Smartwatch-Based Sensing Framework for Continuous Data Collection: Design and Implementation

Yuuki Nishiyama
nishiyama@csis.u-tokyo.ac.jp
The University of Tokyo
Japan

Kaoru Sezaki
sezaki@iis.u-tokyo.ac.jp
The University of Tokyo
Japan

ABSTRACT

Smartwatches are an increasingly popular technology that employs advanced sensors (e.g., location, motion, and microphone) comparable to those used by smartphones. Passive mobile sensing, a method of acquiring human behavior data from mobile and wearable devices inconspicuously, is widely used in research fields related to behavior analysis. In combination with machine learning, passive mobile sensing can be used to interpret various human and environmental contexts without requiring user intervention. Because smartwatches are always worn on the wrist, they have the potential to collect data that cannot be collected by smartphones. However, the effective use of smartwatches as platforms for passive mobile sensing poses challenges in terms of battery life, storage, and communication. To address these challenges, we designed and implemented a tailored framework for off-the-shelf smartwatches. We evaluated power consumption under eight different sensing conditions using three smartwatches. The results demonstrate that the framework can collect sensor data with a battery life of 16-31 h depending on the settings. Finally, we considered potential future solutions for optimizing power consumption in passive sensing with off-the-shelf smartwatches.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; **Ubiquitous and mobile devices**.

KEYWORDS

passive sensing framework, smartwatch, motion sensor, environmental sound collection, battery consumption

ACM Reference Format:
Yuuki Nishiyama and Kaoru Sezaki. 2023. Smartwatch-Based Sensing Framework for Continuous Data Collection: Design and Implementation. In *Adjunct Proceedings of the 2023 ACM International Joint Conference on Pervasive and Ubiquitous Computing the 2023 ACM International Symposium on Wearable Computing (UbiComp/ISWC '23 Adjunct), October 8–12, 2023, Cancun, Quintana Roo, Mexico.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3594739.3612874

UbiComp/ISWC '23 Adjunct , October 8–12, 2023, Cancun, Quintana Roo, Mexico

ACM ISBN 979-8-4007-0200-6/23/10…$15.00
https://doi.org/10.1145/3594739.3612874

1 INTRODUCTION

The market for wearable devices, such as smartwatches, smart bands, and smart rings, has significantly expanded with the increasing popularity of such devices. According to a report by Counterpoint Technology Market Research, the global market for smartwatches is growing by 9% annually [17]. Smartwatches are equipped with smartphone capabilities and advanced sensors such as microphones, positioning sensors, Bluetooth, Wi-Fi, and motion sensors, all of which provide a wide range of services that enrich our lives. More recently, smartwatches have become equipped with operating systems (OS) that allow third-party developers to easily develop and deploy sensor-based applications. Several context recognition applications have been developed to detect fine-grained activities of daily living (ADLs) [3, 7, 9–11, 13, 15, 18, 22] - such as keyboard typing, hand washing, and cooking - using multiple sensors on the smartwatch, primarily motion sensors and microphones.

Passive mobile sensing is used to analyze and interpret human behaviors and environmental contexts using inconspicuously collected sensor data from mobile and wearable devices [19]. This approach enables the automatic recognition of various contexts without manual intervention simply by carrying a device. Various applications, including disease monitoring methods [1, 4, 16], have been proposed based upon this approach. Furthermore, platforms specifically designed for smartphones [2, 5, 8, 14, 21] have supported these advancements by enabling the continuous collection of data from multiple sensors in short steps.

With the advent of these platforms, the use of passive mobile sensing has rapidly expanded beyond computer science to encompass psychology, humanities, and medicine. Establishing an environment in which smartwatches can be utilized effectively is important in any application of passive mobile sensing. However, effectively leveraging the potential of smartwatches for passive mobile sensing remains a challenge. Smartwatches have limitations in terms of battery life, storage capacity, and communication capabilities, necessitating specific design considerations.

In this study, we addressed these challenges by designing and implementing a framework for passive mobile sensing specifically tailored to off-the-shelf smartwatches. Our framework, implemented on an Apple Watch, supports the use of eight types of sensors including motion, microphone, and location sensors. Finally, we evaluated the power consumption of our framework under eight different sensing conditions using three types of smartwatches. Our experimental results demonstrate that the proposed framework collects sensor data within the sufficient range of 16-31 h depending on the settings. In addition, the results reveal that certain sensors exhibit high power consumption tendencies, necessitating the optimization of the processing frequency and sensing approaches.

2 RELATED WORK

In Section 2.1, we summarize prior studies pertaining to mobile sensing using off-the-shelf smartphones and wearable devices, also known as passive mobile sensing. In Section 2.2, we present examples of context-recognition methods using smartwatches. In Section 2.3, we describe existing frameworks and libraries designed for passive mobile sensing and context recognition.

2.1 Passive Mobile Sensing

Passive mobile sensing enables the continuous collection of sensor data from hardware and software without requiring manual user intervention. This approach also allows for the analysis of human behavior using objective data [6, 12, 20], which can be combined with machine learning techniques for the automatic detection of a variety of contexts [1, 4, 16]. For instance, passive mobile sensing has been applied to detect binge drinking [1], depression [16], and loneliness [4].

Doryab et al. [4] collected daily activity data from smartphones and Fitbits of 160 college students over a semester to infer levels of loneliness. For activity data, they collected motion (i.e., accelerometer, gyroscope, and magnetometer), location, screen status, Bluetooth address, phone call events, sleep, and steps from both devices. By applying reinforcement learning to the collected data, they achieved an accuracy of approximately 80% in the binary classification task of loneliness and non-loneliness.

Smartphones, which are the devices primarily used for passive mobile sensing, are equipped with various hardware and software sensors that can collect data to effectively interpret behavioral patterns. However, a smartphone cannot collect data when it is not in use. In addition, movements of the upper arm alone, such as washing hands, brushing teeth, or putting on a face mask, are difficult to detect via smartphone. Furthermore, some OSs restrict sensors from sensing the background. For instance, in iOS, it is impossible to control the endpoints of background audio recordings. Owing to these limitations, existing methods do not allow continuous recording of environmental sounds, conversation events, or noise levels in the background.

2.2 Context Recognition Using Smartwatch

Similarly to smartphones, smartwatches are equipped with numerous sensors, making them ideal sensing platforms. Platforms such as watchOS for Apple Watch, wearOS for Android Wear, and Fitbit are commonly used in research projects as application programming interfaces.

Furthermore, motion sensors and microphones on smartwatches are known to be powerful sensors for the detection of fine-grained ADLs and hand gestures [3, 7, 9–11, 13, 15, 18, 22]. For example, Laput, G. et al. developed an algorithm [11] that classifies 25 hand activities using motion sensors on a wrist-warming device, achieving a classification accuracy of 95.2%. In the case of the microphone, Komatsu et al. proposed a method to detect daily conversations and ambient noise on an off-the-shelf smartwatch, obtaining accuracies of 90% and 85% in silent and noisy conditions, respectively [9]. Bhattacharya, S. et al. proposed a method to detect DALs [3] using acoustic and inertial sensors on an off-the-shelf smartwatch. The method was used to recognize 23 activities including writing, cooking, and cleaning with high accuracy.

The primary objective of these studies was the development of behavior recognition methods and algorithms using smartwatches, which have yet to be sufficiently designed and discussed as passive mobile sensing platforms.

2.3 Mobile Sensing Framework and Library

Recently, several frameworks have been proposed for the implementation of passive mobile sensing. The AWARE framework [5, 14], Sensus [21], EmotionSense[1], Passive Data Kit[2], and SensingKit [8] are mobile sensing frameworks that are frequently and ubiquitously used in computing communities. These frameworks allow users to collect various sensor data simply by installing certain smartphone applications, enabling collaboration between the computer science field and various other fields including medicine, psychology, and social sciences.

The AWARE framework [5, 14] and SensingKit [8] enable the easy embedding of sensing libraries into any application through a de facto standard library manager, such as Jetpack (for Android) and CocoaPods (for iOS). These frameworks are essentially optimized for continuous sensing on smartphones, and are guaranteed to work on major OSs such as iOS and Android.

3 MOTIVATION

The objective of this study was to develop a passive mobile sensing infrastructure specifically designed for smartwatches while considering the constraints associated with smartwatch technology. In the context of passive mobile sensing, smartphones have been recognized as useful sensing platforms, with several frameworks [5, 8, 14, 21] having been developed to support passive mobile sensing. However, standalone smartphones cannot collect data when they are not in use, and are insufficient for detecting activities involving only the upper arm. Although smartwatches offer a potential solution to these limitations [3, 9, 11], they are associated with constraints in communication, storage, power consumption, and background processing, which differ from those of smartphones. Currently, there is a lack of passive mobile sensing infrastructure tailored to commercially available smartwatches, resulting in the underutilization of various behavior recognition methods proposed for smartphone-based passive mobile sensing studies.

Therefore, we designed and implemented a prototype sensing platform on smartwatches capable of continuously collecting data from multiple sensors, and conducted fundamental evaluation experiments on power consumption under different settings. By enabling passive sensing using commercially available smartwatches, we can complement the sensing capabilities of standalone smartphones and improve the granularity of passive sensing. In this paper, we focused on the Apple Watch series.

4 PASSIVE MOBILE SENSING FRAMEWORK FOR SMARTWATCHES

A passive mobile sensing framework for smartwatches requires (1) support from multiple sensors and (2) continuous collection of

[1] https://emotionsense.github.io
[2] https://passivedatakit.org/

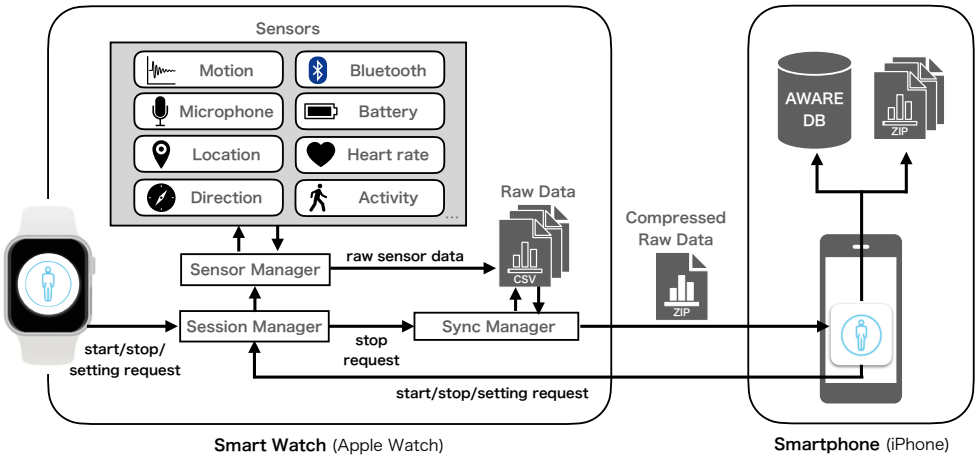


Figure 1: Overview of smartwatch-based passive sensing framework

sensor data. Additionally, (3) it must have a realistic battery life that is not intrusive for the user.

To achieve (1), the proposed framework supports the eight types of sensors (e.g., motion, microphone, and battery) used by current models of the Apple Watch. Any sensor can continuously run on our proposed framework using the background mode on watchOS, which corresponds to (2). In addition, (3) is met by periodically transferring the collected sensor data to the background using a data compression method, thereby minimizing power consumption.

4.1 Design

Figure 1 presents an overview of the proposed framework. The proposed framework can be primarily divided into a smartphone side and a smartwatch side. The smartwatch side includes the session, sensor, and sync managers. The session manager handles operation commands - such as start, stop, and various configurations of a sensing session - from the smartwatch and a smartphone.

The sensor manager activates and deactivates the hardware and software sensors. The received sensor data are periodically saved as comma-separated values (CSV) files. The sync manager controls the CSV file transfer using a host smartphone. When the session manager receives a session stop or forces a sync command, the sync manager starts transferring the files. In addition, the sync manager periodically transfers files if the session enables a sync function with a preset interval. To minimize the file size, the transfer files are compressed using a compression algorithm.

The smartphone saves CSV files received from the smartwatch. This library is designed as a plugin for the AWARE framework [5, 14] Thus, the collected sensor data are saved in a database of the AWARE framework and processed in the AWARE ecosystem.

4.2 Implementation

This library was implemented on watchOS 9, and the source code was written in Swift. This library for iOS was implemented as part of the AWARE framework[3]. Moreover, the proposed framework can be installed through CocoaPods, a de facto standard library manager for iOS and watchOS application development.

4.2.1 Session and Sensor Managers. The sensor manager manages the start, end, and configuration of the sensing phase. The `AWSensor` class (shown in Figure 2), which represents the sensor manager, provides functions to start, end, and configure a sensing session. The class is implemented as a singleton, enabling developers to access the project anywhere. To launch an app in background mode in watchOS, one of the following must be used: audio, location update, voice-over IP, remote notifications, or workout processing. This library uses workout processing for each sensing session.

The sensor manager manages hardware and software sensors on an Apple Watch. In the current implementation, our library supports the continuous collection of data from motion, location, microphone, battery, and heart rate sensors. The collected sensor data are temporarily saved as CSV files on the smartwatch. The motion sensor includes an accelerometer, a gyroscope, and a device that collects sensor data by 100 Hz to the maximum extent. The raw audio file is saved as an MPEG-4 Low Complexity AAC audio object with a sampling rate of 22050 Hz. In addition, the microphone sensor extracts ambient noise levels once per second using real-time audio processing. The ambient noise level contains the sound pressure level (decibel) and root mean square (RMS) value of the sound. Moreover, the audio sensor supports an audio recognition function using CoreML[4], a built-in machine learning framework on watchOS. CoreML can use deep learning-based classification models created by Apple's CreateML or PyTorch[5]. Figure 2 presents sample code for activating motion, ambient noise, and battery sensors using `AWSensor`.

4.2.2 Sync Manager. The sync manager manages file transfer events and schedules them according to the configuration of the sensor manager. File transfers are performed using the transferFile method on WCSession[6]. Prior to transferring a file, the manager compresses

[3]https://github.com/awareframework

[4]https://developer.apple.com/machine-learning/core-ml/

[5]https://coremltools.readme.io/docs/pytorch-conversion

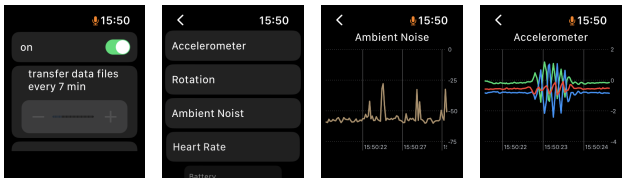
[6]https://developer.apple.com/documentation/watchconnectivity/wcsession

```
AWSensor.shared.start(AWSensorConfig().apply{config in
    // sensor configuration
    config.motionSensorHz = 30

    // list of activated sensors (set `true` need to use)
    config.activateMotionSensor = true
    config.activateAmbientNoiseSensor = true
    config.activateBatterySensor = true

    // file transfer settings
    config.autoFileTransferInterval = 300 // 5 minutes in this case
    config.autoFileTransfer = true  // transfer sensor data during sensing
})
```

Figure 2: Sample code to integrate proposed framework into watchOS application



(a) Main screen (b) Setting screen (c) Ambient noise (d) Accelerometer

Figure 3: Screenshots of example watchOS application

it using the zlib algorithm[7] to minimize the file size. As shown in Figure 2, `AWSensor` can control the interval for periodic file transfers.

4.2.3 Example Application on watchOS. Figure 3 presents screenshots of a sample watchOS application based on the proposed library. On one of the main screens (Figure 3(a)), the toggle button is connected to the `start()` and `stop()` functions on `AWSensor`. `AWSensor` saves the collected sensor data into a CSV file immediately. In addition, `AWSensor` caches the collected sensor data using data visualization and preprocessing. Figures 3(b), 3(c), and 3(d) show examples of data visualization. Each chart is updated in real time when the sensor receives new data.

5 EVALUATION

In our performance evaluation, we investigated the battery consumption of our proposed framework under eight conditions. During the experiments, a smartwatch was fixed on a desk to monitor battery consumption from a fully charged state to a fully discharged state.

5.1 Methods

Table 1 lists the identifiers of the eight evaluation conditions, along with corresponding details. `baseline` is the baseline condition, wherein battery sensors are used to record battery levels 1 min. `audio_raw`, `audio_noise`, and `audio_conv` are conditions pertaining to audio sensors. `audio_raw` continuously records raw audio data and transfers the audio file to a compatible smartphone application every 5 min. `audio_noise` extracts audio features (i.e., noise level and RMS) from the raw audio data every 1 sec (see Section 4.2.1). `audio_conv` detects conversation events 1 sec using CoreML, a built-in deep-learning-based machine learning module on iOS and watchOS. For conversation detection, we used the audio classification model developed in our prior work [9].

As a motion sensor, we used an accelerometer, as they are commonly used to gather information on human activities in context-aware computing. Accelerometer data are collected from three devices with two different sensing frequencies: 5 and 100 Hz. Specifically, 100 Hz is the maximum sampling rate on watchOS, whereas 5 Hz is a common low sampling rate for behavioral detection. In this evaluation, we used the Apple Watch Series 8 (41mm, watchOS 9, and GPS model) as the benchmark device. In addition, we used a larger device (Apple Watch Series 8, 44mm) that was reasonably priced (Apple Watch SE generation 2: SE2). Accelerometer-related identifiers start with `acc`. The identifiers can also be split by an underscore, with the second, third, and fourth labels indicating sensing frequency, device type, and device size, respectively.

5.2 Hypothesis

We set a hypothesis stating that power consumption is likely to be lowest at baseline, and higher for motion sensors with a sampling rate of 100 Hz, as well as microphones. Conversely, power consumption is likely to be low for motion sensors with a low sampling rate of 5 Hz.

Similarly, audio-related sensors, which require higher data processing power, are expected to consume more power than acceleration sensors. In addition, large amounts of data transfer from smartwatches to smartphones may increase power consumption due to increased communication time and data processing.

5.3 Result

Figure 4 and Table 1 present the results of the performance evaluation. The order of results in terms of battery life is almost the same as that of compressed file sizes in the periodic transmissions from the smartwatch to a smartphone. In this experiment, the collected sensor data were transferred once every 5 min. Given a fixed transfer frequency, a smaller transfer file size directly correlates to a longer battery life.

The ambient-sound sensor (`audio_noise`) consumed approximately the same energy level as the motion sensors. The environmental sound sensor calculated the sound pressure level and RMS once every second. Although sound processing was performed, the frequency of data writing was extremely low, and the amount of data to be written was also small; consequently, power consumption was significantly lower than that for `audio_audio` and `audio_conv`.

Comparing the results of battery life between `acc_100Hz_s8_41mm` and `acc_5Hz_s8_41mm`, the lower sampling rate is associated with longer battery life. Sensors with higher sampling rates produce large amounts of sensor data, which leads to an overload in data processing and transmission. Moreover, a comparison of power consumption between devices exhibited no significant differences between 44 mm and 40 mm of Series 8. However, the operation time of SE2 was approximately 69% that of Series 8. SE2 has a second-generation optical heart rate sensor, whereas Series 8 has a third-generation sensor. In workout mode, which is required for background activity on the Apple Watch, the heartrate sensor keeps

[7] https://developer.apple.com/documentation/compression/algorithm/zlib

Table 1: Battery life in eight evaluation conditions

#	sensor name	sensing configuration	device name (series, size)	file size (compressed)	elapse time (hour)
baseline	battery	every 60 seconds	Series 8, 41mm	260 bytes (90 bytes)	31.28
audio_raw	raw audio	continuous (5 min)	Series 8, 41mm	1.2 MB (1.1 MB)	16.82
audio_noise	ambient noise	every 1 second	Series 8, 41mm	53 KB (18 KB)	25.20
audio_conv	conversation	every 1 second	Series 8, 41mm	49 KB (14 KB)	21.60
acc_5hz_s8_41mm	accelerometer	5 Hz	Series 8, 41mm	124 KB (21 KB)	29.17
acc_100hz_s8_41mm	accelerometer	100 Hz	Series 8, 41mm	2.5 MB (398 KB)	25.33
acc_100hz_s8_44mm	accelerometer	100 Hz	Series 8, 44mm	2.5 MB (398 KB)	25.10
acc_100hz_se2_40mm	accelerometer	100 Hz	SE Gen2, 40mm	2.5 MB (398 KB)	17.50

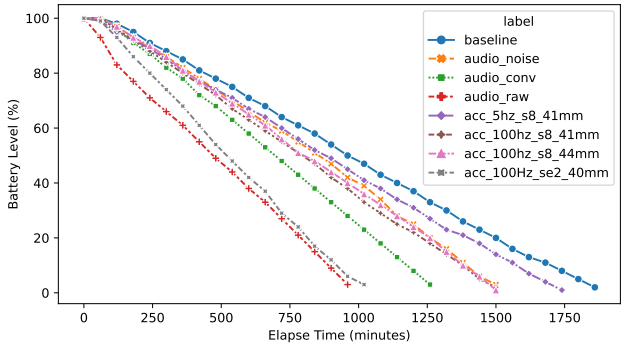


Figure 4: Trend of battery drain for smartwatch from 100% to 0% with different sensor settings

running even if it is not used during data collection. Therefore, the difference in battery consumption of heartrate sensors may be affected by this difference in functionality.

6 DISCUSSION

The following subsections examine the methods for reducing battery consumption in passive sensing on smartwatches based on the evaluation results. In addition, we discuss the effects of multiple sensor usage, as well as potential applications of the proposed passive sensing framework. Finally, we consider the limitations of our study, as well as future directions of research.

6.1 Battery consumption and its optimization

The results for battery consumption largely correlate with our hypothesis. Compared to the baseline, power consumption was higher for sensors that sampled more frequently and transferred larger files. Conversely, sensors with a lower sampling frequency and smaller transfer file size consumed less power.

One approach to optimizing power consumption is incorporating a duty cycle, which involves the cyclical activation and deactivation of resource-intensive processes. For instance, in the case of conversational event detection, the algorithm for detecting such events executes over a one-minute interval every three minutes. If a conversational event is detected during that interval, the detection process is extended by an additional minute. This strategy effectively reduces the frequency of utilizing the computationally intensive conversational event detection algorithm, thereby minimizing power consumption. This approach is practical for sound as well as motion sensors.

From the experimental results, we observe that the amount of data transferred in each transmission significantly affects battery consumption. Therefore, carefully selecting the transferred data volume is necessary to optimize power consumption. Additionally, the frequency of data transfers and communication method employed may also impact power consumption, warranting further investigation in future studies. For instance, data compression techniques tailored to the transferred data may be considered. Another strategy may involve bundling data transfers during battery charging periods, or directly uploading data to the data collection server when a Wi-Fi connection is available. These measures can potentially help mitigate power consumption in data transfer processes.

In multi-sensor tasks, power consumption varies depending on the combination of sensors. For example, given the 31.28 h baseline recording time, the use of an acceleration sensor at 5Hz incurs a reduction of 2.11 h, whereas speech recognition yields a further reduction of 9.68 h. Simultaneously using both sensors would likely result in approximately 19.49 hours of battery life (11.79 h less than the baseline).

6.2 Applications

By leveraging this framework, the seamless integration of passive sensing functionality into iOS can be achieved on Apple Watch hardware. Because the smartwatch is worn on the wrist, it can record the user's activities even without owning a smartphone. For example, the motion sensor can detect ADLs, which may help interpret the users' lifestyle patterns.

Android and iOS-based passive mobile sensing frameworks have been widely applied across various domains, especially as recognition foundations for the mental and physical states of humans in real-world environments [6, 20]. Previous studies have indicated that noise levels, conversations, and ambient sounds significantly impact mental health. However, the background sensing of ambient sounds is limited due to the restrictions associated with the latest versions of iOS. Our developed approach using watchOS offers a practical solution that enables audio sensing and on-device processing. Moreover, the framework allows for easy execution of deep-learning-based speech recognition models developed via CreateML or PyTorch through a simple drag-and-drop process, facilitating the flexible and straightforward utilization of audio-based

sensing. Similarly, the framework can be extended to incorporate motion-sensor-based activity recognition algorithms.

6.3 Limitation and Future Work

In this experiment, the battery consumption of the proposed framework was measured using a smartphone in a fixed position under ideal conditions. Owing to the substantial differences between the experimental and actual usage environments, more power is expected to be consumed than during normal daily usage. Accordingly, future evaluation experiments must consider a more practical experimental setting. The optimization of battery consumption is another future direction of research.

Beyond the scope of mental and physical state recognition, research is currently progressing on intervention methods to promote behavioral change. Consequently, there is potential for this framework to serve as an intervention platform. In the future, we plan to develop intervention mechanisms that utilize collected sensor data and collaborate with external systems to enable interventions at various times.

7 CONCLUSION

Smartwatches offer powerful sensing capabilities that can be leveraged for the detailed analysis of human behaviors and real time interventions in passive mobile sensing. However, the use of smartwatches for passive mobile sensing incurs constraints in communication, power consumption, processing capability, and storage. To address these constraints, we propose a sensing framework specifically designed for smartwatches considering the aforementioned limitations. We designed and implemented a platform on Apple's watchOS capable of continuously recording data from eight different sensors. Through evaluation experiments, we compared the power-consumption performance of the implemented platform under various conditions. The battery life varied with respect to scenario, ranging from 16 to 31 h for different activities. We observed that power consumption was influenced by the processing load on the device, and the sizes of the files transferred. The optimization of power consumption requires fine-tuning the sensing frequency, refining startup and shutdown timings, and exploring data compression techniques. In summary, our proposed smartwatch sensing framework represents a promising avenue for detailed behavioral analyses and real time interventions in passive mobile sensing. Future work should focus on optimizing power consumption by adjusting the sensing frequency, startup and shutdown timings, and data compression methods.

ACKNOWLEDGMENTS

This research was partially funded by JSPS KAKENHI Grant (23K17004) and NICT (222C01, 01101), Japan.

REFERENCES

[1] Sangwon Bae, Denzil Ferreira, Brian Suffoletto, et al. 2017. Detecting Drinking Episodes in Young Adults Using Smartphone-Based Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2, Article 5 (jun 2017), 36 pages. https://doi.org/10.1145/3090051

[2] Jakob E Bardram. 2020. The CARP Mobile Sensing Framework–A Cross-platform, Reactive, Programming Framework and Runtime Environment for Digital Phenotyping. *arXiv preprint arXiv:2006.11904* (2020).

[3] Sarnab Bhattacharya, Rebecca Adaimi, and Edison Thomaz. 2022. Leveraging Sound and Wrist Motion to Detect Activities of Daily Living with Commodity Smartwatches. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 2, Article 42 (jul 2022), 28 pages. https://doi.org/10.1145/3534582

[4] Afsaneh Doryab, Daniella K Villalba, Prerna Chikersal, et al. 2019. Identifying Behavioral Phenotypes of Loneliness and Social Isolation with Passive Sensing: Statistical Analysis, Data Mining and Machine Learning of Smartphone and Fitbit Data. *JMIR Mhealth Uhealth* 7, 7 (Jul 2019), e13209. https://doi.org/10.2196/13209

[5] Denzil Ferreira, Vassilis Kostakos, and Anind K. Dey. 2015. AWARE: Mobile Context Instrumentation Framework. *Frontiers in ICT* 2 (2015).

[6] Daniel Highland and Gang Zhou. 2022. A review of detection techniques for depression and bipolar disorder. *Smart Health* 24 (2022), 100282.

[7] Yuki Kasahara, Yuuki Nishiyama, and Kaoru Sezaki. 2022. Detecting Childcare Activities Using an Off-the-shelf Smartwatch. In *2022 IEEE International Conference on Smart Computing (SMARTCOMP)* (2022-06-20). IEEE, Espoo, Finland, 159–161. https://doi.org/10.1109/SMARTCOMP55677.2022.00038

[8] Kleomenis Katevas, Hamed Haddadi, and Laurissa Tokarchuk. 2016. SensingKit: Evaluating the Sensor Power Consumption in iOS Devices. 222–225.

[9] Yuki Komatsu, Kazuki Shimojo, Yuuki Nishiyama, and Kaoru Sezaki. 2022. Toward Measuring Conversation Duration Using a Wristwatch-type Wearable Device. In *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*. 150–152. https://doi.org/10.1109/SMARTCOMP55677.2022.00035

[10] Utkarsh Kunwar, Sheetal Borar, Moritz Berghofer, Julia Kylmälä, Ilhan Aslan, Luis A. Leiva, and Antti Oulasvirta. 2022. Robust and Deployable Gesture Recognition for Smartwatches. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. Association for Computing Machinery, New York, NY, USA, 277–291. https://doi.org/10.1145/3490099.3511125

[11] Gierad Laput and Chris Harrison. 2019. Sensing Fine-Grained Hand Activity with Smartwatches. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300568

[12] Sujen Man Maharjan et al. 2021. Passive sensing on mobile devices to improve mental health services with adolescent and young mothers in low-resource settings: the role of families in feasibility and acceptability. *BMC Medical Informatics and Decision Making* 21, 1 (2021), 117.

[13] Vimal Mollyn et al. 2022. SAMoSA: Sensing Activities with Motion and Subsampled Audio. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 3, Article 132 (sep 2022), 19 pages. https://doi.org/10.1145/3550284

[14] Yuuki Nishiyama et al. 2020. iOS Crowd-Sensing Won't Hurt a Bit!: AWARE Framework and Sustainable Study Guideline for iOS Platform. In *Distributed, Ambient and Pervasive Interactions* (2020-07-10), Vol. 12203. Springer International Publishing, Cham, 223–243. https://doi.org/10.1007/978-3-030-50344-4_17

[15] Shota Ono et al. 2022. Detecting Face-Mask Wearing Status Using Motion Sensors in Commercially Available Smartwatches. In *2022 IEEE International Conference on E-health Networking, Application & Services (HealthCom)* (2022-10-17). IEEE, Genoa, Italy, 107–112. https://doi.org/10.1109/HealthCom54947.2022.9982766

[16] Kennedy Opoku Asare et al. 2019. Towards Early Detection of Depression through Smartphone Sensing. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers* (London, United Kingdom) *(UbiComp/ISWC '19 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1158–1161. https://doi.org/10.1145/3341162.3347075

[17] Counterpoint Technology Market Research. 2022. Global Smartwatch Shipments Market Share Q4 2022. https://www.counterpointresearch.com/global-smartwatch-shipments-market-share/, Last accessed at June 4th in 2023.

[18] Muhammad Shoaib, Stephan Bosch, Hans Scholten, Paul J. M. Havinga, and Ozlem Durmaz Incel. 2015. Towards detection of bad habits by fusing smartphone and smartwatch sensors. In *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. 591–596. https://doi.org/10.1109/PERCOMW.2015.7134104

[19] Alina Trifan, Maryse Oliveira, and José Luís Oliveira. 2019. Passive Sensing of Health Outcomes Through Smartphones: Systematic Review of Current Solutions and Possible Limitations. *JMIR Mhealth Uhealth* 7, 8 (Aug 2019), e12649.

[20] Rui Wang, Weichen Wang, Alex daSilva, Jeremy F. Huckins, William M. Kelley, Todd F. Heatherton, and Andrew T. Campbell. 2018. Tracking Depression Dynamics in College Students Using Mobile Phone and Wearable Sensing. 2, 1, Article 43 (mar 2018), 26 pages. https://doi.org/10.1145/3191775

[21] Haoyi Xiong, Yu Huang, Laura E. Barnes, and Matthew S. Gerber. 2016. Sensus: A Cross-Platform, General-Purpose System for Mobile Crowdsensing in Human-Subject Studies. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 415–426. https://doi.org/10.1145/2971648.2971711

[22] Xuhai Xu, Jun Gong, Carolina Brum, et al. 2022. Enabling Hand Gesture Customization on Wrist-Worn Devices. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 496, 19 pages. https://doi.org/10.1145/3491102.3501904