, USA, 2011, pp. 301–314.
[74] M. Buettner, G.V. Yee, E. Anderson, R. Han, X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks, in: Proceedings of the 2006 ACM SenSys, ACM, New York, NY, USA, 2006, pp. 307–320.
[75] C. Gomez, J. Oller, J. Paradells, Overview and evaluation of Bluetooth Low Energy: an emerging low-power wireless technology, Sensors 12 (9) (2012) 11734–11753.
[76] Y. Li, J. Yang, J. Wang, DyLoRa: towards energy efficient dynamic LoRa transmission control, in: Proceedings of the 2020 IEEE INFOCOM, IEEE, New York, NY, USA, 2020, pp. 2312–2320.
[77] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, J.R. Smith, Ambient backscatter: wireless communication out of thin air, in: Proceedings of the 2013 ACM SIGCOMM, ACM, New York, NY, USA, 2013, pp. 39–50.
[78] B. Kellogg, A. Parks, S. Gollakota, J.R. Smith, D. Wetherall, Wi-Fi Backscatter: Internet connectivity for RF-powered devices, ACM SIGCOMM Comput. Commun. Rev. 44 (4) (2014) 607–618.
[79] P. Zhang, M. Rostami, P. Hu, D. Ganesan, Enabling practical backscatter communication for on-body sensors, in: Proceedings of the 2016 ACM SIGCOMM Conference, ACM, New York, NY, USA, 2016, pp. 370–383.
[80] P. Suarez, C.-G. Renmarker, A. Dunkels, T. Voigt, Increasing ZigBee network lifetime with X-MAC, in: Proceedings of the 2008 ACM REALWSN, ACM, New York, NY, USA, 2008, pp. 26–30.
[81] M. Buettner, Backscatter Protocols and Energy-Efficient Computing for RF-powered Devices, Ph.D. thesis, University of Washington, 2012.
[82] Q. Huang, Y. Mei, W. Wang, Q. Zhang, Battery-free sensing platform for wearable devices: the synergy between two feet, in: Proceedings of the 2016 IEEE INFOCOM, IEEE, New York, NY, USA, 2016, pp. 1–9.
[83] DATA CONFIDENTIALITY: Identifying and Protecting Assets and Data Against Data Breaches, https://www.nccoe.nist.gov/sites/default/files/library/project-descriptions/dc-ip-project-description-final.pdf. (Accessed 30 January 2021).
[84] Security Analysis of Bluetooth Technology, https://courses.csail.mit.edu/6.857/2018/project/Filizzola-Fraser-Samsonau-Bluetooth.pdf. (Accessed 6 May 2021).
[85] crackle, crack Bluetooth Smart (BLE) encryption, https://github.com/mikeryan/crackle/. (Accessed 30 January 2021).
[86] Y. Kim, W.S. Lee, V. Raghunathan, N.K. Jha, A. Raghunathan, Vibration-based secure side channel for medical devices, in: Proceedings of the 2015 ACM DAC, ACM, New York, NY, USA, 2015, pp. 1–6.
[87] G. Revadigar, C. Javali, W. Xu, W. Hu, S. Jha, Secure key generation and distribution protocol for wearable devices, in: Proceedings of the 2016 IEEE PerCom Workshops, IEEE, New York, NY, USA, 2016, pp. 1–4.
[88] W. Xu, J. Zhang, S. Huang, C. Luo, W. Li, Key generation for Internet of Things: a contemporary survey, ACM Comput. Surv. 54 (1) (2021) 1–37.
[89] W. Shen, P. Ning, X. He, H. Dai, Ally Friendly Jamming: how to jam your enemy and maintain your own wireless connectivity at the same time, in: Proceedings of the 2013 IEEE S&P, IEEE, New York, NY, USA, 2013, pp. 174–188.
[90] M. Nasr, R. Shokri, A. Houmansadr, Comprehensive privacy analysis of deep learning: passive and active white-box inference attacks against centralized and federated learning, in: Proceedings of the 2019 IEEE S&P, IEEE, New York, NY, USA, 2019, pp. 739–753.
[91] R. Shokri, M. Stronati, C. Song, V. Shmatikov, Membership inference attacks against machine learning models, in: Proceedings of the 2017 IEEE S&P, IEEE, New York, NY, USA, 2017, pp. 3–18.
[92] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, H. Qi, Beyond inferring class representatives: user-level privacy leakage from federated learning, in: Proceedings of the 2019 IEEE INFOCOM, IEEE, New York, NY, USA, 2019, pp. 2512–2520.
[93] L. Melis, C. Song, E. De Cristofaro, V. Shmatikov, Exploiting unintended feature leakage in collaborative learning, in: Proceedings of the 2019 IEEE S&P, IEEE, New York, NY, USA, 2019, pp. 691–706.
[94] K. Ganju, Q. Wang, W. Yang, C.A. Gunter, N. Borisov, Property inference attacks on Fully Connected Neural Networks using permutation invariant representations, in: Proceedings of the 2018 ACM CCS, ACM, New York, NY, USA, 2018, pp. 619–633.
[95] F. Tramer, D. Boneh, Slalom: fast, verifiable and private execution of neural networks in trusted hardware, in: Proceedings of the 2019 ICLR, 2019.
[96] F. Mo, A.S. Shamsabadi, K. Katevas, S. Demetriou, I. Leontiadis, A. Cavallaro, H. Haddadi, DarkneTZ: towards model privacy at the edge using trusted execution environments, in: Proceedings of the 2020 ACM MobiSys, ACM, New York, NY, USA, 2020, pp. 161–174.
[97] M. Nasr, R. Shokri, A. Houmansadr, Machine learning with membership privacy using adversarial regularization, in: Proceedings of the 2018 ACM CCS, ACM, New York, NY, USA, 2018, pp. 634–646.
[98] Project Probable-Wordlists, https://github.com/berzerk0/Probable-Wordlists. (Accessed 30 January 2021).
[99] A.T.B. Jin, D.N.C. Ling, A. Goh, Biohashing: two factor authentication featuring fingerprint data and tokenised random number, Pattern Recognit. 37 (11) (2004) 2245–2255.
[100] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: a unified embedding for face recognition and clustering, in: Proceedings of the 2015 IEEE CVPR, IEEE, New York, NY, USA, 2015, pp. 815–823.
[101] N. Karapanos, C. Marforio, C. Soriente, S. Capkun, Sound-Proof: usable two-factor authentication based on ambient sound, in: Proceedings of the 2015 USENIX Security, USENIX Association, Washington, D.C., 2015, pp. 483–498.
[102] X. Xu, J. Yu, Y. chen, Q. Hua, Y. Zhu, Y.-C. Chen, M. Li, TouchPass: towards behavior-irrelevant on-touch user authentication on smartphones leveraging vibrations, in: Proceedings of the 2020 ACM MobiCom, ACM, New York, NY, USA, 2020.
[103] K. Zeng, K. Govindan, P. Mohapatra, Non-cryptographic authentication and identification in wireless networks [Security and Privacy in Emerging Wireless Networks], IEEE Wirel. Commun. 17 (5) (2010) 56–62.
[104] A. Cassola, W.K. Robertson, E. Kirda, G. Noubir, A practical, targeted, and stealthy attack against WPA Enterprise authentication, in: Proceedings of the 2013 NDSS, Internet Society, Reston, VA, USA, 2013.
[105] J. Wang, F. Hu, Y. Zhou, Y. Liu, H. Zhang, Z. Liu, BlueDoor: breaking the secure information flow via BLE vulnerability, in: Proceedings of the 2020 ACM MobiSys, ACM, New York, NY, USA, 2020, pp. 286–298.
[106] D. Boneh, H. Corrigan-Gibbs, S. Schechter, Balloon hashing: a memory-hard function providing provable protection against sequential attacks, in: J.H. Cheon, T. Takagi (Eds.), Proceedings of the 2016 ASIACRYPT, Springer, Berlin, Heidelberg, 2016, pp. 220–248.
[107] S. Jarecki, H. Krawczyk, M. Shirvanian, N. Saxena, Device-enhanced password protocols with optimal online-offline protection, in: Proceedings of the 11th ACM ASIACCS, ACM, New York, NY, USA, 2016, pp. 177–188.
[108] E. Ronen, A. Shamir, A. Weingarten, C. O'Flynn, IoT goes nuclear: creating a Zigbee chain reaction, IEEE Secur. Priv. 16 (1) (2018) 54–62.

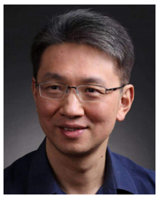
[109] R. Vishwakarma, A.K. Jain, A survey of DDoS attacking techniques and defence mechanisms in the IoT network, Telecommun. Syst. 73 (1) (2020) 3–25.
[110] Q. Niyaz, W. Sun, A.Y. Javaid, A deep learning based DDoS detection system in software-defined networking (SDN), EAI Endorsed Trans. Security Safety 4 (12) (2017).
[111] A. Yaar, A. Perrig, D. Song, Pi: a path identification mechanism to defend against DDoS attacks, in: Proceedings of the 2003 IEEE S&P, IEEE, New York, NY, USA, 2003, pp. 93–107.
[112] J.A. Pienaar, R. Hundt, JSWhiz: static analysis for JavaScript memory leaks, in: Proceedings of the 2013 IEEE/ACM CGO, IEEE, New York, NY, USA, 2013, pp. 1–11.
[113] Y. Shoshitaishvili, R. Wang, C. Hauser, C. Kruegel, G. Vigna, Firmalice-automatic detection of authentication bypass vulnerabilities in binary firmware, in: Proceedings of the 2015 NDSS, Internet Society, Reston, VA, USA, 2015.
[114] F. Dang, Z. Li, Y. Liu, E. Zhai, Q.A. Chen, T. Xu, Y. Chen, J. Yang, Understanding fileless attacks on Linux-based IoT devices with HoneyCloud, in: Proceedings of the 2019 ACM MobiSys, ACM, New York, NY, USA, 2019, pp. 482–493.
[115] R. Goyal, N. Dragoni, A. Spognardi, Mind the tracker you wear: a security analysis of wearable health trackers, in: Proceedings of the 2016 ACM SAC, ACM, New York, NY, USA, 2016, pp. 131–136.
[116] Bluetooth low energy (BLE) enabled devices market volume worldwide, from 2013 to 2020, https://www.statista.com/statistics/750569/worldwide-bluetooth-low-energy-device-market-volume/. (Accessed 30 January 2021).
[117] C. Dwork, A. Smith, T. Steinke, J. Ullman, Exposed! A survey of attacks on private data, Annu. Rev. Stat. Appl. (2017).
[118] C. Wang, X. Guo, Y. Wang, Y. Chen, B. Liu, Friend or foe? Your wearable devices reveal your personal PIN, in: Proceedings of the 2016 ACM ASIACCS, ACM, New York, NY, USA, 2016, pp. 189–200.
[119] X. Liu, Z. Zhou, W. Diao, Z. Li, K. Zhang, When good becomes evil: keystroke inference with smartwatch, in: Proceedings of the 2015 ACM CCS, ACM, New York, NY, USA, 2015, pp. 1273–1285.
[120] H. Wang, T.T.-T. Lai, R. Roy Choudhury, MoLe: motion leaks through smartwatch sensors, in: Proceedings of the 2015 ACM MobiCom, ACM, New York, NY, USA, 2015, pp. 155–166.
[121] Y. Liu, Z. Li, aLeak: privacy leakage through context-free wearable side-channel, in: Proceedings of the 2018 IEEE INFOCOM, IEEE, New York, NY, USA, 2018, pp. 1232–1240.
[122] A. Maiti, R. Heard, M. Sabra, M. Jadliwala, Towards inferring mechanical lock combinations using wrist-wearables as a side-channel, in: Proceedings of the 2018 ACM WiSec, ACM, New York, NY, USA, 2018, pp. 111–122.
[123] F. McKeen, I. Alexandrovich, A. Berenzon, C.V. Rozas, H. Shafi, V. Shanbhogue, U.R. Savagaonkar, Innovative instructions and software model for isolated execution, in: Proceedings of the 2013 ACM HASP, ACM, New York, NY, USA, 2013.
[124] Trustzone – Arm Developer, https://developer.arm.com/ip-products/security-ip/trustzone. (Accessed 6 May 2021).
[125] J.F. Kurose, K.W. Ross, Computer Networking: A Top-Down Approach, 4th edition, Addison-Wesley Longman Publishing Co., Inc., USA, 2007.
[126] J. Shim, K. Jung, S. Cho, M. Park, S. Han, A case study on vulnerability analysis and firmware modification attack for a wearable fitness tracker, IT Converg. Pract. 5 (4) (2017) 25–33.
[127] Understanding Denial-of-Service Attacks, https://us-cert.cisa.gov/ncas/tips/ST04-015. (Accessed 30 January 2021).
[128] KrebsOnSecurity Hit With Record DDoS, https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/. (Accessed 30 January 2021).
[129] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J.A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, Y. Zhou, Understanding the Mirai botnet, in: Proceedings of the 2017 USENIX Security, USENIX Association, Vancouver, BC, 2017, pp. 1093–1110.
[130] (0Day) NETGEAR R6700 httpd strtblupgrade Integer Overflow Remote Code Execution Vulnerability, https://www.zerodayinitiative.com/advisories/ZDI-20-709/. (Accessed 30 January 2021).
[131] 12 DDoS Statistics That Should Concern Business Leaders, https://www.coxblue.com/12-ddos-statistics-that-should-concern-business-leaders/. (Accessed 30 January 2021).
[132] L. Spitzner, Honeypots: Tracking Hackers, Addison-Wesley Professional, 2002.
[133] Y. Chen, X. Qin, J. Wang, C. Yu, W. Gao, FedHealth: a federated transfer learning framework for wearable healthcare, IEEE Intell. Syst. 35 (4) (2020) 83–93.
[134] A. Akbari, R. Jafari, Personalizing activity recognition models through quantifying different types of uncertainty using wearable sensors, IEEE Trans. Biomed. Eng. 67 (9) (2020) 2530–2541.
[135] D.R. Kasar, C.S. Graham, E.S. Jol, Inductive charging between electronic device, https://patents.google.com/patent/US20200112195A1/, 2020. (Accessed 30 January 2021).
[136] S. Yang, X. Mao, Out-of-band communication during wireless battery charging, https://patents.google.com/patent/US20200161907A1, 2020. (Accessed 30 January 2021).
[137] Galaxy S10's PowerShare reverse charging speed test with Samsung Buds and Watch Active, https://www.phonearena.com/news/S10-Samsung-Buds-Galaxy-Watch-batteries-wireless-PowerShare-charging-speed-test_id114565. (Accessed 30 January 2021).
[138] Samsung vs Huawei reverse wireless charging test, https://www.androidauthority.com/samsung-vs-huawei-reverse-wireless-charging-968692/. (Accessed 30 January 2021).
[139] Z. Wang, V. Leonov, P. Fiorini, C. Van Hoof, Realization of a wearable miniaturized thermoelectric generator for human body applications, Sens. Actuators A, Phys. 156 (1) (2009) 95–102, eUROSENSORS XXII, 2008.
[140] M. Magno, D. Brunelli, L. Sigrist, R. Andri, L. Cavigelli, A. Gomez, L. Benini, InfiniTime: multi-sensor wearable bracelet with human body harvesting, Sustain. Comput.: Inf. Syst. 11 (2016) 38–49.
[141] H. Abbasizadeh, S.Y. Kim, T.T.K. Nga, D. Khan, S.J. Oh, S.J. Kim, K.T. Kim, D.I. Kim, K.Y. Lee, A 5.2 GHz RF Energy Harvester system using reconfigurable parallel rectenna, in: Proceedings of the 2018 IEEE MWSCAS, IEEE, New York, NY, USA, 2018, pp. 1–4.
[142] M. Magno, D. Boyle, Wearable Energy Harvesting: from body to battery, in: Proceedings of the 2017 IEEE DTIS, IEEE, New York, NY, USA, 2017, pp. 1–6.

Xinqi Jin is currently an undergraduate student in the School of Software at Tsinghua University, China. His research interests include Artificial Intelligence of Things (AIoT), and mobile computing.

Lingkun Li received the B.E. degree in Software Engineering from Beijing Jiaotong University and an M.E. degree in Software Engineering from Tsinghua University. He is currently pursuing a Ph.D. degree in Computer Science and Engineering at Michigan State University. His research interests include Internet of Things, and mobile computing.

Fan Dang received the B.E. and Ph.D. degrees in School of Software from Tsinghua University. He is currently with the Global Innovation Exchange and School of Software, Tsinghua University. His research interests include AIoT, IoT security, and edge computing.

Xinlei Chen expects to join Tsinghua Shenzhen International Graduate School as an assistant professor in July 2021. He worked as a postdoctoral research associate in Electrical Engineering Department at Carnegie Mellon University from 2018–2020. He received the B.E. and M.S. degrees in Electronic Engineering from Tsinghua University, China, in 2009 and 2012, respectively, and the Ph.D. degree in Electrical Engineering from Carnegie Mellon University, Pittsburgh, PA, USA. His research interests lie in mobile computing, embedded system, edge computing, wearable devices etc.


Yunhao Liu received the B.S. degree in automation from Tsinghua University, China, in 1995, the M.S. and Ph.D. degrees in computer science and engineering from Michigan State University, in 2003 and 2004, respectively. He is currently with the Global Innovation Exchange in Tsinghua University. He is a fellow of the IEEE and ACM.