

Exploring and Improving Crowdsourced Data Labeling for Mobile Activity Recognition

著者	Mairittha Nattaya
year	2021-06
その他のタイトル	行動認識機械学習データセット収集のためのクラウドソーシングの研究
学位授与年度	令和3年度
学位授与番号	17104甲工第526号
URL	http://hdl.handle.net/10228/00008417

Doctoral Thesis

Exploring and Improving Crowdsourced Data
Labeling for Mobile Activity Recognition

Nattaya Mairittha

Graduate School of Engineering
Kyushu Institute of Technology

June, 2021

Contents

Chapter 1	Introduction	6
1.1	Background and Motivation	7
1.2	Problem Statement	8
1.2.1	Response Quality and Quantity	9
1.2.2	Participant Motivation	9
1.2.3	Smartphones as a Research Instrument	10
1.3	Research Questions	10
1.4	Key Contributions	11
1.5	Thesis Outline	11
Chapter 2	Related work	14
2.1	Mobile Crowdsensing Systems	14
2.1.1	Introduction	14
2.1.2	Mobile Crowdsensing Applications	15
2.1.3	Mobile Crowdsensing Challenges	17
2.2	Challenges of Crowdsourced Labeling for Activity Recognition	18
2.3	Active Learning and Activity Recognition	20
2.4	Incentivizing Participation and Gamification	21
2.5	On-Device Deep Learning	22
2.6	Decentralized Machine Learning	23
Chapter 3	Achieving High-Quality Crowdsourced Datasets in Mobile Activity Recognition	25
3.1	Abstract	25
3.2	Introduction	25
3.3	Method	27
3.3.1	Proposed Gamified Active Learning	28
3.3.2	Inaccuracy Detection	30
3.4	End-to-End System	32
3.4.1	A Task Creation Interface	33
3.4.2	A Smartphone Task App	34
3.4.3	A Crowdsourcing Task Manager	38

3.5	Preliminary Evaluation	38
3.5.1	Experimental setup	39
3.5.2	Activity Recognition	40
3.5.3	User engagement	41
3.5.4	Results	41
3.6	Crowdsourcing Deployments	42
3.6.1	Procedure	43
3.6.2	Participants	45
3.7	Evaluation and Results	45
3.7.1	Data Overview	46
3.7.2	Activity Recognition	47
3.7.3	Worker Engagement	52
3.7.4	Inaccurate Data	56
3.8	Discussion and Conclusion	57
3.8.1	Addressing Research Goals	57
3.8.2	Limitations and Future Directions	59
Chapter 4	On-Device Deep Learning Inference for Activity Data Collection	62
4.1	Abstract	62
4.2	Introduction	62
4.3	Method	64
4.3.1	Build an LSTM Model for On-Device Inference	64
4.3.2	Collect Sensor Data and Activity Labels	68
4.3.3	Provide Estimated Activities as Feedback	70
4.4	Experimental Evaluation	71
4.4.1	Experimental Setup	72
4.4.2	Data Description	72
4.4.3	Activity Recognition Using Smartphone Sensors	72
4.5	Results	75
4.5.1	Quality of Collected Activity Data	75
4.5.2	Quantity of Collected Activity Data	78
4.6	Discussion and Future Directions	78
Chapter 5	On-Device Deep Personalization for Activity Data Collection	82
5.1	Abstract	82
5.2	Introduction	83
5.3	Preliminaries	84
5.3.1	Mobile Activity Recognition with Deep Learning	84
5.3.2	Transfer Learning and Fine-Tuning	85
5.3.3	On-Device Personalization	85
5.4	Method	86

5.4.1	Overview	86
5.4.2	Dataset	87
5.4.3	Network Architecture and Implementation	88
5.4.4	Classification Performance	90
5.4.5	Performance on a Smartphone	91
5.4.6	Performance Optimization with Model Pruning	93
5.5	Systems Implementation	94
5.6	Experiments	98
5.7	Activity Recognition: Evaluation and Results	99
5.7.1	Data Preprocessing	99
5.7.2	Evaluation Method	100
5.7.3	Results	101
5.8	Discussion and Future Directions	103
Chapter 6	Discussion	106
6.1	Response Quality and Quantity	106
6.2	Participant Motivation	108
6.3	Smartphones as a Research Instrument	109
6.4	Future Directions	111
Chapter 7	Conclusion	114
Bibliography		115

Abstract

In this thesis, we propose novel methods to explore and improve crowdsourced data labeling for mobile activity recognition. This thesis concerns itself with the quality (i.e., the performance of a classification model), quantity (i.e., the number of data collected), and motivation (i.e., the process that initiates and maintains goal-oriented behaviors) of participant contributions in mobile activity data collection studies. We focus on achieving high-quality and consistent ground-truth labeling and, particularly, on user feedback's impact under different conditions. Although prior works have used several techniques to improve activity recognition performance, differences to our approach exist in terms of the end goals, proposed method, and implementation. Many researchers commonly investigate post-data collection to increase activity recognition accuracy, such as implementing advanced machine learning algorithms to improve data quality or exploring several pre-processing ways to increase data quantity. However, utilizing post-data collection results is very difficult and time-consuming due to dirty data challenges for most real-world situations. Unlike those commonly used in other literature, in this thesis, we aim to motivate and sustain user engagement during their on-going-self-labeling task to optimize activity recognition accuracy. The outline of the thesis is as follows:

In chapter 1 and 2, we briefly introduce the thesis work and literature review. In Chapter 3, we introduce novel gamified active learning and inaccuracy detection for crowdsourced data labeling for an activity recognition system (CrowdAct) using mobile sensing. We exploited active learning to address the lack of accurate information. We presented the integration of gamification into active learning to overcome the lack of motivation and sustained engagement. We introduced an inaccuracy detection algorithm to minimize inaccurate data.

In Chapter 4, we introduce a novel method to exploit on-device deep learning inference using a long short-term memory (LSTM)-based approach to alleviate the labeling effort and ground truth data collection in activity recognition systems using smartphone sensors. The novel idea behind this is that estimated activities are used as feedback for motivating users to collect accurate activity labels.

In Chapter 5, we introduce a novel on-device personalization for data labeling for an activity recognition system using mobile sensing. The key idea behind this system is that estimated activities personalized for a specific individual user can be used as feedback to motivate user contribution and improve data labeling quality. We exploited finetuning

using a Deep Recurrent Neural Network (RNN) to address the lack of sufficient training data and minimize the need for training deep learning on mobile devices from scratch. We utilized a model pruning technique to reduce the computation cost of on-device personalization without affecting the accuracy. Finally, we built a robust activity data labeling system by integrating the two techniques outlined above, allowing the mobile application to create a personalized experience for the user.

To demonstrate the proposed methods' capability and feasibility in realistic settings, we developed and deployed the systems to real-world settings such as crowdsourcing. For the process of data labeling, we challenged online and self-labeling scenarios using inertial smartphone sensors, such as accelerometers. We recruited diverse participants and conducted the experiments both in a laboratory setting and in a semi-natural setting. We also applied both manual labeling and the assistance of semi-automated labeling. Additionally, we gathered massive labeled training data in activity recognition using smartphone sensors and other information such as user demographics and engagement.

Chapter 6 offers a brief discussion of the thesis. In Chapter 7, we conclude the thesis with conclusion and some future work issues.

We empirically evaluated these methods across various study goals such as machine learning and descriptive and inferential statistics. Our results indicated that this study enabled us to effectively collect crowdsourced activity data. Our work revealed clear opportunities and challenges in combining human and mobile phone-based sensing techniques for researchers interested in studying human behavior in situ. Researchers and practitioners can apply our findings to improve recognition accuracy and reduce unreliable labels by human users, increase the total number of collected responses, as well as enhance participant motivation for activity data collection.

Preface

This thesis contains published articles, as detailed in the following part. I appreciate the positive and constructive comments and suggestions of the co-authors on my thesis. My gratitude is detailed in the Acknowledgement chapter and reflected in its scientific use.

First of all, I clarify my contributions. I started the work presented in this thesis and got involved in most of the work. I designed the study design and decided on research questions. I took responsibility for the software system developments and data analyses. I was also responsible for dealing with ethical clearance, recruiting study participants, and analyzing the data. Finally, I arranged an article for submission, worked on the peer-review feedback, and subsequently improved the articles. I contributed closely with the co-authors through every step of the process. My co-authors gave feedback on the proposal and study design, supported some system development phases, and reviewed the work.

Due to my research's collaborative background, I decided to use the scientific plural instead of the singular to write this thesis. I refer to the publications of this thesis by Arabic numerals given below (1-6). I exclude some sections of these publications to avoid redundant information. A brief introduction regarding the context of the thesis preceded each publication.

1. Nattaya Mairittha and Sozo Inoue. Crowdsourcing system management for activity data with mobile sensors. In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 85–90. IEEE, 2019.
2. Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep learning inference for efficient activity data collection. *Sensors*, 19(15):3434, 2019.
3. Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Optimizing activity data collection with gamification points using uncertainty based active learning. 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*, pages 761–767, 2019.
4. Nattaya Mairittha and Sozo Inoue. Robust activity data collection with on-device recognition using long short-term memory. *研究報告高齢社会デザイン (ASD)*, 2019(12):1–8, 2019.
5. Nattaya Mairittha and Sozo Inoue. Improving annotation for activity recognition with active learning and gamification. *研究報告高齢社会デザイン (ASD)*, 2019(5):1–

- 8, 2019.
6. Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Improving activity data collection with on-device personalization using fine-tuning. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 255–260, 2020.
 7. Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep personalization for robust activity data collection. *Sensors*, 21(1):41, 2021
 8. Nattaya Mairittha, Tittaya Mairittha, Paula Lago, and Sozo Inoue. Crowdact: Achieving high-quality crowdsourced datasets in mobile activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(1):1–32, 2021.

Acknowledgements

The work introduced in this thesis was not accomplished in isolation. I am very grateful for the consistent support of many people contributing to this work over the past years.

First and foremost, I would like to sincerely thank my primary supervisor, Professor Sozo Inoue, for continuous support, valuable advice, optimism, understanding, and patience during my research study. I cannot thank you enough for putting his trust in me and allowing me to work at one of the best laboratories in Japan. The work done in this thesis benefited tremendously from his direction. Working with him closely in the past four years has helped me grow both professionally and personally. I would also like to express my sincere gratitude to my co-supervisor, Associate Professor Mitsunori Mizumachi, for being part of this thesis in a supervisory role. His unconditional support and assistance in every stage helped me fulfill this thesis. Further, I would like to thank Professor Kenichi Asami and Professor Takeshi Ikenaga for giving me such valuable time and serve as my committee members. Additionally, I am also indebted to Matsuoka Sensei, the international student counselor of Kyutech. I am so thankful for all of her assistance during the master and doctoral course period she has done for me.

Second, I would like to say a massive thank you to everyone at Sozo Laboratory of Kyutech for creating a home-like atmosphere for me. I have been fortunate to work closely with many talented people and see their ability up close, which is truly inspirational. I am so thankful and humbled for the kindness, memories, and opportunities this lab has given me during these past years. It really helped me to keep going forwards. I will never forget the experience of being with friendly people when we had a very pleasant time and went through a tough time together. I cannot explain what it has been like to stay here with all of you; the whole journey has been surreal. I hope that our good relationships and discussions will continue for many years to come.

Lastly and most importantly, support from my family and close friends has been invaluable. I owe my deepest gratitude to all of you for the unconditional love. I am so grateful that I always have good people by my side, and it makes me think I did something right and could get through the hard times.

Chapter 1

Introduction

Technologically-advanced mobile device accessibility provides researchers the chance to recognize human behavior in everyday situations. This research tool enables us to systematically capture human action in natural settings rather than observing human behavior through extensive surveys or artificial laboratory settings. Thus, the measurements obtained would provide researchers insights into the problem of interest. This method can be used across a wide range of research fields and applied to various applications. For example, in the field of Human-Computer Interaction (HCI) to perceive how people use and experience new technologies in daily life, in Medical Sciences to assess the pain level of patients experienced, in Psychology to infer the nature of human well-being, in Economics to predict customer decisions, as well as in Nursing Care to create automatic nursing care services to improve the efficiency of nursing care works. Although originating from different research fields and applications, these cases are directed to the same goal: understanding human behavior and experience by repeatedly gathering human-labeled data in situ using mobile devices.

Researchers commonly employ mobile devices to capture two valuable different data streams. First, talking the sensor data embedded within these devices to study participants' context. Second, asking participants to frequently and actively describe data on events that cannot be reliably collected using the aforementioned sensor streams. This process is known as label collection (i.e., data labeling or data annotation) for mobile activity recognition. Label collection is a vital part of preparing training data for learning tasks to design and evaluate activity recognition systems. Consequently, the quality of labeled data can significantly impact the performance of the derived systems. Label collection using mobile devices yields many considerable advantages over existing methods, such as observing changes in participant experiences over time and context and reducing participants' cognitive overload to recall past experiences. Since data collection's responsibility relies on participant contributions rather than researchers, collecting the reliability of the participants' participation is important and increasingly considered by researchers. While increasing the number of participant contributions has been explored over the years, these contributions' quality remains scarce.

This thesis, we propose novel methods to explore and improve crowdsourced data labeling for mobile activity recognition. This thesis concerns itself with the quality (i.e., the performance of a classification model), quantity (i.e., the number of data collected), and motivation (i.e., the process that initiates and maintains goal-oriented behaviors) of participant contributions in mobile activity data collection studies. We focus on achieving high-quality and consistent ground-truth labeling and, particularly, on user feedback’s impact under different conditions.

This chapter introduces the motivation underlying my doctoral thesis and presents the existing research issues and challenges. Furthermore, it describes my thesis’ s goals to address those research issues and how to achieve them. Finally, it summarizes and reviews the organization of my thesis.

1.1 Background and Motivation

Labeling collection for activity recognition with smartphone sensors refers to the process of segmenting and assigning labels to data gathered from smartphone sensors. The data are related to a person’ s behavior, activity, health or mood state, at different timestamps [80]. Activity or behavior recognition is mostly implemented using supervised learning algorithms. The training of these supervised algorithms requires labeled data, or “ground truth”. Incomplete or inaccurate labeling may result in classification errors that lead to unreliable systems; thus, achieving high-quality labels is a critical requirement. Data labeling using smartphone sensors can be done in various ways, depending on the type of data being labeled and the situations in which the activity is being performed and observed. We categorize the approaches to data labeling concerning four different criteria: when, who, where, and how is the data labeled.

First, data can be labeled when the activity of concern is being performed (online) [31], or it can be labeled later (offline) [19]. Both ways impose challenges. Labeling while the activity is being performed is difficult because it requires an accurate timestamp. On the other hand, offline data labeling is subject to response bias challenges since it is difficult to recall past experiences.

Second, data labeling can be done either by the individual performing the activity (self-labeling) [31, 149] or by an observer [65]. Self-labeling has the advantage of being less invasive and less costly, so it can be used in natural environments. Additionally, some instances can only be assessed by the person experiencing it, such as personalized emotion recognition. Contrarily, observers might give more accurate labels in other instances, e.g., when individuals might be impaired to or have great difficulty labeling their data. For example, in real nursing activities, nursing the patient has the highest priority, so, when self-labeling, there are many missing labels and inaccurate timestamps. However, recruiting observers to label large amounts of data can be costly, or the observers may not have enough context to accurately label the data. In sum, labeling data either by the

individual or by an observer is a challenging task.

Third, smartphone sensor data labeling can be performed in three scenarios based on the level of naturalness with which the data is collected. First, we examine a laboratory scenario [45], where users perform activities systematically in a controlled environment with previously defined steps. Second, semi-natural scenarios [65, 80, 128], where users perform their daily activities as usual but are asked to complete the activities from the experiments at least once to ensure that the activities have been performed. Third, a natural scenario [15], where users perform their daily activities commonly without intervention in their behavior by the application. Literature shows that laboratories typically facilitate data labeling and tend to generate more accurate classification models. However, the models produced in laboratory settings lose accuracy when applied to real contexts owing to the diversity of users' behavior. Contrarily, the models produced with natural datasets tend to be more generic, can be applied to groups of people with similar behaviors, and provide feasible solutions to scale data collection. Still, naturalistic data collection is more complicated.

Finally, data labeling can be undertaken by manual [74], fully automated [164], or semi-automated [62] mechanisms, depending on specific cases (e.g., reliability, cost, and time). While employing domain experts to manually label data typically results in more reliable labeling, it can be costly and time-consuming. Nevertheless, the use of fully automated labeling tools can reduce time and costs compared to domain experts; still, the resulting labeling may not be as accurate as those produced by a domain expert, and it still requires a pre-trained model. Alternatively, the use of semi-automated labeling allows the use of machines in combination with human domain expertise, such as active learning. Although semi-automated labeling has led to the reduction of the human labeling burden without sacrificing the model's performance, it is still inefficient under some circumstances. For instance, the acquisition of a large number of human labeling is impractical or entirely unfeasible.

Overall, label collection for activity recognition with smartphone sensors has various challenges according to the criteria outlined above. This thesis challenges the online and self-labeling scenarios using inertial smartphone sensors, such as accelerometers. We employ the participants both in a laboratory setting and in a semi-natural setting such as crowdsourcing. We also apply semi-automated labeling assistance such as using active learning.

1.2 Problem Statement

Although researchers in the broader scientific community are increasingly considered label collection for mobile activity recognition, researchers employing this method face many challenges when designing their studies. We classify remained challenges into three main categories and elaborate on the challenges addressed in this thesis below.

1.2.1 Response Quality and Quantity

The better of quality and quantity response of the labeling, the more time it demands. Human labeler needs additional time to label data precisely and collect sufficiently reliable data, but time is undoubtedly a limited resource for internal labeling. Although contacting outsourcing companies specializing in training data preparation instead of recruiting temporary employees or relying on a crowd might give more high-quality results, it is more expensive than crowdsourcing. As a result, the current literature on label collection has primarily focused on obtaining sufficient data from crowdsourcing. Although asking others to label data gives fast results and cost-saving and has the ability to evaluate participants' skills, the risk of obtaining a low-quality dataset is the main one. Participants whose daily income depends on the number of finished tasks may fail to follow task instructions to get as much work done as possible. Sometimes mistakes in data labeling can occur due to an inexperienced or unskilled participant in the field. This situation causes the inconsistent quality of the labeled data. Furthermore, crowdsourcing and outsourcing to individuals require the need to organize workflow. For instance, if researchers have sensor data, they must create a task template (e.g., a mobile app interface) and ensure it is intuitive for labeling tasks. In sum, we categorize the problem labeled training data in a supervised learning setting concerning three different criteria:

- *Insufficient quantity of labeled data*: there is often insufficient training data available to apply traditional processes when machine learning techniques are initially used in new industries or applications.
- *Insufficient subject-matter expertise to label data*: creating a usable training data set can quickly become costly when labeling training data requires specific relevant expertise (e.g., in nursing care applications of machine learning).
- *Insufficient time to label and prepare data*: preparing data sets is most of the time required to implement machine learning. When a research field has to do with rapidly evolving problems, it is often impossible to gather and prepare data quickly enough for results to be useful in real-world applications (e.g., in fraud or anomaly detection applications).

1.2.2 Participant Motivation

Researchers or other internal experts ordinarily intend to perform a great job because they are the ones who will be working with a labeled dataset. On the other hand, an on-demand workforce exhibits early enthusiasm but often loses interest and drops out over time. However, researchers usually rely on the data collected by study participants instead of in-house labeling owing to a limited resource, understanding participant motivation is crucial. The use of crowdsourcing or outsourcing labeling for mobile activity recognition

introduces many motivation challenges and uncertainties. Due to the frequency of labeling for each activity change, participant annoyance may increase over time, potentially reducing the collected information's reliability. Establishing motivation for long hours or over multiple days is much more challenging when relying on participant data. A study observing an extraordinary event, such as nursing activities, will expect a longer duration to capture sufficient data points. Asking numerous pieces of information in a day will quickly reduce participant motivation, making it infeasible to run for longer durations of time. Furthermore, the nature of the complexity of mobile activity recognition and its interpretation affects participant motivation.

1.2.3 Smartphones as a Research Instrument

Using a wide range of sensor-rich and highly interactive smartphones consolidated with the capability to deploy custom-developed software to these devices, researchers have the potential to adopt smartphones as advanced research tools. Additionally, these devices can monitor the participants' lives due to integrating human-labeled data and automated sensor data collection. Therefore, researchers increasingly consider smartphones as a scientific instrument. Mobile technologies have allowed new opportunities for a widespread label collection in activity recognition while simultaneously leading to new methodological and technological challenges. Examples of methodological challenges include assuring consistent data quality across devices, designing effective user interfaces, exploring novel sensing modalities and strategies for data labeling and personalization, as well as examining the performance of these solutions in real-world settings with diverse populations. Similarly, technological challenges include device-to-device communication across devices and software systems, computational constraints reduction, extreme energy efficiency, model optimization, federated learning, and diverse input techniques between devices.

1.3 Research Questions

Regarding my problem statements, several issues of human-labeled data collection focusing on mobile activity recognition aspects must be solved to design and build the system that can collect high-quality datasets. Based on these issues, I particularly set out to answer the following research questions:

- **RQ1:** How can researchers improve the quality and increase the number of participant contributions in data collection for mobile activity recognition studies?
- **RQ2:** How can researchers motivate participants to participate and carry out data labeling tasks to their best abilities?
- **RQ3:** How can researchers employ smartphones as an effective research instrument in data collection for mobile activity recognition studies?

1.4 Key Contributions

This thesis contributes to the methodological application of smartphones in data collection for activity recognition studies. Specifically, it introduces and evaluates novel techniques to assess participants' contributions in data collection for mobile activity recognition studies and provides suggestions and insights to researchers employing this method. Researchers can employ our findings to improve the reliability and contextual diversity of participant contributions and increase the total number of contributions.

This thesis aims to improve the collection of human-labeled data in activity recognition, strengthening smartphones' role as valuable research instruments. The research goals and contributions are described below.

1. This thesis proposes novel gamified active learning and inaccuracy detection for crowdsourced data labeling for a mobile activity recognition system to achieve high-quality crowdsourced datasets by overcoming three main issues in crowdsourced labeling: lack of accurate information, loss of motivation and engagement, and inaccurate data.
2. This thesis proposes using on-device deep learning inference instead of using cloud-based deep learning inference to alleviate the self-labeling effort for a mobile activity recognition system to address the main limitation of adopting the cloud-based approach.
3. This thesis proposes a novel on-device personalization using resource-constrained mobile devices and mobile optimization to create individualized recommendations and operate effectively on mobile devices for data labeling for a mobile activity recognition system.
4. This thesis deployed the proposed systems to realistic settings demonstrating their capability and feasibility. They gathered real activity labels with smartphone sensors. They empirically evaluated the systems' quality by comparing the proposed conditions with baseline conditions using various techniques such as machine learning and descriptive and inferential statistics. Lastly, they reported that all proposed systems could achieve the goals we set out.

This thesis discusses the results, challenges, limitations, as well as implications of the proposed system on the design of efficient human-labeled data collection for mobile activity recognition.

1.5 Thesis Outline

This thesis consists of seven chapters, including this one. The remainder of this thesis is organized as follows:

Chapter 2 surveys the critical researches in achieving my thesis goals. The first three subsections provide an overview of the related work in crowdsourced labeling for activity recognition, active learning and activity recognition, and gamification that contributed to Chapter 3. The two following subsections present a literature review of the current research in on-device deep learning and decentralized machine learning that contributed to Chapter 4 and Chapter 5.

Chapters 3–5 present original work conducted per the objectives of this thesis.

Chapter 3 describes a field study investigating crowdsourced data labeling for an activity recognition system. This study applied active learning and inaccuracy detection to crowdsourcing to assess and improve the quality of labels provided by participants. This knowledge can be used to effectively collect crowdsourced activity data and demonstrate the validity of crowdsourced data for a real application.

- Nattaya Mairittha and Sozo Inoue. Crowdsourcing system management for activity data with mobile sensors. In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 85–90. IEEE, 2019.
- Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Optimizing activity data collection with gamification points using uncertainty based active learning. 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*, pages 761–767, 2019.
- Nattaya Mairittha and Sozo Inoue. Improving annotation for activity recognition with active learning and gamification. *研究報告高齢社会デザイン (ASD)*, 2019(5):1–8, 2019.
- Nattaya Mairittha, Tittaya Mairittha, Paula Lago, and Sozo Inoue. Crowdact: Achieving high-quality crowdsourced datasets in mobile activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(1):1–32, 2021.

Chapter 4 reports on a user study in which we quantify the effect of using on-device deep learning for activity collection on participant response quality and quantity. This study discussed the implications of the research outcomes and identified future research opportunities in exploiting on-device deep learning for mobile activity recognition before making it more advanced in the next chapter.

- Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep learning inference for efficient activity data collection. *Sensors*, 19(15):3434, 2019.
- Nattaya Mairittha and Sozo Inoue. Robust activity data collection with on-device recognition using long short-term memory. *研究報告高齢社会デザイン (ASD)*, 2019(12):1–8, 2019.

Chapter 5 explores the impact of on-device personalization on the accuracy of human contributions in achieving high-quality and consistent ground-truth labeling for mobile activity recognition. This study employed the idea of deep transfer learning for personalization, investigated model optimization and computational constraints reduction, as well as examined the performance of these solutions in real-world settings.

- Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Improving activity data collection with on-device personalization using fine-tuning. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 255–260, 2020.
- Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep personalization for robust activity data collection. *Sensors*, 21(1):41, 2021.

Following the presentation of our original research contributions, **Chapter 6** discusses this thesis, especially how it was researched, built, evaluated, and solved. Furthermore, we discuss the remaining limitations and challenges that stimulate future research. Finally, **Chapter 7** presents a summary of the thesis and offers concluding remarks.

Chapter 2

Related work

In this section, we introduce five key ideas that drive this thesis: (1) a survey on mobile crowdsensing systems: applications, challenges, solutions, and opportunities; (2) practical challenges in crowdsourced labeling for activity recognition; (3) implementation of active learning and minimization of the effort of data labeling in activity recognition applications; (4) exploiting gamification to incentivize participation in ubiquitous crowdsourcing; (5) adopting on-device deep learning for activity data collection; and (6) study on decentralized machine learning and applications to mobile activity recognition.

2.1 Mobile Crowdsensing Systems

2.1.1 Introduction

With mobile sensing and mobile internet technology, a new sensing paradigm called Mobile Crowd Sensing [46] has been studying many works [52, 85]. This paradigm leverages many individuals with mobile devices capable of sensing and computing (such as smartphones, tablet computers, wearables) for large-scale sensing. In short, mobile crowdsensing can be considered as crowdsourcing where the data provided by the crowd workers is their sensing capabilities. Using this paradigm, individuals can collect multi-modal data streams from the surrounding environment using their mobile devices and share the data with existing communication support (e.g., wireless network protocols). They can then extract information to measure, map, analyze, or estimate any processes of common interest. Consequently, taking advantage of the ubiquitous crowdsourcing and robust mobile computing devices (particularly smartphones) in recent years, it has become an appealing method to businesses that want to obtain data without making large-scale investments.

Current mobile crowdsensing applications can be broadly classified into personal sensing and community sensing based on the type of phenomena being observed [120, 161]. Regarding personal sensing applications, the phenomena concern an individual, for instance, observing physical activity patterns (e.g., running, walking, exercising) of an individual for personal activity recoding or healthcare purposes or recognizing an individual's facial expressions to determine his or her emotions. Contrarily, community sensing involves

observing large-scale phenomena that cannot easily measure by each individual. For example, intelligent transportation system applications [157] may require road traffic congestion monitoring and measurement. This phenomenon can be measured accurately only when massive numbers of people provide speed information from their daily commutes, which are then aggregated to monitor road traffic congestion or provide real-time navigation in urban areas. Based on the type of involvement from the users, community sensing can also be classified into two types: participatory sensing [21] and opportunistic sensing [85]. While participatory crowdsensing requires the users to voluntarily and actively participate in contributing information (e.g., reporting an issue with roadworks), opportunistic crowdsensing involves automated systems with less user involvement (e.g., continuous location sampling without the user's explicit knowledge). However, there are many challenges for mobile crowdsensing applications in practice, which should be further examine.

In the rest of this section, we present a brief overview of the existing mobile crowdsensing applications, describe their various challenges, and discuss possible solutions. Lastly, we introduce new challenges to be explored in future research.

2.1.2 Mobile Crowdsensing Applications

This section briefly discusses existing mobile crowdsensing applications, which provide a reason for drawing various research challenges. Mobile crowdsensing applications can be classified into several types based on the type of phenomena being measured. Here, we provide an example of common mobile crowdsensing applications: healthcare, environmental, infrastructure, and social. However, there have been many other applications such as smart cities, tourism, sports, public safety, and the military.

Smartphones can connect patients with medical services through mobile communications networks for sensing and diagnostic capabilities in healthcare applications. Healthcare monitoring employs sensors embedded within these devices to monitor patient vital signs both locally and remotely. Since it provided improved patient care through the early detection of adverse events of high-risk medications or health conditions, it can influence patients' behavior to improve their health [153]. The use of crowdsensing in healthcare applications has been the subject of study of many works. For example, Poh et al. [123] proposed sensor earphones and mobile application for non-obtrusive health monitoring. Hanson et al. [57] introduced wireless body area sensor network technology for motion-based health assessment. Abualsaud et al. [2] proposed an efficient framework for evaluating the power-accuracy trade-off for EEG-based compressive sensing and classification techniques in the context of epileptic seizure detection in wireless tele-monitoring, and they also presented [3] an ensemble classifier for epileptic seizure detection for imperfect EEG data.

Environmental mobile crowdsensing applications adopt mobile phones as environmental sensing platforms that support community action to drive positive societal change.

A combination of mobile smartphone sensors and fixed localized sensors can efficiently monitor context from personal and local perspectives. As such, extensive research and applications in environmental sensors have been in demand in recent years. Examples of environmental applications include mobile monitoring of air pollution in cities [156], monitoring the water level of the river [142], and monitoring wildlife and their habitats [118]. For example, Wallace et al. [156] used mobile monitoring to improve spatial coverage of pollution concentrations over the city of Hamilton, Ontario, and enhance knowledge of the short-term bursts of pollution to which the population is exposed. Sulistyowati et al. [142] proposed long-term monitoring systems for wildlife and their habitats in the Southern Ocean and around Indian Research Stations in Antarctica.

Infrastructure comprises the basic physical and organizational structures and facilities needed for the operation of a society or enterprise, such as buildings, roads, and power supplies [87]. Infrastructure applications can be defined as the fundamental components of related systems to improve people's daily lives. For instance, evaluating civil infrastructures and critical facilities (e.g., schools, nursing homes, and hospitals) after natural disasters is very important in our lives. One of the most well-known infrastructure applications is traffic monitoring [14, 148]. Traffic monitoring exploits Global Positioning System (GPS) and smartphones to allow critical information about traffic conditions. This monitoring can sense a driver's behavior, unexpected traffic events, risky vehicle activities, or aggressive driving. For example, Basudan et al. [14] propose a privacy-preserving protocol to enhance security in vehicular crowdsensing-based road surface condition monitoring system using fog computing. Thiagarajan et al. [148] presented a system for travel time estimation using sensor data to address energy consumption and sensor unreliability issues. These studies exhibited promising future research directions regarding employing crowdsensing to allow reliable transportation services.

Social applications can be classified as social networking and social sensing information. In social networking applications, the users can share their information by using social networks (e.g., Facebook, Twitter, LinkedIn) and media sharing sites (e.g., Instagram, YouTube, Snapchat) [95]. Contrarily, social sensing applications gather data on personal activity (e.g., personal health, location, pictures and videos) and transfer it to the remote server for further processing. Accordingly, users in such a system can share their sensitive information only among specific groups of friends or community for privacy purposes [51]. For example, Guo et al. [51] integrated mobile computing and social networks to build a group-aware system that delivers assistance during several group activity organizational stages. Bulut et al. [20] presented a mobile crowdsensing system for coffee shop wait-time monitoring. The system utilized continuous streams of accelerometer data given by hundreds of users at a coffee shop to monitor and estimate the waiting time to enter a coffee shop.

2.1.3 Mobile Crowdsensing Challenges

Although mobile crowdsensing has excellent potential and offers many opportunities and applications, it also has several challenges that need to be considered before deploying such systems on a large scale. We discuss three major research challenges faced by mobile crowdsensing applications: (1) user perception, (2) privacy and security, (3) data accuracy, and (4) data size. However, researchers are still faced with other obstacles to widely perform experiments despite the considerable demand for mobile crowdsensing applications for smartphones, such as localized analytics, resource limitations (e.g., energy and bandwidth computation), aggregate analytics, and a unifying architecture

User perception

The important implication for human involvement is incentives. Thus, one of the significant research challenges in mobile crowdsensing applications is exploring an appropriate incentive mechanism that encourages users to participate in such a system. Mobile crowdsensing applications commonly face the availability of a sufficient number of participants for the required application. In response to this requirement, researchers commonly employed incentive strategies, such as monetary rewards, to increase the number of participants [170]. Reducing the effect of operating these applications on smartphones' performance (e.g., optimizing energy consumption, processing needs, and network requirements) also helps maintain users' interest in participation in mobile crowdsensing applications. When it comes to active participation, user participation in this situation requires more user involvement, which is more challenging. Xiao et al. [162] revealed the barriers hampering mobile crowdsensing applications' scale-up. They offered their initial thoughts that people can help lower the barriers when a minimum effort is required without extra cost.

Privacy and security

Another major research challenge in mobile crowdsensing is the authenticity and integrity of the data obtained from diverse user populations participated in the system. Privacy and security are essential in applications that collect sensitive information related to participants (e.g., by tracking a user's current location). While authentication and integrity verification of the information provided is critical since it leads to decision making and may impact the whole sensing performance, this sensitive information is also critical as it may affect the user's privacy. Many researchers have touched on these challenges [29, 72]. For example, Khan et al. [72] and Christin et al. [29] studied how to protect the participant's privacy without sharing his/her sensitive information while still enabling mobile crowdsensing applications. Furthermore, when malicious participants act dishonestly or unfairly to maximize profit and minimize effort, i.e., cheating behavior (e.g., contribute with inaccurate sensor data), this could affect the integrity of the data collected from the system. This situation produces a severe problem that could lead to a lack of trust in the

mobile crowdsensing application. Some researchers have studied finding new approaches to solve this problem [107, 143]. For example, Manshae et al. [107] presented a list of works highlighting the application of game theory in addressing different forms of security and privacy problems in computer networks and mobile applications. Sun et al. [107] presented a secure and privacy-preserving object-finding system via mobile crowdsourcing to ensure strong object security that only the object owner can discover his/her lost object's location and offer location privacy to mobile users involved.

Data accuracy

Although increasing the quantity of collected data improves monitored event accuracy, gathering sensor data from crowd users might overwhelm the communication network and remote servers. Hence, many researchers carefully marked the tradeoff between the data accuracy and the overloading of the communication network and servers, such as the work in [14, 138]. Shin et al. [138] and Basudan et al. [138] identified devices that are likely to produce accurate sensing data, aggregate global centralized data, and analyze these devices' spatiotemporal mobility patterns. The crowd data is often collected from different smartphone models from different vendors, making sensitivity and noise immunity. Thus, their study provided useful information that can improve data accuracy in mobile crowdsensing applications. Although data accuracy challenge is common to mobile crowdsensing and more traditional IoT networks, mobile crowdsensing faces further specific issues. For example, there is more limited control on a wide variety of devices, as mentioned previous paragraph, or data accuracy in mobile crowdsensing can be involved in creating an inaccurate dataset intentionally by malicious users, as discussed in the previous "Privacy and security" section.

Data size

The large scale of mobile crowdsensing results in a large amount of data traffic that may damage the network. Contrary to traditional networking in the Internet of things (IoT) with completely automated sensor transmissions, mobile crowdsensing can sometimes generate unforeseen traffic owing to unexpected human participation. Consequently, remarkable methods need to be considered to minimize the amount of traffic. For instance, advanced big data techniques and data analytics algorithms can be employed to effectively manage the enormous amount and variety of sensed data [114]. Additionally, researchers have also explored other techniques to achieve data size challenges, for example, proportional fair traffic splitting and aggregation in heterogeneous wireless networks [140] or local data aggregation and processing on personal mobile devices [171].

2.2 Challenges of Crowdsourced Labeling for Activity Recognition

Crowdsourcing with ubiquitous technologies is increasingly considered by researchers, particularly for mobile devices [155]. With crowdsourcing, it is possible to solve a problem

using a mobile device and its sensors to collect data. Studies on crowdsourced labeling for activity recognition technology with smartphone sensors are becoming increasingly popular [6, 23, 139, 168]. Using crowdsourcing to gather activity and contextual data in natural settings has a considerable advantage over a controlled data collection method in laboratory settings. Specifically, the collected data are more diverse, naturalistic, and representative of users' real-life behaviors. Moreover, crowdsourcing potentially allows the production of a larger number of activity labels in a timely manner. Despite the various benefits of crowdsourced labeling for activity recognition with smartphones, there are some serious drawbacks.

First, researchers question the accuracy of the collected labels and the reliability of crowdsourced data [4, 13, 62]. Basiri et al. [13] regard that crowdsourced data is contributed by the public. Some workers with little experience and data expertise might have contributed to the perception of the unreliability of this data source. Additionally, Hossain et al. [62] recently explored how different social and educational backgrounds may lead to unreliable and noise-prone labels. Particularly, understanding the data collected from the sensors of an inertial measurement unit (IMU) in the activity recognition domain, such as smartphone sensors, is difficult for human labelers who are typically not experts in the field [168]. Hence, crowdsourced data generate an unacceptably high rate of labeling errors. Low trust in the accuracy of the crowdsourced data is also produced by the inability to inspect the users' input [4]. This aspect contributes to a lack of complete control over the participants and leads to inaccurate information when compared to lab settings. Such issues have impeded the adoption of crowdsourced data in several projects.

A second drawback is the lack of sustained engagement of crowd workers. Crowdsourced labeling relies on the number of participants who are willing to devote and contribute their time and efforts. However, ordinary individuals are not willing to perform the tasks unless there are sufficient incentives. Typically, the crowd consists of two worker types: the triers and the cheaters. The triers, who perform faithfully, may not always deliver the best work. Many researchers studied how crowdsourcing projects have failed because they were unable to gain sufficient attention from the public [108], or because workers gradually lose interest and drop out of the tasks over time [4, 68, 132, 166]. When it comes to online scenarios, collecting personal contextual and activity data is much more challenging. Labels describing the current activity need to be assessed while the activity is on-going or recent to ensure that the dataset is labeled correctly. Human labelers must also start and stop the data capture process manually to avoid inaccurate timestamps, which requires high effort. Consequently, it is inevitable to rely on users and keep them motivated to constantly provide labels. Contrarily, cheaters are workers who act dishonestly to maximize profit and minimize effort. Researchers have explored [38, 88] the lack of sustained engagement and motivation as a reason for cheating behavior (i.e., act dishonestly or unfairly to gain an advantage.) in crowdsourcing scenarios, which affects the task output. For example, the worker shakes the smartphone but annotates a "cleaning" class instead.

However, detecting cheating behavior in activity recognition is challenging. For instance, with smartphones, changing the pattern of cheating behavior is effortless, and it is difficult to determine whether a sensor pattern is likely to be cheating or not.

Although prior works [1, 86, 162] have used crowdsourced data to improve classifier accuracy, differences to our approach exist in terms of the end goals, proposed method, and implementation. Xiao et al. [162] proposed an activity classification framework that keeps the user burden low by leveraging crowdsourced data labels, where natural language processing combined with multi-instance learning is used to handle labeling errors. Abdullah et al. [1] exploited community-scale behavioral patterns to leverage the increasing capacity to gather data at a population-scale towards improving models of human behavior. Lane et al. [86] exploited crowdsourced data to personalize classifiers with data contributed by other similar users to cope with the diverse user populations routinely found in large-scale popular mobile applications. In this study, our challenge is to create strategies to obtain high-quality labels, keep the crowd workers engaged with the labeling task, and prevent inaccuracies.

2.3 Active Learning and Activity Recognition

The key idea behind active learning is that if a machine learning algorithm can select the data to be learned, it can potentially achieve the same or higher accuracy than that of standard supervised techniques with fewer training labels. Hence, active learning is a method in which a model only queries data that can add knowledge and improve performance. This concept is well-motivated in many modern machine learning applications because it helps in minimizing manual labeling efforts and cost, as well as reducing model training computation time. There are several scenarios in which active learners may be able to ask queries, and there are also several different proposed ways of formulating such query strategies to decide which instances are most informative. The three principal sampling strategies that have been considered in the literature are (1) membership query synthesis, (2) stream-based selective sampling, and (3) pool-based sampling. Membership query synthesis [7] is reasonable for many problems, where the learning model generates a data instance from a certain distribution. However, labeling such arbitrary instances can be awkward if the oracle is a human labeler. To resolve these limitations, the stream-based and pool-based scenarios have been proposed for an alternative strategy [8, 91]. A key theory for these scenarios is that labeling an unlabeled instance is effortless. Instead of synthesizing queries, instances can be selectively sampled from a real data distribution. While pool-based sampling has access to all the unlabeled data from which it selects the best, stream-based selective sampling examines unlabeled data sequentially and decides based on some querying strategy whether to request a query its label or not. Both scenarios have been studied in several real-world tasks in machine learning. In most cases, uncertainty sampling is commonly used to evaluate the informativeness of unlabeled data

for both pool-based and stream-based settings. However, there have been many proposed ways of formulating such query strategies in the literature such as query-by-committee, expected model change, expected error reduction, variance reduction, and density-weighted methods. For a detailed review of active learning, we refer an interested reader to [135].

The use of active learning in activity recognition has been the subject of study of many works [5, 9, 60, 62, 96, 141], where sensor (unlabeled) data are readily available or easily obtained, but labeling is challenging. For example, Hossain et al. [62] investigated and analyzed different active learning strategies to scale activity recognition and proposed a dynamic k-means clustering-based active learning approach. Ho et al. [60] addressed the problem of learning and recognizing human daily activities in a dynamic environment using active learning. They showed that utilizing active-learning assistance minimized human effort in relabeling training data. Stikic et al. [141] systematically analyzed several different techniques to significantly reduce the required amount of labeled training data in activity recognition. With the active learning technique, the required amount of training data was considerably reduced while obtaining similar and sometimes better performance than that of standard supervised methods. Longstaff et al. [96] investigated methods of automatically augmenting activity classifiers to improve the performance of an initial classifier. By comparing active learning and several semi-supervised learning methods, they proved that active learning achieved the greatest improvement. Other studies, such as [5,9], described several active learning methods on datasets collected in home settings that they designed for their applicability to the activity recognition task. Both Alemdar et al. [5] and Bagaveyevet al. [9] allowed activity recognition on a large scale and showed a reduction in the number of labeled data required. In this study, we explore two commonly used mechanisms: the pool-based active learning setting and an uncertainty sampling strategy. Unlike the abovementioned literature that used active learning to alleviate the labeling effort and ground-truth data collection in activity recognition pipelines by relabeling the data instances, our research aims to exploit such a strategy to evaluate the activity data collection performance for gamification points.

2.4 Incentivizing Participation and Gamification

Incentives are a commonly used mechanism to entice participation and ensure sufficient data quality in crowdsourcing [4, 30, 112]. Existing work has identified intrinsic (e.g., reputation) and extrinsic (e.g., rewards) as motivations frequently used for incentivizing participation [59]. One of the most popular incentive designs in recent years has commonly been titled as "gamification". Deterding et al. [36] defined gamification as "the use of game design elements in non-game contexts" to improve user experience and engagement. The use of points (i.e., score) as gamification elements proved to be the most effective method in motivating participants [53]. Gamification points are the simplest way to reward users for completing an action or a series of actions. It can create a sense of progression that mo-

tivates continued effort. Gamification in the context of ubiquitous crowdsourcing has been explored [42, 116, 160], while the investigation of gamification aimed at data quality and quantity has been increasing in popularity over the years. [17, 103, 151]. However, the underlying gamification for label collection in activity recognition remains scarce, particularly regarding the role of crowdsourcing. Van et al. [151] showed the potential for gamification in an experience sampling study, describing both its effect on response quantity and quality. Palacin et al. [116] compared two versions (gamified and non-gamified) of a mobile application designed to capture lake ice coverage data to understand the effects of gamification on citizen engagement. The gamified version included gamification elements, such as points and a feedback module. This related work has proved the overall involvement was significantly higher with the gamified application than with the traditional application. The work of Palacin et al. [116] implies there is a statistically significant effect of gamification on participant engagement levels in crowdsourcing as well as data collection and quality. Thus, we set out to exploit a gamification concept for high-quality data in activity recognition. Although a previous study [103] uses gamification points as rewards, they are several key differences from our approach, such as the objective, experimental setup, algorithm behind gamification, and systems development. The major difference between the two studies is that the prior work employed gamification in laboratory settings with the students who knew the purpose and cooperated with the experiment. Contrarily, we implemented gamification as a powerful motivator in crowdsourcing settings. However, Mairittha et al. [98] suggested that specific crowdsourcing challenges such as the difficulty and complexity of tasks, diversity, and digital literacy arise, which of those challenges are addressed in this paper. The proposed CrowdAct approach, which is laid out in Chapter 3, has not been introduced in previous works.

2.5 On-Device Deep Learning

To make capital out of the cloud, we occasionally offload data on small devices such as smartphones and smartwatches to the cloud for storage and processing. For example, physical activity information derived from the accelerometers of wearables is often transferred to and stored in the cloud. The ability to offload complicated tasks from devices with limited computation capabilities to virtual process capacity in the cloud is interesting; however, there are many limitations, for instance, advances in hardware capabilities and privacy threats. With the fast development of the Internet of Things (IoT), combined with improvements in machine learning such as deep learning, technology-based solutions to recognize and model human behaviors automatically are becoming possible. Therefore, mobile computing has been a move to reduce communication latency and network communication while preserving privacy [32, 79, 127].

Deep learning with ubiquitous technologies is increasingly considered by researchers, particularly for mobile devices [63, 83]. With the powerful mobile devices' hardware, it is

possible to exploit deep learning to solve a problem using a mobile device and its sensors to collect data without cloud support. Consequently, the use of deep learning on mobile devices has been researched in many works [12, 84, 101, 159, 163]. The use of Convolutional neural networks (CNNs) and Recurrent neural networks (RNNs) have been the subject of study of many activity recognition applications [113]. Both kinds impose challenges when applied to practical applications owing to the complexity of their architecture. In this study, we deeply explore both CNNs and RNNs due to the suitability for temporal data for building the proposed system blocks.

While the literature mentioned above focused on analyzing deployed different deep learning models on devices to improve its performance, no one has studied the use of on-device deep learning to optimize activity data collection by providing estimated activities as feedback. We present this innovative approach in Chapter 4. Furthermore, while the use of knowledge transfer for on-device deep learning has been the subject of study of some works [70], there are some critical drawbacks concerning deep learning practical on resource-constrained devices [147]. Some present works have been proposed to build deep learning effective on resource-constrained devices, such as model compression [34, 54, 82, 84, 163] and customized hardware design assistance [26, 27, 55] for deep learning. Some of these works are utilized in our work (e.g., layer compression), but mostly they target only the inference phase of deep learning algorithms. Contrarily, we introduce a technique to minimize the complexity of optimizing on-device deep learning inference.

2.6 Decentralized Machine Learning

Machine learning over distributed data collected by many users has essential applications where data privacy is a crucial concern, or central data storage is not a choice. Recently, decentralized machine learning was proposed to solve this problem [76, 110, 152, 167]. The key idea behind this paradigm is that machine learning models are trained on decentralized data. Instead of gathering data on a single server, this paradigm leaves the raw data on the device and manage it using distributed aggregation. The trained models are then transferred to a central component and combined.

Google presented federated learning to overcome this challenge [75, 109]. This approach corresponds to distributed learning. The parameter server controls the current model and commonly distributes it to the users to perform a parameter update and send it back to the server. Then, the server applies all the updates to the central model. The process is then iterated until the model converges. This approach enables multiple researchers to build a standard, robust machine learning model without sharing data, allowing them to address various practical challenges such as data privacy, data access rights, and heterogeneous data access. For this reason, its applications are spread over many industries, including data security, IoT, and telecommunications. For instance, Konevcny et al. [76] explored

federated learning in which users do not send the data they generate to a data center at all, but rather provide part of their computational power to solve optimization problems.

While the field has advanced in recent years, our study remains a relevant and accessible introduction. In Chapter 5, we exploit fine-tuning training where the locally trained models or parameter updates will not be uploaded to the cloud as we already trained and generalized the global model. This solution improves upon the traditional approaches by working better in bandwidth and power-constrained environments and provides a straightforward and effective mechanism for personalization at scale.

Chapter 3

Achieving High-Quality Crowdsourced Datasets in Mobile Activity Recognition

3.1 Abstract

In this study, we propose novel gamified active learning and inaccuracy detection for crowdsourced data labeling for an activity recognition system using mobile sensing (CrowdAct). First, we exploit active learning to address the lack of accurate information. Second, we present the integration of gamification into active learning to overcome the lack of motivation and sustained engagement. Finally, we introduce an inaccuracy detection algorithm to minimize inaccurate data. To demonstrate the capability and feasibility of the proposed model in realistic settings, we developed and deployed the CrowdAct system to a crowdsourcing platform. For our experimental setup, we recruited 120 diverse workers. Additionally, we gathered 6,549 activity labels from 19 activity classes by using smartphone sensors and user engagement information. We empirically evaluated the quality of CrowdAct by comparing it with a baseline using techniques such as machine learning and descriptive and inferential statistics. Our results indicate that CrowdAct was effective in improving activity accuracy recognition, increasing worker engagement, and reducing inaccurate data in crowdsourced data labeling. Based on our findings, we highlight critical and promising future research directions regarding the design of efficient activity data collection with crowdsourcing.

3.2 Introduction

Ubiquitous crowdsourcing, or the crowdsourcing of jobs in settings beyond the desktop, has drawn attention owing to the increasing maturity of ubiquitous technology and mobile devices, such as public displays and smartphones [155]. This model has become a popular means of acquiring labeled data with any device, in any format, and in any location. Further, it can be applied to a wide variety of tasks where humans are not limited to a stationary use and are more accurate than computers (e.g., labeling activity data).

While crowdsourcing work is efficient, cost-effective, and highly scalable for collecting data, relying solely on the crowd is highly impractical and challenging for most real-world situations. Fundamental approaches to crowdsourcing suffer from two major problems. First, the collected labels lack in accuracy as workers are typically not experts in the field and may not be specifically trained for labeling. For instance, inertial sensor data, which is commonly used in activity recognition applications, is hard to interpret and annotate. Additionally, it is impossible to examine the users' input (i.e., the actual action performed by users). Consequently, the collected labels may be noisy by nature. Second, workers often lack motivation and sustained engagement. Although they exhibit early enthusiasm, workers may lose interest and drop out over time or might not be highly invested in producing high-quality labels. This situation leads to low-quality data collection and biased data (e.g., generating uneven, subjective, and unreliable label). We assume this as inaccurate data and propose a method to detect these situations.

In response to these challenges, we introduce the CrowdAct system, which allows crowdsourcing applications for smartphone sensor systems to achieve highly accurate training datasets in activity recognition based on three features. First, we employ *Active Learning* [135] to solve the lack of accurate labels. Active learning is especially well suited for activity recognition because of the abundance of sensor data. Still, ground truth labels are expensive to obtain. Contrary to the intuition of active learning to select the samples that require labeling, we implement aggregate confidence scores of the predicted label samples and utilize those scores as information feedback. Second, we exploit *Gamification*: the idea of using game design elements in non-game contexts to improve user experience [35]. We use gamification to motivate and sustain worker engagement by packaging the labeling task in the form of a game. For the game point system, we use the confidence scores of the predicted labels gained from the aforementioned active learning method as gamification points. Finally, we propose an *inaccuracy detection algorithm* using supervised classification techniques to reduce inaccurate data in crowdsourcing applications. With these three features, we overcome the low-quality labeling problem of ubiquitous crowdsourcing.

The proposed CrowdAct system does not focus on gathering a large amount of data from the crowd. Contrarily, this study focuses on the accuracy of human contributions in achieving high-quality and consistent ground-truth labeling and, particularly, on the impact of the "gamified active learning system" and feedback under different conditions. An overview of CrowdAct is shown in Figure 3.1. In short, the main contributions of this work to the field are the following:

1. We developed a novel system, called CrowdAct, to achieve high-quality crowdsourced datasets in activity recognition by overcoming three main issues in crowdsourced labeling: lack of accurate information, loss of motivation and engagement, and inaccurate data.
2. We deployed the CrowdAct system to a realistic crowdsourcing platform demon-

strating its capability and feasibility.

3. We recruited 120 diverse workers and gathered 6,549 activity labels with smartphone sensors. We also collected user engagement information, such as survey data and log events, from smartphones. We reviewed, analyzed, and used the obtained data for evaluations.
4. We empirically evaluated the quality of CrowdAct by proposing two conditions (AL and AL+CD) and comparing them with a baseline condition (BL) using techniques such as machine learning and descriptive and inferential statistics. The results indicate that the proposed method can achieve accurate, and consistent labeling in crowdsourced datasets.

We discuss the results, challenges, limitations, as well as implications of this research on the design of efficient activity data collection methods with crowdsourcing.

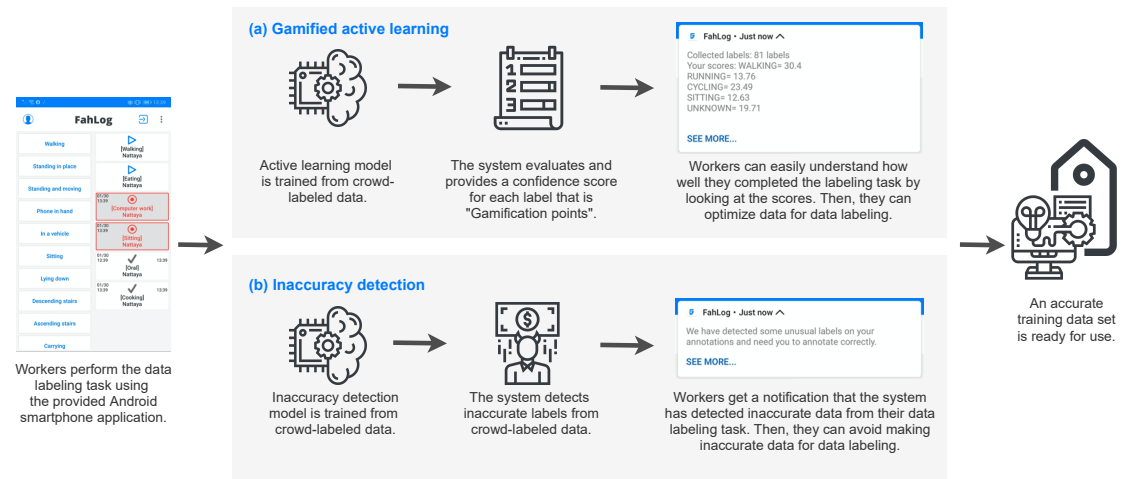


Figure 3.1 High-level overview of the proposed CrowdAct system for robust activity data collection with crowdsourcing. We propose two novel independent ideas directed to the same goal: (a) Gamified active learning to improve recognition accuracy and user engagement; (b) Inaccuracy detection to enhance the quality of crowdsourced datasets.

3.3 Method

We specifically aim to address the challenges and barriers of crowdsourced data collection for activity recognition identified in Chapter 2.2. Therefore, we introduce CrowdAct, novel gamified active learning and inaccuracy detection for crowdsourced data labeling for an activity recognition system. The CrowdAct learning procedure is reflected in Figure 3.1.

In summary, the goal of the CrowdAct system is three-fold:

- **G1:** Employing an active learning algorithm *to address the lack of accuracy* in crowdsourced data labeling.

- **G2**: Applying the gamification concept *to overcome the lack of motivation and sustained engagement* in crowdsourced data labeling.
- **G3**: Building an inaccuracy detection algorithm *to minimize inaccurate data* in crowdsourced data labeling.

To achieve **G1** and **G2**, we designed the Gamified Active Learning algorithm (Section 3.3.1). To accomplish **G3**, we developed the Inaccuracy Detection algorithm (Section 3.3.2). In the following subsections, we describe these algorithms in more detail.

3.3.1 Proposed Gamified Active Learning

In this section, we introduce the idea of integrating active learning and gamification. We applied two standard approaches in the pool-based active learning setting (in which queries are selected from a large pool of unlabeled instances U) using an uncertainty sampling query strategy (which selects the instance in the pool with the highest label uncertainty), as described in [135]. Because our crowdsourced labeling system relies on real-world learning problems, large collections of unlabeled data can be gathered from sensors data simultaneously. This motivates pool-based sampling, while the most straightforward and commonly used query framework is uncertainty sampling. The main difference between the original pool-based active learning pipeline and the proposed pipeline is that the former calculates the uncertainty of the current model in predicting each label, detects the most informative unlabeled samples, and queries the user for a label. Contrarily, the proposed pipeline evaluates the entire collection of unlabeled data from past data before discarding the most uncertain samples and utilizing the remaining predictions as gamification points (i.e., information feedback). The gamification points are then returned as feedback to the user to motivate user contribution and improve the quality of data labeling. The proposed pipeline is typically suitable for mobile activity recognition using online crowdsourcing settings in which a label does not require the timestamps to be precise. In other words, straightforward querying of the user's event at an accurate timestamp for a label is hardly possible, impeding the user from labeling it. Additionally, the prior study (Section 2.4) disregards the most uncertain sample in its prediction by assigning it gamification points. Contrarily, we discard this sample.

In short, we did not create a new active learning algorithm nor did we optimize our code for fast real-time operation. Instead, we applied, adapted, and integrated the idea of active learning with gamification for achieving high-quality crowdsourced datasets in mobile activity recognition. Algorithm 1 illustrates the pseudo-code of the proposed gamified active learning algorithm. The key component of the algorithm concerning the design of a gamified active learning system is found in line 12. The algorithm consists of eight steps:

1. Let X be the input space of instances. Each $x \in X$ is presented in a p -dimensional space as a feature vector $x \stackrel{\text{def}}{=} (x_{f_1}, x_{f_2}, \dots, x_{f_p})$ of sensor data, where $x_{f_i} \in \mathbb{R}$ and

f_i is the i^{th} element in x . Each $y \in Y$ represents an activity class from the set of possible activities C , where $Y \subset C$

2. Let U be the set of unlabeled instances $\{(x^0, x^1, \dots, x^i)\}$
3. L be the set of initial labeled instances $\{(x^0, y_0), (x^1, y_1), \dots, (x^j, y_j)\}$, where $j < i$
4. Use L to train a model θ (i.e., the subject to optimization) by using eXtreme Gradient Boosting (Xgboost) [25].
5. Compute a probability Δ_x^θ of each x in U by using an uncertainty sampling strategy with entropy [136]:

$$\Delta_x^\theta = \left(- \sum_y P_\theta(y|x) \log P(x|y) \right), \quad (3.1)$$

where y_i ranges over all possible labelings of x .

6. Map the predicted label of x , y , with a probability Δ_x^θ .
7. Select x^* , the most uncertain instance according to Equation 3.2. Then, discard x^* .

$$x^* = \underset{x}{\operatorname{argmax}} \Delta_x^\theta \quad (3.2)$$

8. Calculate the entropy of each class c , by averaging Δ_x^θ of all instances predicted as c .

Algorithm 1 Gamified Active Learning Algorithm

INPUT:

U = pool of unlabeled instances $\{(x)^{(u)}\}_{u=1}^U$

L = set of initial labeled instances $\{(x, y)^{(l)}\}_{l=1}^L$

OUTPUT: A = The Δ_x^θ on average of each class $\{(n)^{(c)}\}_{c=1}^C$, where $n \in N$

- 1: **function** GAMIFIEDACTIVELEARNINGALGORITHM(L)
 - 2: $D = \{\}$
 - 3: $\theta = \text{train}(L)$
 - 4: **for** each x in the pool U **do**
 - 5: $\Delta_x^\theta = \text{compute } \Delta_x^\theta \text{ of } x \in U \text{ using model } \theta \text{ and Equation 3.1}$
 - 6: $d = (y, \Delta_x^\theta) : \text{map the predicted label } y \text{ with a probability } \Delta_x^\theta$
 - 7: $D \leftarrow D \cup d$
 - 8: **end for**
 - 9: select x^* , the most uncertain instance according to Equation 3.2.
 - 10: discard x^* from U and D
 - 11: **for** each c in C **do**
 - 12: $A[c] = \text{average scores } D \text{ for class } c \text{ and assign into } A$
 - 13: **end for**
 - 14: return A
 - 15: **end function**
-

3.3.2 Inaccuracy Detection

In this section, we address the problem of inaccuracy detection by using supervised learning. Here, we followed the classical steps of a data science project: data preparation, model training, evaluation, and finally, deployment. To perform supervised learning, we need both "accurate" and "inaccurate" labeled samples to train a model. However, training a supervised machine learning model to detect inaccurate data is very difficult owing to the low number of actual confirmed examples of inaccurate data (imbalance classification problems) [93]. Therefore, we first created a synthetically generated dataset to train our model. One of the challenges of creating inaccurate samples is the extreme variability and flexibility of the inaccurate data patterns, i.e., a situation where labels are uneven or unreliable. Consequently, to ensure diversity among the inaccurate samples in our training data, we created inaccurate samples that contain both an intentional inaccurate dataset and a fake inaccurate dataset by random permutation. One risk of the experiment was that the nature of actual cheating behavior is fundamentally different from randomly ordered and instructed inaccurate data where no purposeful cheating pattern existed. Please note that given this inaccurate dataset, these results cannot be claimed human's cheating is detected, but certainly, carry a significant indication that a machine can efficiently detect inaccuracies by attempting to simulate cheating as best as possible. To build these detection patterns, we implemented four steps:

1. We employed our common crowdsourced dataset (accurate data) collected in the field [98] to create a fake inaccurate dataset by random permutation by (1) randomly changing the label to the signal and (2) taking signals from two different classes and randomly permutating parts of them.
2. We used the reference study [97] as a model for the protocol of inaccurate sample collection. We crowdsourced workers to intentionally create an inaccurate dataset using a similar protocol of [98] for a completely different purpose.
3. We combined the random fake and intentional inaccurate datasets; subsequently, we obtained inaccurate samples containing possibly several inaccurate patterns (intentional or unintentional).
4. We merged our common crowdsourced and inaccurate datasets to a synthetic dataset, which assists in creating a set of synthetic labels and an initial set of features for training.

For the training algorithm, we divided the dataset into training and test sets. The datasets contain different users to evaluate the robustness of the classifier to new users. We used the training dataset to build and validate the model and treated the test dataset as the unseen new data. With the training data created, we handled imbalanced classes with upsampling using the SMOTE algorithm [24]. By oversampling only on the training data,

none of the information in the validation data is used to create synthetic observations. Therefore, these results should be generalizable. The models were trained using Xgboost and evaluated with three repeats of 10-fold cross-validation. Because accuracy is not a helpful metric for this task ^{*1}, we used area under the curve (AUC) of a receiver operating characteristic (ROC) curve [41] as a metric of accuracy. This is ideal for maximizing the true positive rate while minimizing the false positive rate. Consequently, our training algorithm can achieve both a high AUC and recall of 98% because it can capture many true positives but also many false positives. Hence, the model is almost always able to correctly identify inaccurate data. The visual insight of the results is presented in Figure 3.2. Because the model performance is acceptable by our standards, we used it on real-world data. If the detection is classified as "inaccurate", a notification is sent to the user to caution the inaccurate data. In the following, we present the inaccuracy training and detection algorithms.

Algorithm 2 Inaccuracy Training Algorithm

INPUT: $L =$ sensor dataset X of size N , and the same size of labels $C =$

$\{"inaccurate", "accurate"\}^N$

OUTPUT: $f =$ inaccuracy estimation function f

- 1: **function** INACCURACYTRAININGALGORITHM(L)
 - 2: remove samples with no activity labels from X and C
 - 3: calculate feature vectors V from X
 - 4: using supervised machine learning, train a model f with X and C to estimate the inaccurate or accurate
 - 5: return function f
 - 6: **end function**
-

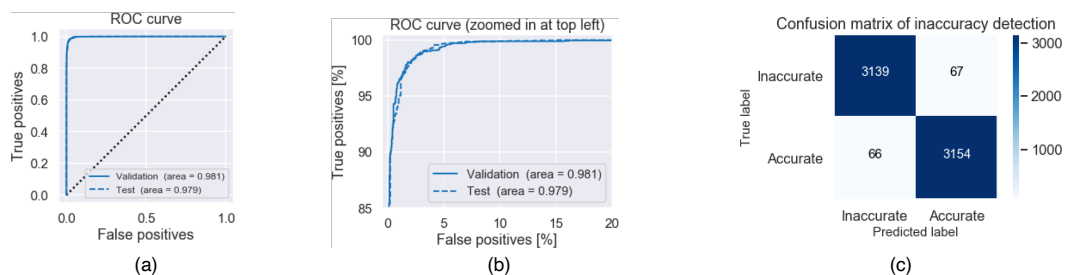


Figure 3.2 (a) ROC curve and AUC value for the inaccuracy detection algorithm; (b) Zoom-in view of the upper left corner of panel (a); (c) Confusion matrix for the inaccuracy detection algorithm without normalization.

^{*1} A 99.8%+ accuracy can be achieved on this task by predicting False all the time

Algorithm 3 Inaccuracy Detection Algorithm**INPUT:** X = sensor dataset X , f = function f from Algorithm 2**OUTPUT:** O = "inaccurate" or "accurate"

- 1: **function** INACCURACYDETECTIONALGORITHM(X, f)
- 2: remove samples with no activity labels from X
- 3: calculate feature vectors V from X
- 4: using f , estimate inaccurate or not by $f(v_i)$ for $\forall v_i \in V$ and assign to a predicted label set \hat{Y}
- 5: count number of occurrences of an "inaccurate" label in the given the set \hat{Y}
- 6: $pct = N_{inaccurate} / \text{length of the set } \hat{Y}$
- 7: $\delta = 0.5$ # the final decision is made using a threshold value δ
- 8: return output the result by

$$O = \begin{cases} \text{"inaccurate"}, & \text{if } pct > \delta \\ \text{"accurate"}, & \text{if } pct \leq \delta \end{cases}$$

9: **end function**

3.4 End-to-End System

In this section, we describe the system implementation and study design in Figure 3.3 to evaluate the differences between the three conditions in Table 3.1. Motivated and informed by preliminary studies [98,103] and the related works presented in Chapter 2, we designed a large-scale activity data gathering system. This enables us to collect activity labels and sensor data from the crowd and to evaluate our CrowdAct in a realistic crowdsourcing platform. Essentially, in designing the system, we considered three major components:

1. a Task Creation interface: enables us to recruit massive numbers of workers on a crowdsourcing platform to perform our labeling task.
2. a smartphone Task App: enables us to efficiently collect labels and sensor data from the smartphones of individuals and allow each individual worker to receive information from us.
3. a crowdsourcing Task Manager: enables us to store and manage massive crowd-sourced data, evaluate the data by implementing Algorithm 1 and Algorithm 3 presented in Section 3.3, return the information to each worker, and review the results programmatically.

In the following subsections, we detail the design rationale of each component.

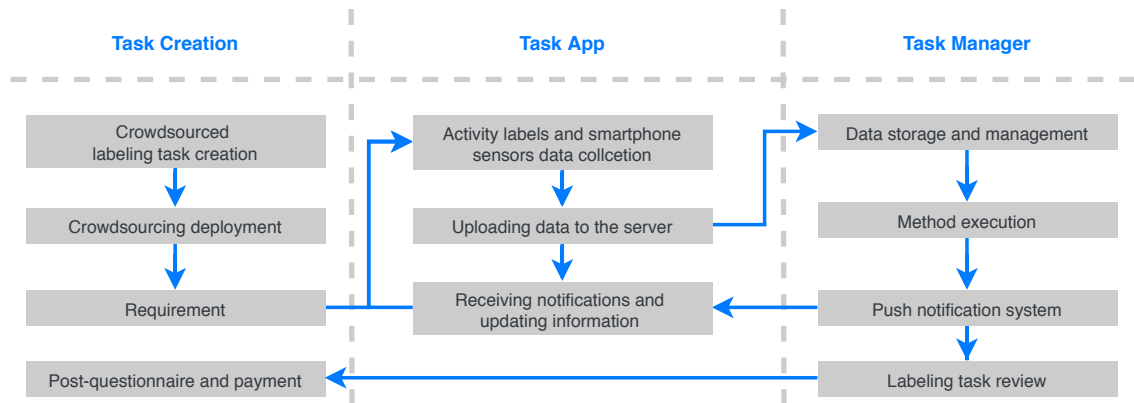


Figure 3.3 End-To-End system design.

Table 3.1 Experimental design summary.

Baseline condition (BL)	Proposed CrowdAct conditions	
Providing random text messages without any algorithms (e.g., "Are you walking?", "Are you eating?", "What are you doing?").	Gamified Active Learning (AL)	Gamified Active Learning and Inaccuracy Detection (AL+CD)
	Providing gamification points using active learning (Section 3.3.1).	Providing gamification points using active learning (Section 3.3.1), and inaccurate-detected messages (Section 3.3.2).

3.4.1 A Task Creation Interface

To allow us to recruit a large crowd of human workers on a crowdsourcing platform, we created a web-based interface for task creation and submitted it to Amazon Mechanical Turk (MTurk) [119] for workers to perform (a so-called Human Intelligence Task, or HIT^{*2}). We aimed to solve barriers founded in the instruction state [98] because an accurate response depends on careful reading of the instructions; e.g., workers might have been unable to follow the instructions that we provided due to a low technical literacy. To alleviate these issues, we described every step and included screenshots. While the initial page provides a brief instruction (Figure 3.4-1) of our labeling task, the second page offers more detailed instructions, including the labeling interface of the smartphone application (Figure 3.4-2). To evaluate user experience, we created a post-study questionnaire, focusing on demographic information and worker engagement (Figure 3.4-3), as detailed in the next subsection. We also asked workers to sign a consent form agreeing to participate in

^{*2} <https://www.mturk.com/worker/help>

the study, enabling us to collect their data. To be sure that the post-study questionnaire was filled in after completion of the labeling task, the worker needed to provide a Task Token acquired from the smartphone application after the task had been completed. The entire Task Creation component is written in HTML5 and JavaScript as the front-end, as well as Ruby and Python as the back-end.

A Post-Study Questionnaire

The post-study questionnaire (Figure 3.4-3) is organized in two tasks: a demographic survey and the user engagement scale short form (UES-SF) [115]. All of the questions are required input fields. The demographic survey contained 5 general questions; the outcome variables included the gender, age, digital literacy, smartphone usage, and smartphone gaming. The UES-SF is the most tested questionnaire that measures user engagement and has been validated in a variety of HCI contexts thus far [124, 175]. It consists of four subscales (focused attention, perceived usability, aesthetic appeal, and reward factor) with 12 items in total, containing a tool that is widely used for measuring user engagement in various digital domains. Each item is presented as a statement using a 5-point Likert scale from "1: strongly disagree" to "5: strongly agree". The higher the UES scores, the greater the engagement was. The UES-SF perfectly fits our context of labeling crowdsourcing tasks with a total of only 12 items; it is easy to drive workers to respond and empowers us to measure their engagement.

3.4.2 A Smartphone Task App

To collect data from the crowd, we relied on the FahLog^{*3} smartphone application, written in Java, which is inspired by the work in [101]. This app can simultaneously acquire both the labels and sensor data, generating streams that are temporarily stored on the smartphone and periodically synchronized to the server on the cloud when the smartphone is connected via WiFi or mobile data. There are numerous features to support labeling for activity recognition with smartphones, such as acquisition of smartphone sensor data, download of activity classes and users, activity label input function, upload of data to the cloud server, offline functionality, and collection of logs events in the application with Firebase Log Events^{*4}. The application can run on any recent Android device (5.0+).

User Interface

The FahLog application consists of three major screens, as shown in Figure 3.5.

- **Sign-in screen:** The welcome screen of the FahLog application prompts workers to sign in for user authentication (Figure 3.5-1) to verify the identity of the worker and the data collection process. Users must enter the username and password given

^{*3} <https://play.google.com/store/apps/details?id=jp.sozolab.fahlog&hl=en>

^{*4} <https://firebase.google.com/docs/analytics/events>

Activity Data Collection with Android Smartphone 1


[Summary](#) | [Detailed Instructions](#) | [Survey](#)

Summary
Labeling activity data on Android smartphone, then getting feedback and improving your labeling.
 A summary of the assignment is shown below:

To avoid being rejected, please read much more step-by-step instructions inside the "Detailed Instructions" tab.


1. **Installing** – Download the app from Google Play and install it on your smartphone.
2. **Labeling** – Do activity from the pre-defined list and annotate it.
3. **Uploading** – Connect your smartphone to a WiFi network, the data will be then uploaded automatically, but please carefully check the information on the app so that you can confirm your data was successfully uploaded.
4. **Improving** – Get suggestions on your labeling through mobile notifications and improve it by relabeling.
5. **Doing a survey** – Do a post-survey inside the "Survey" tab. **When you complete the labeling task on the smartphone app, please let us know what you thought of this assignment and our systems by doing a post-survey.**
6. **Meet the requirements**
 - o Collect 50 labels or more
 - o Perform and annotate the activities for 3 hours or more
 - o Do a post-survey after the labeling task is complete!
7. **Finishing and getting money!** – Please wait for a while to review. If you meet all the requirements, the assignment will be accepted; otherwise, it will be rejected.

4. Download FahLog app from Google Play Store
 Click this [link](#), or open the Google Play Store on your Android smartphone and find "FahLog" app to download. Tap the install button.



2

5. Sign in to the app with your account
 Open the app on your phone and click the "Signin" button. Set the same e-mail address and password that you received on the HIT, and click the "Signin" button again to sign in. Waiting for a moment... If you successfully sign in, the screen will transit to the Main screen and show the list of predefined activities on the left column.



3

Focused attention
 I lost myself in this experience
 The time I spent using Fahlog application just slipped away.
 I was absorbed in this experience.

Perceived usability
 I felt frustrated while using this Fahlog application.
 I found this Fahlog application confusing to use.
 Using this Fahlog application was taxing.


Aesthetic appeal
 This Fahlog application was attractive.
 This Fahlog application was aesthetically appealing.
 This Fahlog application appealed to my senses.

1

3

4

1) Before signed in, no information here.
2) Click the "Sign in" button.
3) Set the same email and password when you created your account and click the "Sign in" button.
4) After signed in, the activity list is shown.



3






Choose...
 1 - Strongly agree
 2 - Agree
 3 - Somewhat agree
 4 - Neither agree or disagree
 5 - Somewhat disagree
 6 - Disagree
 7 - Strongly disagree

Choose...
 Choose...
 Choose...
 Choose...
 Choose...
 Choose...

Figure 3.4 Example of the labeling task in crowdsourcing: (1) brief instructions, (2) detailed instructions, (3) post-study questionnaire.

in the HIT to be able to perform the labeling task. A username and password are designed to be unique for each user and HIT.

- **Data labeling screen:** To recognize activities using supervised learning, we need to collect supervised information (i.e., activity labels) of activities along with sensor data. Activities are temporal information with specific duration; hence, it is important to record both the start and end times. For this reason, we provided the data labeling screen that has this function. After the worker have successfully signed in, the screen displays a labeling screen (Figure 3.5-2), which enables the user to perform the labeling task. (Figure 3.5-2a) presents the list of predefined activities (Figure 3.10), which can be used for the activity recognition classes. After selecting the activity label from the left column, the activity label is created as a box in the right column. Each time the worker taps an activity label box, it transitions to before start (▶) (Figure 3.5-2b) → doing activity (⊙) (Figure 3.5-2c) → finish (✓) (Figure 3.5-2d), so one can record the start and end of the activity. The left and right columns provide the functionality of the scroll view to see more content.
- **Dialog interface screen:** To allow workers to efficiently perform the data labeling task, we provided the dialog interface screen that includes the required information, namely, the starting time, working time, number of activity labels, and list of label-

ing scores. The dialog interface presents the necessary information (Figure 3.5-3a) and gamification points (Figure 3.5-3b) of each user and is displayed when the user presses the () button (Figure 3.5-2e). The () symbol represents the starting time, i.e., the time that the user starts using the application. The () symbol represents the working time, i.e., the time that the user spends using the application. The () symbol represents the number of activity labels that the user has annotated and successfully uploaded to the server. The () symbol represents the list of labeling scores for each activity class that was measured and suggested to improve, i.e., gamification points.

To evaluate the differences between the three conditions shown in Table 3.1, we created interfaces for each condition consisting of a dialog interface and a notification interface. These were designed to provide necessary user information, including the starting and working time, the number of uploaded labels, and gamification points calculated by Algorithm 1. We designed variations of the same dialog and notification interfaces for each condition: Baseline, Gamified Active Learning, and Gamified Active Learning and Inaccuracy Detection.

The three conditions were designed as follows:

- **Gamified Active Learning (AL):** This condition consists of a notification interface with gamification points calculated by Algorithm 1 (Figure 3.6-1) and a dialog interface showing the user's information and list of gamification points (Figure 3.5-3), which enable users to get feedback on their labeling tasks.
- **Gamified Active Learning and Inaccuracy Detection (AL+CD):** This condition closely resembles the AL interface, with a notification and a dialog interface presented to the user. However, there is a significant difference from the AL condition; a notification with inaccurate-detected messages provided by Algorithm 3 is implemented (Figure 3.6-2).
- **Baseline (BL):** This condition consists of a notification interface with random text messages (Figure 3.6-3) and a dialog interface showing only the user's information (Figure 3.5-3a). No gamification points information is provided in this condition as (Figure 3.5-3b) is removed from the interface.

The user can press the button to open the dialog interface or look at a notification to review their data. Both gamified (Figure 3.6-1) and baseline (Figure 3.6-3) notifications were designed to alert every 15 minutes after the user first signs in on the smartphone. Contrarily, an inaccuracy notification (Figure 3.6-2) was designed to alert every time the inaccuracy detection system is activated, as defined by Algorithm 3. The dialog interface (Figure 3.5-3) for all conditions was designed to display every time the user presses a button. In addition, to prevent multiple clicks on a dialog notification. When users click the "SEE MORE..." button inside a dialog notification (e.g., Figure 3.5-1), the CTR is

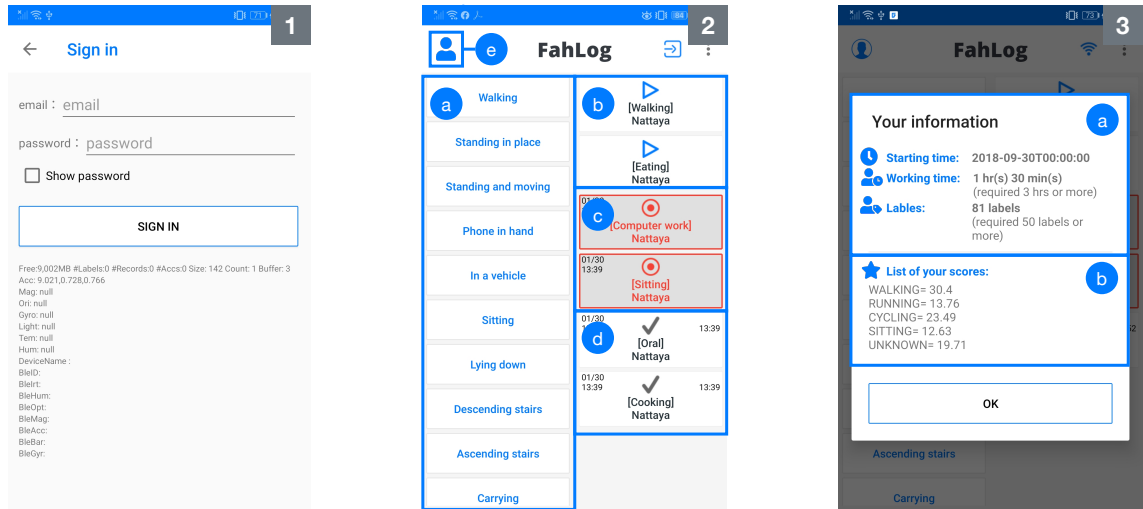


Figure 3.5 The Fahlog Android application for data collection: (1) Sign-in screen, (2) data labeling screen, (3) dialog interface screen.

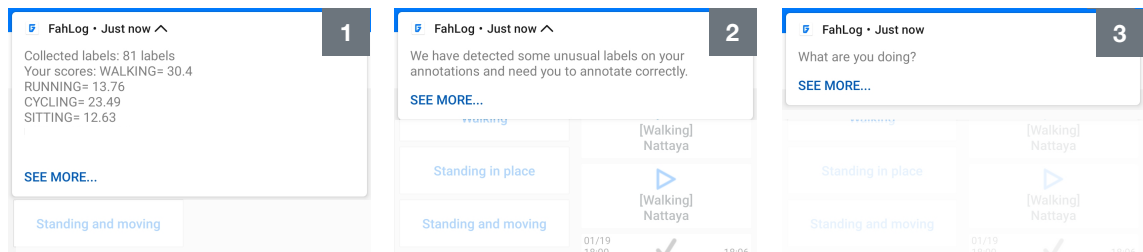


Figure 3.6 Notifications on the smartphone app: (1) Gamified notification displaying the gamification points, (2) inaccuracy notification displaying inaccurate-detected messages, (3) baseline notification displaying random text messages.

recorded, the dialog disappears, and the MainActivity of FahLog application launches instead.

Smartphone Sensors

We collected 3-axis acceleration sensor data available in the application. We set a 20-Hz sampling rate for the smartphone application, which is the standard and lowest setting. Since we used 1-minute time windows, the sampling rate was sufficient for the data collection. Since the crowd workers were using their smartphones, we could not drain their battery. This configuration allowed us to optimize the sensing process to balance the amount of data generated and battery consumption, even if this meant having less frequent sensor readings.

3.4.3 A Crowdsourcing Task Manager

The Task Manager is a cloud server^{*5}, which is inspired by the work in [67,98]. It enables us to obtain and maintain large-scale data from the crowd. The server receives sensor and label files in the JSON format, which are sent between smartphone applications for each user by the POST method in the HTTPs protocol. By using past activity labels and sensor data stored as training data, the machine learning method presented in Section 3.3 is performed once every 15 minutes. After evaluation, the system updates and notifies users of user information, gamification points, inaccurate-detected messages, or random text messages, according to their condition assigned through smartphone notifications. Besides, there is a labeling task review function, which enables us to review submitted tasks and decide whether to approve or reject the HIT of each user. If the task has been successfully finished, a Task Token is given to the user through smartphone notifications, enabling the users to complete the post-study questionnaire described in Section 3.4.1. The server management was written using the Ruby on Rails framework^{*6}. The data is managed by a relational database with Amazon relational database service (RDS) for MySQL^{*7}. The machine learning methods were written in R and Python. The notification system was implemented by Firebase Cloud Messaging^{*8}. All systems and data were run on the Amazon web services (AWS) elastic beanstalk^{*9} environment equipped with the Amazon simple storage service^{*10} and an elastic load balancer^{*11}.

3.5 Preliminary Evaluation

This section builds an initial investigation phase for crowdsourcing experiments. We deployed a similar gamified active learning system to a lab environment and asked students to perform the preliminary experiments proving the system's capability and feasibility. There are a few differences in the experimental design and evaluation method, which are described below. The preliminary evaluation provided us technical information that can highlight important areas for consideration and provide valuable insight into requirements and standards before real deployments.

^{*5} <https://fahact.sozolab.jp/>

^{*6} <https://rubyonrails.org/>

^{*7} <https://aws.amazon.com/rds/mysql>

^{*8} <https://firebase.google.com/docs/cloud-messaging>

^{*9} <https://aws.amazon.com/elasticbeanstalk>

^{*10} <https://aws.amazon.com/s3/>

^{*11} <https://aws.amazon.com/elasticloadbalancing>

3.5.1 Experimental setup

An overview of our experimental design is explained in detail in Table 3.2. A total of 11 students who have mobile activity data collection experience participated in two-day experiments. The participants were required to carry Android phones in their pants pockets, install the app on the phones that we extend from [101] by including notifications, to select and record their daily life activities from the list of predefined labels (depicted in Figure 3.5), get notifications of gamification points, and submit data to our server. Each participant performs the experiments for 6 days (12 hours from 8 AM to 8 PM). We do and repeat our processes as we described in the method section every 50 minutes.

To compare our proposed method with others, we created notifications on smartphones that displays 3 different versions. Each notifications version only differs in the user interface and algorithm for calculating gamification points. Each participant will receive all three conditions of the notifications, each of which will show 2 days. We randomly display the conditions for participants to ensure that they are not affected by the day of experiments for each term. We also request the users click the push notifications sent to assure that the users have seen the notifications. Then, we collected log events when the user clicks on notifications; these log events are such valuable to get insight into message delivery and user engagement.

Table 3.2 Experimental design

Method	Conditional detail
Proposed	Receive proposed gamified active learning (Algorithm 1)
Traditional	Receive traditional gamification points (Equation 3.3)
Without	Receive messages "What are you doing?".

Equation 3.3 presents traditional gamification points using the accuracy. Accuracy [41] is one traditional metric for evaluating classification models, it can measure the performance of activity data collection. Consequently, we used the accuracy as gamification points for a baseline method to compare with our proposed method. To use the accuracy as gamification points, we trained a machine-learning algorithm with each user's collected smartphone sensor data and activity labels. We then evaluated the accuracy of a classifier for gamification points. However, such a strategy is often not feasible in reality due to a problem of accuracy paradox [150] (i.e., when a model may have a high level of accuracy but be too crude to be useful). For example, if the incidence of 'Walking' is dominant, being found in 99% of cases, then predicting that every case is 'Walking' will have an accuracy of 99%. Thereby, we computed gamification points reasonably for users to avoid the accuracy paradox problem: where a dataset is unbalanced, the overall accuracy is not representative of the actual performance of a classifier. The following formula is used to

calculate the accuracy as traditional gamification points:

$$\text{gamification points} = \text{accuracy}^1 / c \quad (3.3)$$

by bending the curve using accuracy to the power of one over classes, the traditional gamification points will be weighted by the inverse of classes, where c is the number of activity classes by the user.

3.5.2 Activity Recognition

Data description

The dataset was collected between May 2019, from 11 subjects within an age bracket of 21-26 years, performing one of 12 regular activities (as shown in the left column of Table 3.3) while carrying an Android smartphone (Wiko Tommy3 Plus) in the pants pockets that recorded the movement data (accelerometers in smartphones). The total number of labels is 1,236.

Table3.3 Number of activities for each activity class

Activity class	# labels	Activity class	# labels
Walking	410	Running	3
Sitting	370	Standing	213
Downstairs	61	Upstairs	50
Lying	40	In vehicle	32
Cycling	25	On train	15
Phone	6	Carrying	11
Total = 1,236 labels			

Data preprocessing

We put together the dataset by including 3-axis accelerometer sensor data and the activity labels on the smartphones without clock and time synchronization because the sensor and the labeling system are both in the same device. We used sliding windows of one minute with no overlapping. For each axis, average, standard deviation, maximum value and minimum value were extracted as features. Before data proceeding, we excluded missing values. As a result, we obtained multivariate data of 21,132 samples with 12 variables for feature vectors. Figure 3.8 shows the activity labels distribution of the data samples in our dataset. It is worth noting that the distribution is highly skewed, where some classes appear more frequently than others. Since imbalanced dataset can negatively influence the generalization and reliability of supervised learning algorithms [77], we employed the SMOTE algorithm: Synthetic Minority Over-sampling Technique as presented in [24] (an oversampling technique that creates new synthetic data samples in the minority classes, varying the features values of the existing data points based on their k nearest neighbors

in the feature space) in order to balance our dataset.

Evaluation method

To evaluate the proposed method, we use a technique of supervised machine learning algorithm for multiclass classification. We trained each participant separately (one model for each participant) using Random Forest classifier: an ensemble learning method for classification by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees. To test the model’s ability we used stratified k-fold cross-validation. The folds are made by preserving the percentage of samples for each class to ensure each fold is a good representative of the whole. To account for label imbalance, the model performance was presented using the weighted average of precision, recall, F1-score of each class for the multiclass task. (i.e., averaging the support-weighted mean per label) So the average is weighted by the support, which is the number of samples with a given label.

3.5.3 User engagement

In this section, we propose Click-through rate (CTR) [89] to assess users’ depth of engagement with each notifications version displayed. To measure CTR, we use the click logs collected; the CTR formula is defined as follows:

$$CTR = \left(\frac{\text{Total measured clicks}}{\text{Total measured notification impressions}} \right) \times 100 \quad (3.4)$$

where ‘Total measured clicks’ are the total amount of clicks on notifications and ‘Total measured notification impressions’ are number of times notification was sent on smart-phones (which were counted by Google Analytics for Firebase^{*12}).

3.5.4 Results

Following the evaluation approach discussed above, we report our results of the validation together with a discussion of such results. The classification performance results are shown in Figure 3.7. The collected activity label results are shown in Figure 3.8 and Table 3.4. And the user engagement results are presented in Figure 3.9.

Quality of collected activity data

As we can see from Figure 3.7, all of the classification performance metrics improve with our proposed method. F1-score was improved from 0.71 to 0.83 (+0.12) compared to traditional method. F1-score was improved from 0.70 to 0.83 (+0.13) compared to without method. Recall was improved from 0.71 to 0.84 (+0.13) compared to traditional method.

^{*12} <https://firebase.google.com/docs/analytics>

Recall was improved from 0.71 to 0.84 (+0.13) compared to without method. Precision was improved from 0.72 to 0.83 (+0.11) compared to traditional method. Precision was improved from 0.70 to 0.83 (+0.13) compared to without method. Although some participants did not improve the classification performance in the proposed method, the overall proposed method generally significantly outperform than the other methods.



Figure3.7 The performance results for each method

Quantity of collected activity data

As we can see from Figure 3.8, the number of collected activity labels was increased with our proposed method. The number of activity labels was increased from 409 to 498 (+89) compared to traditional method. The number of activity labels was increased from 329 to 498 (+169) compared to without method.

User engagement

As we can see from Figure 3.9, the percentage of CTR was increased with our proposed method. The percentage of CTR was increased from 78.3% to 80.4% (+2.1%) compared to traditional method. The number of activity labels was increased from 14.3% to 80.4% (+66.1%) compared to without method.

3.6 Crowdsourcing Deployments

To verify the proper function of the protocol and data collection process and to assess the effect of the proposed CrowdAct on labeling, we performed a verification experiment on MTurk with 120 workers across the three conditions of the system: AL, AL+CD, and BL, as described in Table 3.1. Following prior work [98], we smoothly proceeded with the HIT since the procedure of labeling tasks for activity recognition with smartphones in

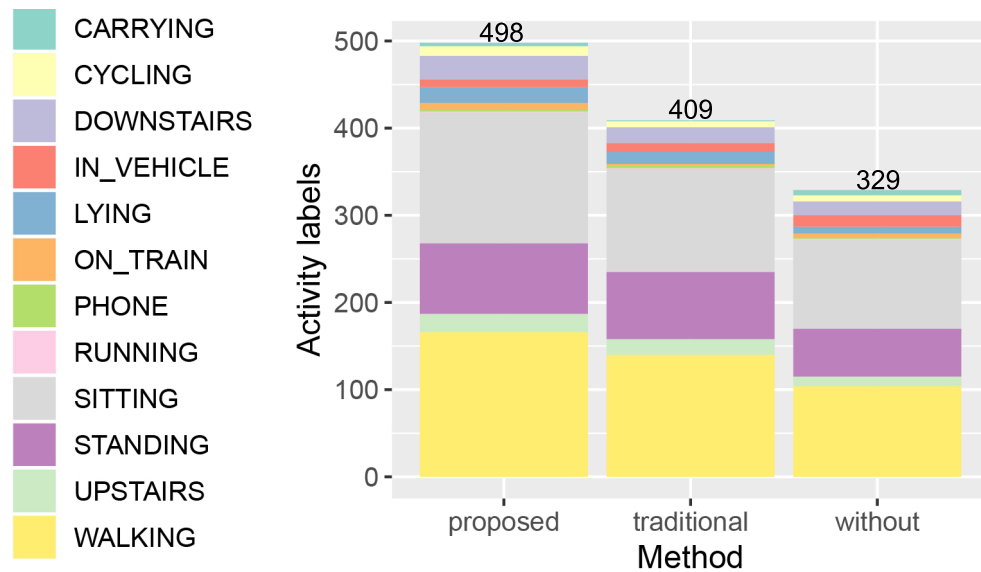


Figure3.8 The number of activity labels for each method

Table3.4 The number of activity labels of each activity class for each method

Activity class	Proposed	Traditional	Without
Walking	166	140	104
Sitting	150	118	102
Standing	81	77	55
Downstairs	27	18	16
Upstairs	21	18	11
Lying	18	14	8
Cycling	11	7	7
In vehicle	9	10	13
On train	8	2	5
Phone	2	3	1
Carrying	4	1	6
Running	1	1	1
Total	498	409	329

crowdsourcing presented several challenges. In this work, we followed the prior pipeline and modified some requirements, as detailed in the following section.

3.6.1 Procedure

The objective of our HIT directed workers to perform an activity labeling task using the provided Android smartphone application and subsequently complete the post-study questionnaire. The data was collected using different smartphone models from different

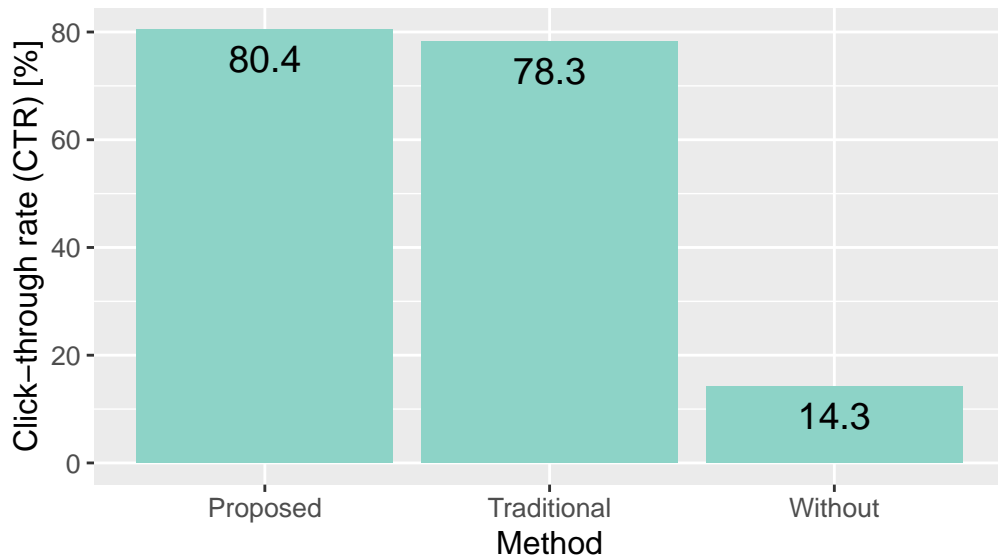


Figure3.9 CTR for each method

vendors to support the generality of our approach and its usability in real-world conditions. We posted 120 HITs (one HIT per worker) on MTurk for workers globally. The HIT was a between-subject design with 3x40 participants who were randomly assigned to one of three experimental conditions: BL, AL, and AL+CD (Table 3.1). The design choices and related user interface are detailed in Section 3.4.2 *A smartphone Task App*. HITs were deployed in small batches to avoid latency issues resulting from concurrent usage. Participants were asked to engage only in one HIT; all honored this request. We also ensured that each participant submitted at most one HIT across all experimental conditions by tracking worker-ids to avoid learning biases owing to repeated participation.

Participants were asked to carry Android phones with at least 5.0, or more API levels in their pants pocket and perform activities from the classes predefined in Figure 3.10. They had to assign 50 labels and spend 3 hours on the smartphone application at least. We did not specify the length of the recording; however, we suggested they record at least 2 to 3 minutes per activity. Participants required roughly 4 hours on average and were given up to 1 day. However, the HIT typically takes less than 10 minutes for reading the instructions, 5 minutes for the application preparation, 3 hours or more for the data labeling, and approximately 10 minutes for the questionnaire. Thus, participants did not need to rush. Participants were paid \$5 per HIT, resulting in an average hourly rate of \$1.25. The standard rate depended on several factors, such as the difficulty and complexity of tasks. Low-paying jobs remained attractive to workers as they could smoothly perform the labeling task using the provided Android smartphone application without decreasing their efficiency. However, a low-pay rate marks the trade-off between low-paid and speed recruitment (e.g., high-paying tasks can accelerate the recruitment process). While the workers earned very low hourly wages in this study, future research should attempt to

set minimum wages based on common requirements for higher quality crowdsourced research [58]^{*13}. All data were collected in accordance with Mturk’s acceptable use policies for research with human subjects and data retention^{*14}.

3.6.2 Participants

The study was conducted in January and February 2020 on MTurk. 120 people (52 female, 68 male) between the ages of 22 and 57 years old (37.64 ± 9.37) participated in the study. Of the participants 11 were extremely digitally literate, 44 were moderately digitally literate, 42 were quite digitally literate, 12 were slightly digitally literate, and 11 were digitally illiterate. Regarding smartphone usage, 50 participants spend more than 6 hours on their smartphones per day, 40 participants spend 4 to 5 hours, 24 participants spend 2 to 3 hours, and 6 participants spend 1 to 2 hours. Regarding smartphone gaming, it was reported that 49 participants regularly played games on their smartphones, 64 participants sometimes played games on their smartphones, and 7 participants never played games on their smartphones. The demographic data of workers who participated in this study are summarized in Table 3.5 and were used for data analysis.

Table 3.5 Demographic data for the user study (see Section 3.6.2). The 120 participants are categorized according to the experimental design conditions.

Outcome Variable	AL	AL+CD	BL
Age (m \pm std)	36.35 \pm 9.27	40.03 \pm 9.77	35.40 \pm 9.28
Gender (female, male)	20,20	13,27	15,25
Digital Literacy (extreme, moderate, quite, slight, never)	3,9,9,16,3	2,7,13,17,1	8,11,14,5,2
Smartphone usage (>6 hrs, 4-5 hrs, 2-3 hrs, 1-2 hrs)	19,12,8,1	13,15,9,3	23,9,5,3
Smartphone gaming (regularly, sometimes, never)	22,16,2	17,22,1	12,26,2
Working hours (m \pm std)	4.45 \pm 0.68	4.81 \pm 0.87	4.69 \pm 0.73

3.7 Evaluation and Results

In this section, we review the data obtained and evaluate the proposed CrowdAct system in depth to verify whether it can achieve the goals we set out in Section 3.3. Hence, the following evaluation procedures were adopted:

- **Evaluation 1** to assess G1. To address the lack of accuracy, we applied machine learning algorithms using the labels and sensor data collected for activity recognition and compared the recognition accuracy results using the AUC value and F-measure for all conditions described in Section 3.7.2.
- **Evaluation 2** to assess G2. To overcome the lack of motivation and sustained engagement, we measured the user engagement by using two popular approaches;

^{*13} <https://cacm.acm.org/magazines/2018/3/225476-responsible-research-with-crowds/fulltext>

^{*14} <https://www.mturk.com/acceptable-use-policy>

1) the short form of the UES, and 2) the user events (e.g., button press on the smartphone, the click-through rates of push notifications). We then compared the results for all conditions described in Section 3.7.3.

- **Evaluation 3** to assess G3. To minimize the inaccurate data, we compared the collected data of inaccurate-detected data for all conditions described in Section 3.7.4

3.7.1 Data Overview

As a result of the crowdsourced labeling experiments described in Section 3.6, we gathered data from **120 workers** for three conditions. In total, we collected **6,549 activity labels** from **19 activity classes**. The Shapiro–Wilk test [137] was significant ($p < 0.05$), so the assumption of normality could not be met. As such, we employed non-parametric tests, using the Kruskal–Wallis test [78] in all analyses. Where there was a significant main effect, we performed post-hoc tests to compute multiple pairwise comparisons using Dunn’s test with Bonferroni correction [39] to account for multiple testing and identify which groups differed. In this section, we assess the data obtained and present our results.

Participants

Owing to the between-subjects experimental design, each participant received only one condition, which potentially directly affected the final results. We observed whether there was a significant difference between the participants in the three experimental conditions. We included variables such as age, gender, digital literacy, smartphone usage, and smartphone gaming because these variables may impact the data labeling performance of workers. For example, we assumed that workers with a high digital-literacy level could perform the data labeling tasks using the provided Android smartphone more efficiently than workers with a low digital-literacy level. We compared participants’ demographics across conditions to find statistical differences in Table 3.1.

A Kruskal–Wallis test revealed that there was a statistically significant difference in digital literacy score between the different experimental conditions ($H(2) = 9.584$, $p = 0.008$) with a mean rank digital literacy score of 2.83, 2.80, and 3.45 for the AL, AL+CD, and BL conditions, respectively (larger number was best). No statistically significant difference was observed in the remaining dependent variables. The Dunn’s test with Bonferroni correction on the different digital literacy scores between conditions showed that both of the BL versus AL and BL versus AL+CD conditions were statistically significant (Dunn’s test, $p = 0.026$ and $p = 0.019$, respectively).

Activity Classes

Figure 3.10 shows the number of activity labels submitted for each of the 19 classes and conditions in our study. In total, the participants in the AL condition submitted 2,191 labels, in the AL+CD submitted 2,187 labels, and in the BL submitted 2,171 labels. We

notice that some classes have more labels than others depending on the complexity of the activity recording; the activity records are also dependent on the different participant input.

We found that participants in the AL and AL+CD conditions submitted more labels on average (+18 labels across all classes) compared to the number of submitted labels in the BL condition. Specifically, they submitted more labels in 15 out of 19 classes, which corresponds to 79% of the classes (a minimum of +3 and maximum of +70 labels and an average of +17.3 labels per class). Nevertheless, we could collect one more activity class from our two proposed CrowdAct conditions since participants in the AL and AL+CD conditions recorded cleaning activities. Contrarily, participants in the BL conditions did not record the activities.

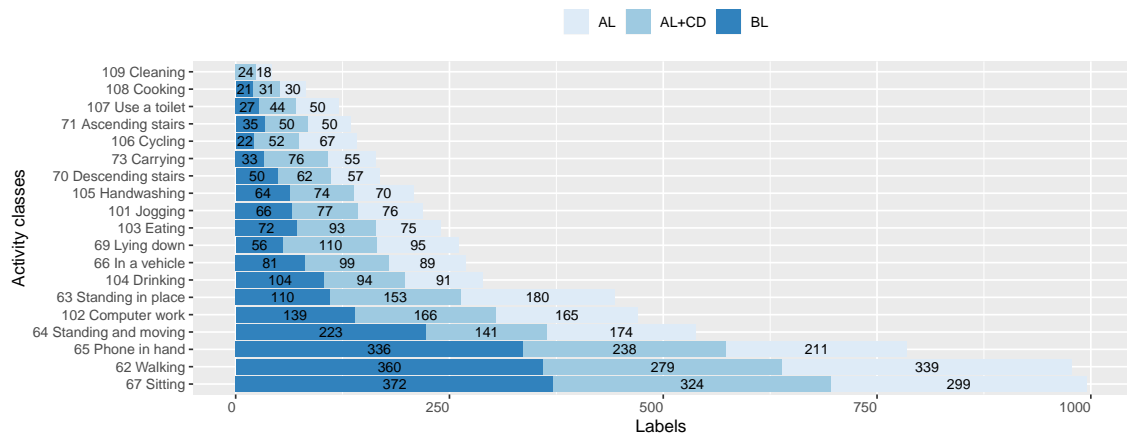


Figure 3.10 Number of labels per activity class recorded by the participants under each condition.

3.7.2 Activity Recognition

We followed a standard activity recognition chain using a supervised learning approach — data preprocessing, segmentation, feature extraction, training, and testing. We then evaluated the accuracy of a classifier trained on crowdsourced data by (1) assessing the feasibility of applying the classifier in each experimental condition using the crowd gathered data and (2) comparing it to a classifier trained on an existing curated dataset. Additionally, we demonstrated the impact of varying crowd size on performance.

Preprocessing

We collected three-dimensional periodic data that include acceleration sensors on the smartphone recording data every 1/20 seconds. The norm of the axes for each row falling in the time slot was computed so as to aggregate the data. Hence, discrepancies deriving from different smartphone positions/orientations at the time of the reading were diminished. We then combined the periodic sensor data and activity labels without clock and

time synchronization since both are located on the same device.

Because standard classification algorithms require features with some specific characteristic to work properly, we first transformed the raw time-series data into examples. We divided the data into 1-minute segments and then generated features. For each axis, the average and standard deviation, and maximum and minimum value were extracted as features. We also included the participants' IDs for user-dependent training, as described in the next section. In total, we used 13 features per sample.

Evaluation Method

For the evaluation, we used the one-versus-the-rest (OvR) multiclass strategy. For each classifier, the class was fitted against all other classes, so each class was represented by a single classifier. Therefore, it is possible to gain knowledge about the class by inspecting its corresponding classifier. With the data prepared, we created a training and a test dataset. We used the training dataset to build and validate the model and treated the test dataset as the unseen new data, as if the model was in production.

Rather than applying the model to new users by comparing it with other labels from the crowd, we focused on the accuracy of human contributions in each condition (e.g., personal context and activities to be used by the user him/herself) by comparing it with the machine's knowledge. Therefore, we performed the following different evaluations: (1) we applied user-dependent training to show accuracy improvements for each participant in each condition without considering side effects such as different sensor positions; (2) we applied user-independent training to show overall accuracy improvements in each condition that can be applied to new users. The models were trained using Xgboost. The real data are highly imbalanced, as shown in Figure 3.10. A popular metrics of precision, such as the F-measure, can be affected by class imbalances. To address this issue, we used the output of the model in a probability form and calculated the ROC curve. We then utilized the AUC for the multiclass problem using the OvR scheme as a metric of accuracy because it is not affected by class imbalances. Further, we used the F-measure after resampling to avoid the negative effects of class imbalances, which reside in natural datasets. We sampled 10,000 feature samples per activity (including other activity classes and features with no labels) to obtain a balanced dataset of each activity class. We then utilized the F-measure after applying thresholds to focus on true positive samples. Three rounds of the 10-fold cross-validation process were carried out at each iteration of the boosting process, and the hyper-parameters of the thresholds of ROC curves were determined. While there are several tuning parameters associated with a model, we performed the grid search tuning the parameters with the highest impact on the model outcome owing to the heavy-duty grid searched. We included the maximum depth of a tree, the minimum sum of the instance weight needed in a child, the minimum loss reduction required, the subsample ratio of the training instances, and the subsample ratio of columns when constructing each tree. We also applied regularization to reduce overfitting. Then, we tested the achieved

accuracy with the prepared test dataset.

Results

Figure 3.11 shows the activity recognition accuracy by AUC and F-measure of user-dependent training and user-independent training for the validation and test data, respectively. Overall, the data indicate that the recognition accuracy of the AL and AL+CD conditions were higher than that of the BL. Figure 3.12 summarizes the performance of each classifier on a set of test data using a confusion matrix with normalization by class to support the size of user-dependent and -independent training for each condition. The fact that the BL matrix was quite sparse explains the low accuracy score of the overall results, while the AL and AL+CD matrices demonstrated better overall performance. Regarding the activity recognition accuracy of the test data with user-dependent training, we observed that the AL+CD condition had the highest recognition accuracy improvement of +24% of the AUC in the "sitting" class; the AL condition followed with +22%. The AL+CD condition had the highest recognition accuracy improvement of +40% in F-measure in the "lying down" class. The AL condition had the next-highest recognition accuracy improvement of +33% in F-measure in the "computer work" class. By contrast, regarding the activity recognition accuracy of the test data with user-independent training, the AL+CD condition had the highest recognition accuracy improvement of +26% of the AUC in the "lying down" class. The AL condition had the next-highest recognition accuracy improvement of +25% of the AUC in the "standing in place" class. The AL+CD condition had the highest recognition accuracy improvement of +60% in F-measure in the "lying down" class; the AL condition followed with +51%. For all conditions, the AUC and F-measure performance evaluated on the user-dependent training was higher than that of user-independent training. Recognition accuracy also slightly decreased for the test data after hyperparameter tuning. Note that the AUC and F-measure are undefined for attributes with zero entries as they have constant values for all test classes of the split chosen.

The Validity of Crowdsourced Data

We assessed the validity of the crowdsourced data to determine whether the accuracy levels are sufficiently high for application to real-world data. We conducted an experiment to quantify the performance of a classifier trained on crowdsourced data on a new user and compare it to that of a classifier trained on an existing curated dataset assuming the same activities. Additionally, we demonstrated the impact of varying crowd sizes on performance and compared to that of a curated dataset of equal size. However, in practice, collecting all same-label activities is challenging, and the crowdsourced dataset is more diverse than the datasets collected in a more controlled setting (e.g., curated data). Consequently, we carefully chose the test set that closely resembles our crowdsourced dataset in terms of the protocol and data collection process, segmentation, and labeled set of activities. To

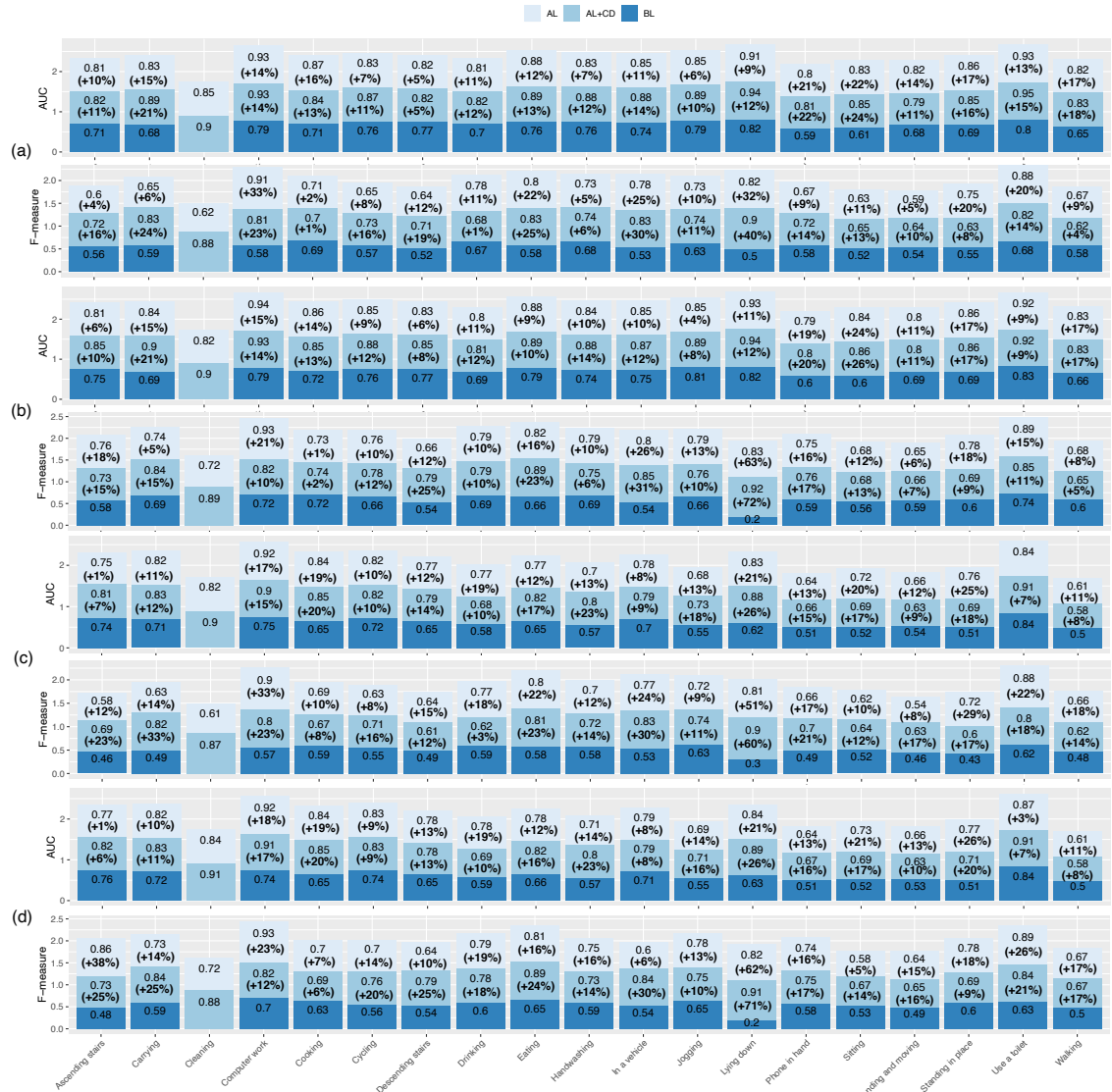


Figure 3.11 Average AUC and F-measure for each condition. The bold text shows the percentage of AUC or F-measure improvements for each class compared to that of the baseline condition. (a) AUC and F-measure / test data with user-dependent training. (b) AUC and F-measure / validation data with user-dependent training. (c) AUC and F-measure / test data with user-independent training. (d) AUC and F-measure / validation data with user-independent training.

that end, we evaluated the classifier on physical activities (PAMAP2 [129]), which is a commonly used activity recognition dataset. The PAMAP2 dataset contains 18 different physical activities captured using 3 inertial measurement units and heart rate monitor data from 9 subjects. Our implementation modeled 6 activity classes as defined in the dataset, including modes of locomotion (e.g., lying down, cycling, walking, standing, and sitting) and high-level activities (e.g., cleaning). As for the evaluation process, we used the same approach employed in the evaluation of crowdsourced data. We divided the dataset

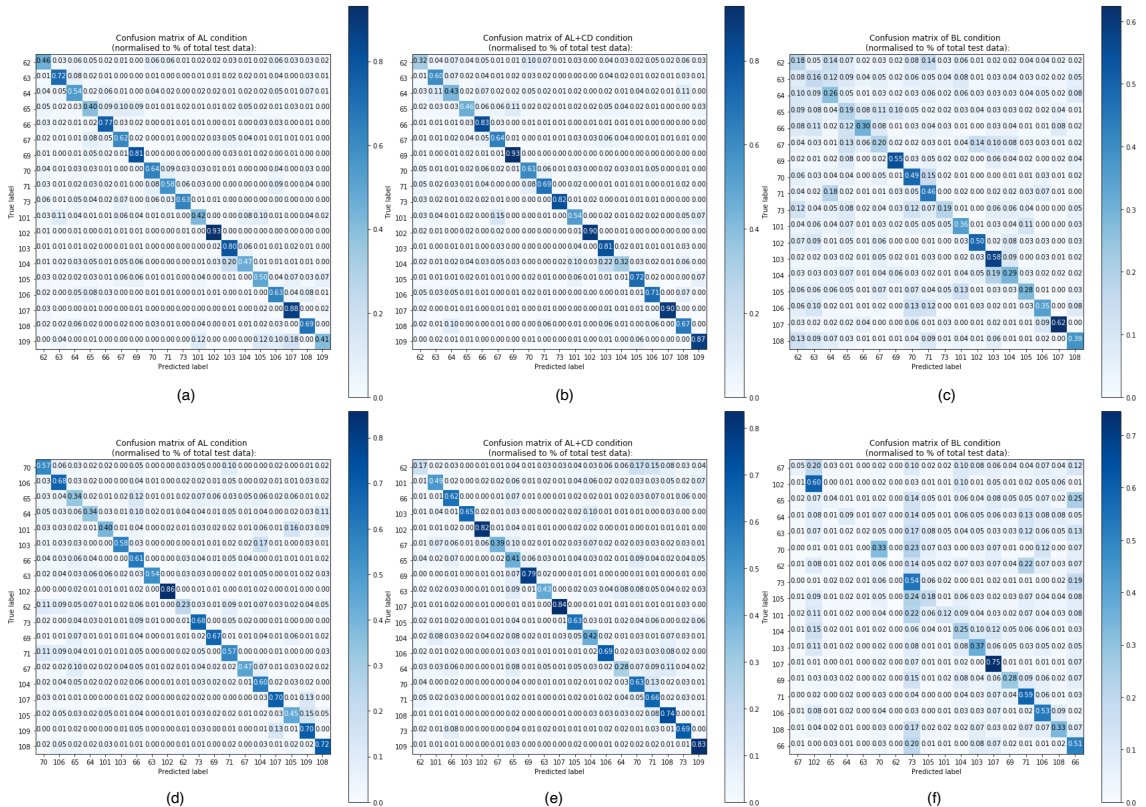


Figure 3.12 Normalized confusion matrices for all conditions. (a) AL condition with user-dependent training. (b) AL+CD condition with user-dependent training. (c) BL condition with user-dependent training. (d) AL condition with user-independent training. (e) AL+CD condition with user-independent training. (f) BL condition with user-independent training. In each matrix, the activity IDs represent the class labels; each ID corresponds to the activities in Figure 3.10.

into a training and test set, applied one-class classification for each class, evaluated the classifier with balanced samples, and used the F-measure as a metric of accuracy.

Overall, a pre-existing dataset slightly outperformed active learning for deployed activities since crowdsourced data labeling was more noisy and error-prone than curated conventionally collected training data. However, we found that the accuracy levels are acceptable for a real application, and the crowdsourced dataset is sufficient to generalize to the diversity of users and conditions these devices were deployed into, as shown in Figure 3.13. Figure 3.13a presents the learning curves of the average F-measure over all classes of crowdsourced and PAMAP2 datasets for varying sizes. Intuitively, training on more data increases the model’s performance for both datasets. We observed that using only 70% of the training data, which corresponds to approximately 7,000 data samples, the crowd model achieved the closest performance to the PAMAP2 model with a difference of approximately 3.66% in the F-measure (an F-measure of 84.58% and 88.24% for the crowdsourced model and PAMAP2, respectively). Figure 3.13a indicates that the performance is approximately equal to the PAMAP2 model with a maximum difference of

5.5% in the average F-measure when training on 40% of the training data. Figure 3.13b depicts the F-measure evaluation results of the classifier trained on crowdsourced data compared to those of the PAMAP2 dataset using 100% of the training data. Training with 100% of the data, which corresponds to approximately 10,000 data samples, results in a comparable performance to that of the PAMAP2 model with a difference of only 1% in F-measure for the "sitting" class. Training the crowdsourced model on the entire dataset achieved a performance with an average F-measure of 85% (SD = 0.03) for all activities; 90% for the "walking" class and 80% or higher for the other 5 activities.

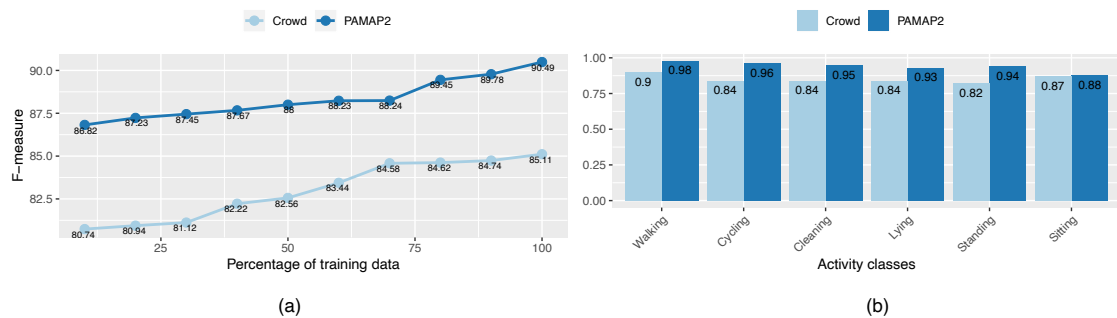



Figure 3.13 (a) Learning curves of average F-measure over all classes of crowdsourced and PAMAP2 data for varying sizes. (b) F-measure of the classifier trained on crowdsourced data compared to PAMAP2 data using 100% of the training data.

3.7.3 Worker Engagement

In this section, we verify the engagement of participants on their labeling tasks by implementing the USE short form. We recorded how often participants pressed the () button to view their information, as detailed in Section 3.4.2. We also measured the percentage of notifications that were followed within 120 seconds via a button press. By proposing the CTR as a metric, which is simply the ratio of clicks on a smartphone notification to the number of impressions, we found that a higher CTR reflects higher engagement. By comparing the results to these values across three conditions, we can estimate how effective our conditions are. Lastly, we further investigated the relationships between the button press, CTR, and activity recognition accuracy. In the following subsections, we report the results presenting a statistical significance across conditions.

User Engagement Scale

Figure 3.14 displays the distribution of data from the UES-SF questionnaire results, and Table 3.6 lists the UES-SF scores on average across the three experimental conditions. The Kruskal-Wallis test (expected $\alpha = 0.0125$) shows that the overall scores of UES-SF present no significant difference across three conditions. However, considering the mean scores for each category, a Kruskal-Wallis test revealed that focused attention could significantly affect the UES-SF score ($H(2) = 3.4e+01$, $p = 5e-08$) (Figure 3.14b). Contrarily, perceived

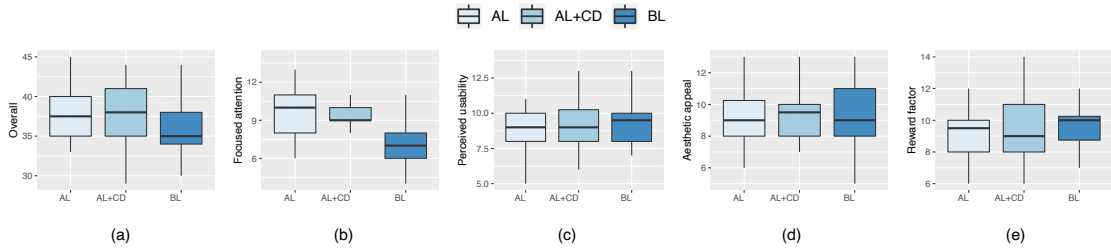


Figure 3.14 Results of the UES-SF questionnaire, grouped by category; (a) overall, (b) focused attention, (c) perceived usability, (d) aesthetic appeal, and (e) reward factor.

usability, aesthetic appeal, and reward factor were found to have no significant impact. The Dunn’ s test with Bonferroni correction on the different focused attention scores between conditions showed that both the BL versus AL and BL versus AL+CD conditions were statistically significant (Dunn’ s test, $p = 7.897e-07$ and $p = 2.856e-06$, respectively). p-values between-task pairwise independent t-tests revealed that the mean focused attention of the AL and AL+CD conditions were significantly higher than that of the BL condition. The mean rank focused attention score was 3.24, 3.18, and 2.45 for the AL, AL+CD, and BL conditions (higher score was best).

Table 3.6 The UES-SF score ($m \pm std$: mean and standard deviation) of all labeling tasks under the three conditions.

Category	AL	AL+CD	BL
Focused attention	3.24±0.66	3.18±0.52	2.45±0.61
Perceived usability	3.06±0.47	3.14±0.49	3.18±0.53
Aesthetic appeal	3.14±0.54	3.14±0.48	3.13±0.61
Reward factor	3.15±0.55	3.18±0.55	3.21±0.52
Overall	3.15±0.29	3.15±0.3	3±0.28

Button Press and CTR on the Smartphone

In this section, a button press refers to the (👤) button. In total, we recorded 3,442 button presses and 83.4% of CTR for push notifications over the three conditions of the experiments. Figure 3.15a-b displays the distribution of data from button presses and CTR for push notifications for each condition. Table 3.6 lists both the average number of button presses and CTR across three experimental conditions.

We found that the condition significantly impacts the number of button presses, with less presses in the BL condition as compared to the AL or AL+CD conditions (Figure 3.15a). On average, participants checked their information 24 times per labeling task in the BL condition, 32 times in the AL condition, and 30 times in the AL+CD condition ($H(2) = 20$, $p = 5e-05$). We measured the percentage of button presses occurring within 120 seconds

of a push notification. However, we did not find a significant dependence on condition ($H(2)$ 0.42, $p = 0.81$) for these estimates. In all three conditions, participants checked their information over 80%. Overall, the CTR for push notifications was higher in the AL condition compared to that of the other conditions.

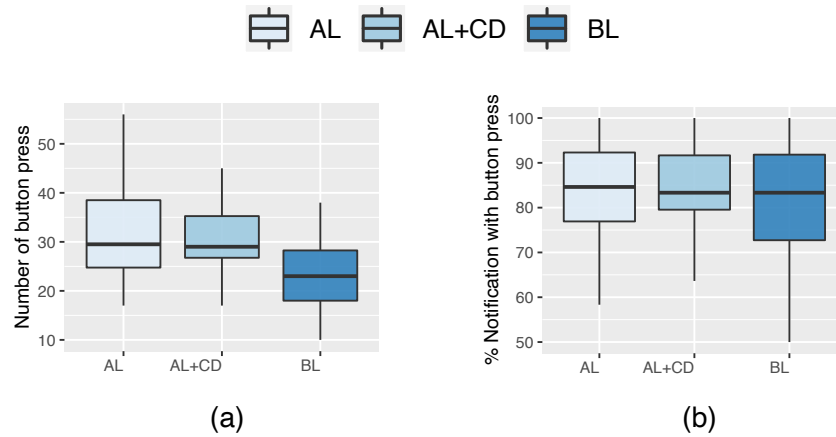


Figure3.15 figure

(a) Number of button presses. (b) Notification CTR.

Table3.7 Average number of button presses ($m \pm \text{std}$ corresponds to the mean and standard deviation) push notification CTR.

Category	AL	AL+CD	BL
Button press	32.2 ± 9.90	30.4 ± 6.20	23.5 ± 7.61
CTR (%)	83.65 ± 14.3	83.19 ± 12.84	82.02 ± 13.72

Regression Analysis

We hypothesized that if users recognize the gamification points as feedback, their performance on labeling tasks may improve. Thus, we further explored the relationships between the number of button presses, CTR, and activity recognition accuracy, in addition to the analyses discussed above.

We began by running linear regression to understand whether there is a relationship between the number of button presses or CTR and recognition accuracy, and how strong is that relationship (i.e., if it is possible to accurately predict the activity recognition accuracy given a certain number of button presses or a CTR value). We intuitively set out to reject the null hypothesis (H_0 : there is no relationship between X and Y) and considered the alternative hypothesis (H_a : there is a relationship between X and Y) if the 95% confidence interval does not include zero. By performing the regression, the p-value for the number of button presses and CTR were far less than 0.05, allowing us to conclude that the number of button press and CTR versus were related to the recognition accuracy.

Note that we generally ignored the p-value for the interception.

Once we have rejected the null hypothesis in favor of the alternative, we quantified the extent to which the model fits the data. We computed R-squared (R^2) to measure the proportion of variability in recognition accuracy that can be explained using the number of button presses or CTR value. Figure 3.16a and 3.16b display the linear regression of activity recognition accuracy on the number of button presses and CTR for push notifications along with the R^2 statistic for each condition, respectively. We noticed that the R^2 values in all conditions for both the number of button presses and CTR were over 0.7 (close to 1 was best), indicating that a large proportion of the variability in the response could be explained by the regression.

While the approach of fitting a separate simple linear regression model for each predictor is probably not entirely satisfactory, we performed multiple regression using simultaneously the number of button presses and CTR for predictions. Consequently, the number of button presses and CTR had significant p-values ($p < 0.05$). Therefore, we rejected the null hypothesis for the number of button presses and CTR, since we observed an association between those features and the activity recognition accuracy. Because R^2 is susceptible to overfitting for the multiple regression analysis, we also interpreted the adjusted R^2 that penalizes model complexity. Interesting results were obtained as both the R^2 and adjusted R^2 reached over 70% ($R^2 = 0.743$ and adjusted $R^2 = 0.738$).

We considered the network plot using multidimensional clustering for the two predictor variables (number of button presses and CTR) and the response variable (recognition accuracy), as displayed in Figure 3.16c. Notice that the correlation between the number of button presses and recognition accuracy tends to be higher than the CTR and recognition accuracy.

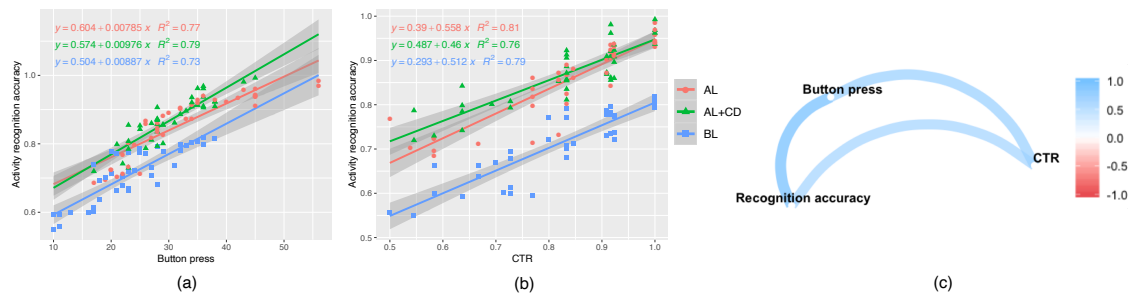


Figure 3.16 (a) Linear regression of activity recognition accuracy on the number of button presses. (b) Linear regression of activity recognition accuracy on the CTR for push notifications. (c) A network plot of a correlation data frame in which variables that are more highly correlated appear closer together and are joined by stronger paths. Paths are also colored by their sign (blue for positive and red for negative).

3.7.4 Inaccurate Data

In this section, we evaluate and compare the inaccurate data in the collected data under all conditions. In online crowdsourcing experiments, we only applied inaccuracy detection (Algorithm 3) to the AL+CD condition; therefore, we performed an offline analysis for other conditions. We applied the trained inaccuracy model of Section 3.3.2 that had assessed the performance and obtained ground truth (accurate and inaccurate) data for this analysis because of the model's high performance on the training, test, and validation datasets (the model had an AUC score of 0.98, indicating its potential to detect the false negatives and false positives). Given the trained inaccuracy model, we could evaluate the inaccurate data in the collected data as follows:

- We used the experimental results from all users in each condition (i.e., post-data collection) as raw data for the analysis.
- We predicted the class (accurate or inaccurate) of these data instances using the inaccuracy detection model.
- We counted the number of occurrences of an "inaccurate" label in each dataset and calculated the percentage of true and false samples.
- We averaged the results and obtained the inaccurate rate percentage for each condition based on occurrence frequency.

Figure 3.17 reports the results of inaccurate data in the collected data on average for each condition. As expected, the AL+CD condition had the lowest inaccurate rate compared to that of the AL and BL conditions. In the AL+CD condition, 20.9% of participants were detected as "inaccurate" and 79.1% as "accurate" (SD = 13.79). The corresponding percentages in BL were 37.01% and 62.99% (SD = 27.26), respectively, and in AL 33.79% and 66.21% (SD = 23.45), respectively. These results showed that with the AL+CD condition we could decrease the percentage of inaccuracies to 16.11% and 12.89%, compared to the BL and AL conditions, respectively.

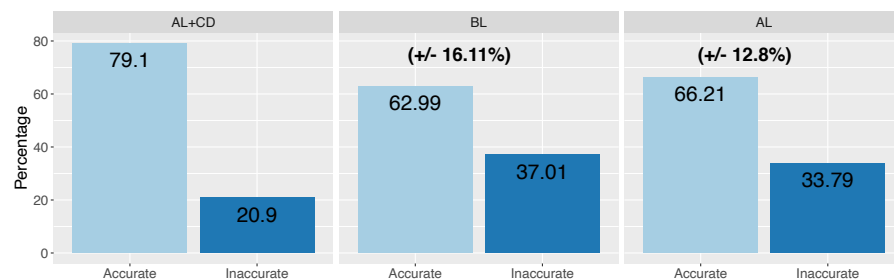


Figure 3.17 Average percentage of inaccuracies in the collected data. The AL+CD condition shows significantly less inaccurate data compared to the other conditions. +/- (%) indicates an increase/decrease in the percentage of accurate/inaccurate for each condition.

3.8 Discussion and Conclusion

In this study, we proposed CrowdAct, a system for robust activity data collection with crowdsourcing. The results showed that G1 and G3 were fully supported, and G2 was partially supported. Interestingly, we found significant and positive correlations between G1 and G2 in the corresponding directions. In addition, we conducted an experiment to quantify the performance of a classifier trained on crowdsourced data and compare it to that of a classifier trained on an existing curated dataset. We also investigated the impact of varying crowd sizes on performance to demonstrate the effectiveness of increasing the amount of labeled data. While this study enabled us to effectively collect crowdsourced activity data, there are limitations to be addressed for future research.

3.8.1 Addressing Research Goals


G1: Improving Accuracy in Crowdsourced Data Labeling with an Active Learning Algorithm

The results presented in Figure 3.11 indicate that the use of the proposed approach using active learning increased the accuracy of the AUC and F-measure by 5% to 22% and by 2% to 33% (8% to 51%), respectively, for user-dependent training. The corresponding accuracy for the AUC and F-measure for user-independent training increased by 1% to 25% and by 8% to 51%, respectively. Further, when combined with an inaccuracy detection algorithm, the accuracy of the AUC and F-measure increased by 5% to 24% and by 1% to 40%, respectively, for user-dependent training. The corresponding accuracy for the AUC and F-measure for user-independent training increased by 7% to 26% and by 3% to 60%, respectively. The use of active learning to calculate gamification points, one of the main concepts of this study, is the key enabler for this improvement. Although the use of active learning to calculate the gamification points could address the lack of accuracy in crowdsourced data labeling, we found that it has further potential. In the proposed pointing system, classes with low uncertainty received a high score, while classes with high uncertainty received a low score. To achieve a high score in most classes, a user had to collect enough samples for each class. As illustrated in Figure 3.10, for 15 out of a total of 19 classes, data quantity increase, and an additional complex activity (e.g., "cleaning") was collected. While the data labeling performance of workers seems to significantly improve with our active learning algorithm, workers can be either encouraged or discouraged by the complexity of the activity performed. For example, complex activities, such as cleaning and cooking, may discourage workers from labeling. However, because the scores of every class correspond to gamification points, workers were motivated to label data from several classes, i.e., both simple and complex activities. This further quantifies existing research; for example, the use of active learning with gamification can increase classes or enable the collection of complex activities in activity recognition. Simultaneously, the samples for each class should differ from those of other activities. In a way, this pointing system can

also reduce inaccuracies. Performing user labeling on the same activities using different labels achieves a low score in most classes. Another compelling outcome was the observed difference in the level of digital literacy. By employing 120 different workers (see Table 3.5), we found that, although the BL condition had the highest average digital literacy score of participants among all conditions, it did not impact data labeling. Conversely, using our gamified active learning approach affected recognition accuracy improvements. These results provide some interesting insight into the algorithm performance.

The study findings suggest that the use of gamified active learning can improve the quantity and quality of data in data collection. Gamification points in active learning increase the “value” of uncertain class tasks and encourage workers to provide labeled sensor data for them. Although previous studies had suggested gamification as an effective technique in experience sampling [151], the present study suggests it is also effective in ubiquitous crowdsourcing data collection tasks.

G2: Increasing in Motivation and Sustained Engagement in Crowdsourced Data Labeling with Gamification

Using the UES, we measured four dimensions of user engagement, i.e., focused attention, perceived usability, aesthetic appeal, and reward factor. We found a significant increase in the "focused attention" for the two proposed conditions (AL and AL+CD) compared to BL, as presented in the Section 3.7.3. However, even in the absence of significant differences in other dimensions, we argue that this research was fundamentally focused on the challenge of the focus attention dimension rather than other dimensions. This assumption roots from the fact that the use of gamification points is directed to focused attention (feeling absorbed in the interaction and losing track of time). Contrarily, other dimensions, such as aesthetic appeal, are related to the attractiveness and visual appeal of the interface. Therefore, it is possible to observe no significant difference between conditions, since our user interface design is similar (e.g., color, button symbols). Similarly, perceived usability is related to a negative effect, experienced as a result of the interaction, while reward factors are related to long-term user engagement. Neither is highlighted in the proposed conditions. However, we found that overall, all four dimensions received high scores, as shown in Figure 3.14. These results indicate that the crowd workers engaged with their labeling tasks. Moreover, using the user events as an engagement measure, we found a significant effect on the number of button presses for the different conditions. However, we did not observe any significant differences in CTR, as discussed in the Section 3.7.3. We found that the number of button presses was more prominent in the proposed conditions (AL and AL+CD) since the user may press the button () several times to inspect the gamification points. However, in all conditions, delivering information through notifications was not significantly different. Despite the similarities in CTR provided in all conditions, the crowd workers were aware of push notifications up to 83%. Additionally, the results implied that the crowd workers received the notifications

and feedback. This goal was therefore partially supported by this research.

Further, the two engagement measurements (button press and CTR) were strongly correlated with activity recognition accuracy. It can be observed from Figure 3.7.3 that the workers proactively launched the smartphone application to check their scores and engaged in participatory sensing as a response to notifications. Our findings support these results; gamification points could be used as a motivator for efficient activity data collection.

G3: Reducing Inaccurate Data in Crowdsourced Data Labeling

The use of gamification may increase the quantity and quality of labels, but the quality of the raw data can be compromised if the workers construct inaccurate data. To eliminate this issue, we introduced the inaccuracy detection algorithm. The goal was supported as we found that inaccurate data in crowdsourced data labeling could be reduced in our proposed AL+CD condition, as measured by the diversity of labels and their quantity, as discussed in Section 3.7.4. The inaccurate data was decreased up to 79%, 62%, and 66% in the AL+CD, AL, and BL conditions, respectively. The inaccurate data in the AL+CD condition decreased by 16% of that of the BL condition and by 13% of that of the AL condition. Another interesting finding is that the percentage of inaccurate data in both the AL+CD and AL conditions was lower than that of the BL condition, as reflected in Figure 3.17.

We, therefore, hypothesize that data labeling with an active learning algorithm (G1) and gamification (G2) can lead to more faithful workers. In other words, when improving the lack of accuracy with active learning and increasing motivation and sustained engagement with gamification, inaccurate data may also be reduced. We claim that these results provide strong indicators that future research should further examine these relationships. While we are satisfied with the achieved reduction in the inaccurate data, several limitations remain in this data collection study.

3.8.2 Limitations and Future Directions

Based on the discussion outlined above, certain recommendations and considerations can be extracted for designing gamified active learning and inaccuracy detection for crowdsourced activity data collection.

First, as seen in Figure 3.11, overall, the activities with user-dependent training achieved higher accuracy than user-independent training. We reasoned that this may occur depending on smartphone positions and orientations, which differed for each participant. We sincerely acknowledge that the scale of our data and the size of our user group is limited. However, we expect there will be no significant impact for large-scale data collection because our results imply that large-scale high-quality activity datasets can be produced with crowdsourcing. We intend to investigate this in future work.

Second, while we explored the use of "points" as a gamification element, other elements

were not explored. It would be preferable to scrutinize several elements, such as timers, badges, staging, quests, and leaderboards since the use of other gamification elements proved to be effective in motivating participants [36]. Consequently, we can employ the mechanism's knowledge and apply it to the further mechanism discussed above. Similarly, there are relevant models of user engagement that were not tracked, such as time in application. It would be of interest to track and measure the time spent on tasks as this will help us understand the level of worker engagement. While we handled the number of participants who accept the notifications, we were unable to ascertain the number of participants who explicitly dismissed the notifications. We believe that capturing this event will be valuable to gaining insight into message delivery and user engagement. Future work should attempt to explore other gamification elements and collect more user engagement metrics. Additionally, we did not explore task interruptions in smartphone notification systems (i.e., the duration of the gamification points sent to the user was designed without considering interruptability and task performance when interrupted). According to existing work, high interruptability from notifications causes users to disable (or not enable) notifications for particular applications [122]. Hence, the labeling task can be interrupted by a poorly timed smartphone notification as well. In future work, it is important to focus on the proper timing for interruption by notifications when notifying users of their gamification points. Examining opportune moments for interruptions might produce better results.

Third, we dealt with the problem of inaccuracy detection by using classic machine learning techniques. However, there are many challenges for inaccuracy detection in practice with this technique, and we intend to move forward to address them. While manually labeled inaccurate data requires tremendous human effort, the continuous and rapid evolution of cheaters makes it difficult to detect new inaccurate patterns based on existing detection rules. There are several techniques to be considered in the future. For example, to overcome manually labeled data, we can treat inaccurate data as an outlier or as an anomaly and use a consistent approach, such as implementation of an isolation forest algorithm for outlier detection [37] or a neural autoencoder for anomaly detection [169]. Similarly, to defeat unseen inaccurate patterns, we can utilize unsupervised learning, since there is no prior knowledge [22]. Although we could not verify that the participants correctly performed the required inaccurate data (i.e., participants were also instructed to cheat), the existing study [97] that deployed a similar protocol of inaccurate sample collection to ours (where they asked users to intentionally generate inaccurate data) provided evidence of the reliability of inaccurate samples by machine learning with an accuracy of over 70%. While they collected inaccurate samples from students who complied with the experimental settings, we collected inaccurate samples from crowdsourcing workers, which is more challenging. Future research should attempt to explore a verification of inaccurate samples and their challenges. While the inaccurate data of workers seems to drastically decrease with inaccuracy detection, workers can be discouraged by repeated inaccuracy

messaging. Therefore, randomly generating inaccurate-detected messages without an operating algorithm is not advisable. Further, the proposed inaccuracy detection algorithm is expected to collect enough false-positive samples to make accurate predictions. Consequently, we can avoid redundant and unclear information that might discourage workers. However, we recognize that it would be interesting to deeply analyze the workers' reactions upon receiving the 'inaccurate' message. This investigation can be explored in future work. Despite these limitations, we believe that our inaccuracy detection algorithm is representative of one of the solutions of inaccurate data in crowdsourced data labeling and is an essential first step towards understanding inaccurate data patterns in activity recognition.

Finally, as mentioned in Section 3.4.3, we completely retrained the model for active learning and made the prediction for inaccuracy detection every 15 minutes. While this was manageable, practical, and feasible to handle the load of millions of sensor data and massive crowd workers, there was a gap in receiving real-time feedback, which potentially impacted worker engagement. To address this gap, several techniques can be explored in future work. For example, incremental learning techniques [146] can be used instead to reduce the computational load of the server. Similarly, sending raw data to the server increases the network and storage load in both the device and the server. Instead, features can be calculated in the device rather than on the server to reduce overloading, e.g., on-device machine learning [102, 163]. However, the trade-off is an increase in battery consumption, which should be carefully evaluated. We are confident that further study will introduce new challenges to be explored in future research.

Chapter 4

On-Device Deep Learning Inference for Activity Data Collection

4.1 Abstract

Labeling activity data is a central part of the design and evaluation of human activity recognition systems. The performance of the systems greatly depends on the quantity and “quality” of annotations; therefore, it is inevitable to rely on users and to keep them motivated to provide activity labels. While mobile and embedded devices are increasingly using deep learning models to infer user context, we propose to exploit on-device deep learning inference using a long short-term memory (LSTM)-based method to alleviate the labeling effort and ground truth data collection in activity recognition systems using smartphone sensors. The novel idea behind this is that estimated activities are used as feedback for motivating users to collect accurate activity labels. To enable us to perform evaluations, we conduct the experiments with two conditional methods. We compare the proposed method showing estimated activities using on-device deep learning inference with the traditional method showing sentences without estimated activities through smartphone notifications. By evaluating with the dataset gathered, the results show our proposed method has improvements in both data quality (i.e., the performance of a classification model) and data quantity (i.e., the number of data collected) that reflect our method could improve activity data collection, which can enhance human activity recognition systems. We discuss the results, limitations, challenges, and implications for on-device deep learning inference that support activity data collection. Also, we publish the preliminary dataset collected to the research community for activity recognition.

4.2 Introduction

A central challenge in smartphone-based activity recognition is data annotation studies in order to assess the labels describing the current activity while this activity is still ongoing or recent to ensure that the dataset is labeled correctly. The quality and quantity of

annotations can have a significant impact on the performance of the activity recognition systems. Hence, it is unavoidable to rely on the users and to keep them motivated to provide labels. To overcome the challenge of self-labeling [97], we introduce the idea of utilizing on-device deep learning inference for optimizing activity data collection. The rapid performance increase of low-power processors and the huge demand of internet of things (IoT) applications brought new ways for deploying machine/deep learning models on edge devices. On-device machine learning by fusing the inertial sensors such as smartphones have been explored [44, 50, 126]. These findings allow the activity recognition system to be feasibly identify frequent behavioral patterns on edge devices; meanwhile, deep learning revolution in the field of machine learning tends to result in higher accuracy and performs exceptionally well on machine perception tasks on smaller devices with limited resources [90, 92, 134]. TensorFlow Lite*¹ was designed to enable easy to perform machine/deep learning inference on mobile, embedded, and IoT devices with low latency and a small binary size, “at the edge” of the network, instead of sending data back and forth from a server. Thus, we will exploit the power of on-device deep learning to provide estimated activities on a smartphone in order to optimize activity data collection.

In this study, we want to show that if we give estimated activities using on-device deep learning inference through notifications as feedback to users while they are requested for labeling, we can improve data annotation tasks for activity recognition systems. The novel idea works by the user getting estimated activities through notifications on a smartphone as feedback that motivates for efficient activity data collection. Estimated activities are automatically inferred by periodically reading short bursts of smartphone sensor data and processing them using on-device deep learning with a long short-term memory (LSTM) model [113] without the model retrained. To evaluate this contribution, we trained the model for on-device deep learning with the open dataset [80], conducted the experiments with proposed and traditional methods (see Table 4.1) to collect the evaluated dataset, validated the dataset collected using several machine/deep learning algorithms, and showed that our proposed method outperformed the traditional method. In summary, the contribution of this study are listed as the following:

- We introduce a system design of integrating on-device deep learning inference and activity recognition. We describe on-device deep learning inference using an LSTM-based method which can be used for efficient activity data collection, where estimated activities are used as feedback through notifications on a smartphone.
- We present the proposed method where we provide estimated activities using on-device inference through notifications and the traditional method where we provide simple sentences without estimated activities through notifications. Our proposed method can be applied not only to LSTM but also other models for on-device inference. To evaluate in a realistic setting, we train the model used for on-device

*¹ <https://www.tensorflow.org/lite>

deep learning with the open dataset, implement a system and deploy the system to a laboratory, conduct the experiments, review and use the dataset obtained for evaluations.

- We evaluate the quality of the proposed method using a standard activity recognition chain by comparing the performance results of several machine learning algorithms as well as a deep learning algorithm with the traditional method. We show that when estimated activities using on-device inference are provided to users as feedback, we can improve the quality of data collection (e.g., the accuracy of several machine learning algorithms has improvements with the proposed method). We also compare the quantity of data collected between the proposed method and the traditional method by showing that the amount of data collected has increased with the proposed method.
- We discuss the results, limitations, challenges, and implications for on-device deep learning inference that support activity data collection and spark future studies.
- We also publish the preliminary dataset openly as Supplementary Information in this study, which might be useful for activity recognition and the research community.

Table4.1 Experimental design.

Method	Conditional Detail
Proposed	Receive estimated activities using on-device deep learning inference
Traditional	Receive with messages “What are you doing?” without estimated activities.

4.3 Method

In this section, we provide a descriptive view of the proposed on-device deep learning inference for efficient activity data collection system. The architecture of this system is depicted in Figure 4.1. The system is composed of several technical building blocks including the following: (1) to build an LSTM-based deep learning model used for on-device inference; (2) to collect accelerometer sensor data and activity labels efficiently; and (3) to provide estimated activities as feedback through smartphone notifications for efficient data collection.

4.3.1 Build an LSTM Model for On-Device Inference

In this section, we propose how to build an LSTM model used for on-device inference. We employ the open dataset provided by the wireless sensor data mining (WISDM) Lab [80] to build an activity recognition model for on-device deep learning inference. A schematic diagram of the proposed LSTM-based deep learning model for activity recog-

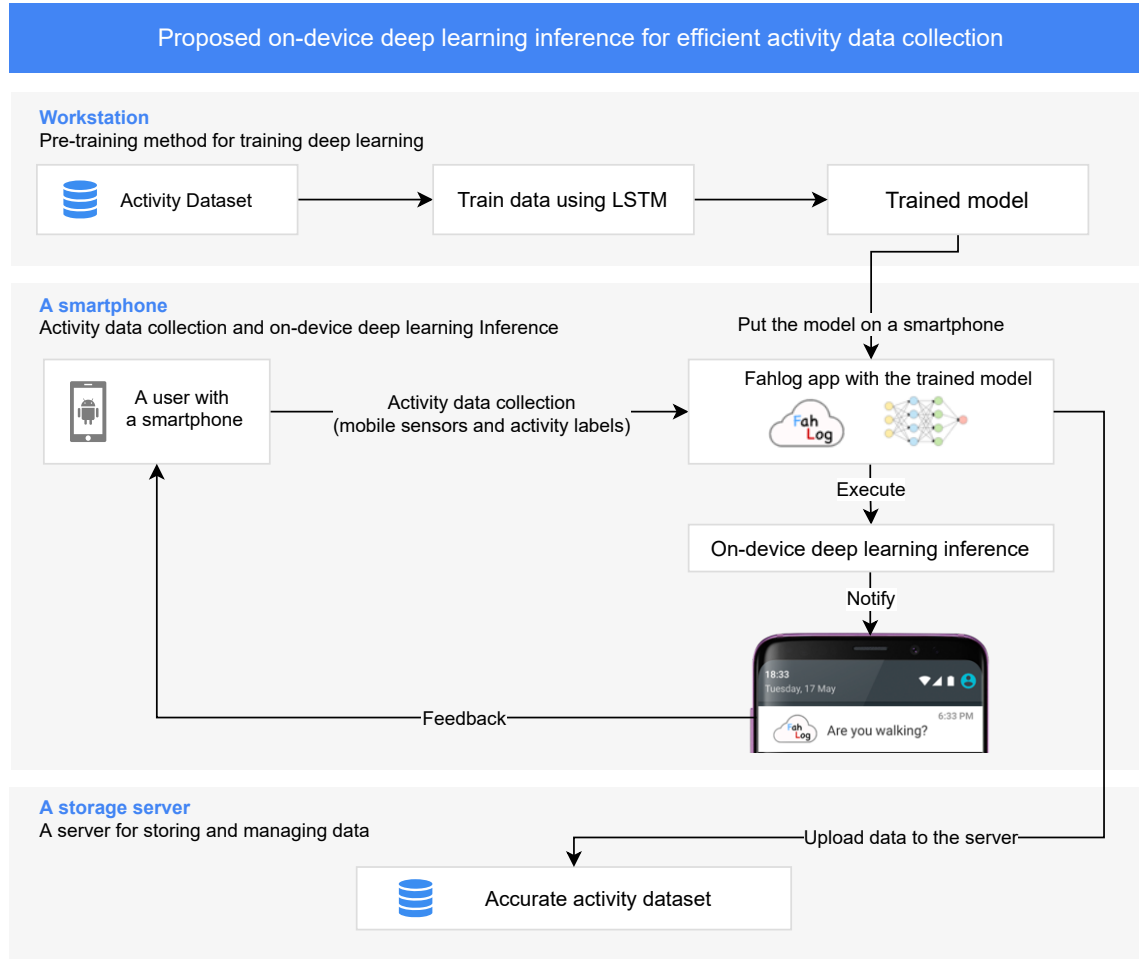


Figure 4.1 The system architecture of the proposed on-device deep learning inference for efficient activity data collection works.

nition system is depicted in Figure 4.2.

Firstly, let us describe the core idea behind LSTMs. The RNN dynamics can be described using deterministic transitions from previous to current hidden states. The deterministic state transition is a function

$$\text{RNN} : h_t^{l-1}, h_{t-1}^l \rightarrow h_t^l$$

For classical RNNs, this function is given by

$$h_t^l = f(T_{n,n}h_t^{l-1} + T_{n,n}h_{t-1}^l), \text{ where } f \in \{\text{sigm}, \text{tanh}\}$$

The LSTM has complicated dynamics that allow it to simply “memorize” information for an extended number of timesteps. The “long term” memory is stored in a vector of *memory cells* $c_t^l \in \mathbb{R}^n$. Although many LSTM architectures that differ in their connectivity structure and activation functions, all LSTM architectures have explicit memory cells for storing information for long periods of time. The LSTM can decide to overwrite the memory cell, retrieve it, or keep it for the next time step. The LSTM architecture used

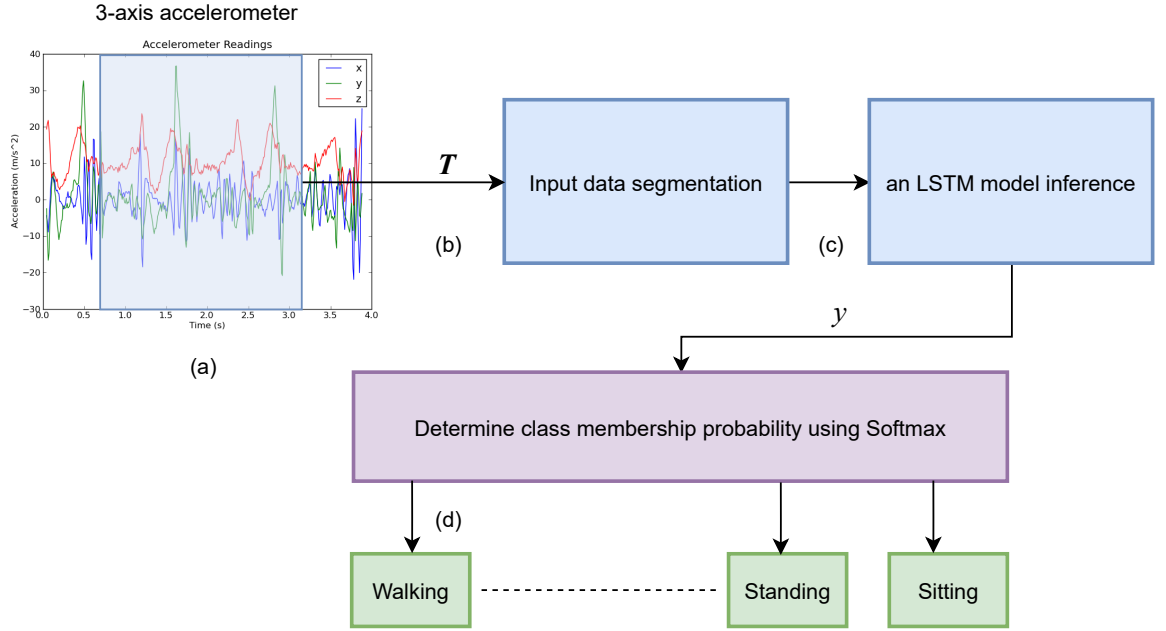


Figure 4.2 A schematic diagram of the proposed LSTM-based deep learning model for activity recognition system works as following (a) the inputs are raw signals obtained from acceleration sensors, (b) segment into windows of length T , (c) fed into LSTM-based deep learning model, (d) and finally, the model outputs class prediction for each time step.

in this paper is given by the following equations [48]:

$$\text{LSTM} : h_t^{l-1}, h_{t-1}^l, c_{t-1}^l \rightarrow h_t^l, c_t^l$$

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \text{tanh} \end{pmatrix} T_{2n,4n} \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

$$h_t^l = o \odot \tanh(c_t^l)$$

In these equations, sigm and tanh are applied elementwise. We use the WISDM dataset mentioned to build an activity recognition model. The reason why we first build the model by employing an existing dataset, and we then utilize it for our proposed on-device inference method because we concern the issue that our system cannot draw any inferences for users since it has not yet gathered sufficient information. This problem usually occurs in computer-based information systems which involve a degree of automated data modeling. It is a well-known and well-researched problem, so-called the cold start problem [133]. The human activity recognition dataset built from the recordings of 29 subjects performing regular activities while carrying a waist-mounted smartphone with

embedded inertial sensors. This dataset contains 1,098,207 examples and six attributes, including, user, activity, timestamp, x-acceleration, y-acceleration, z-acceleration without missing attribute, collected through controlled, laboratory conditions. There are 6 activity types of movement that we try to classify: walking (38.6%), jogging (31.2%), upstairs (11.2%), downstairs (9.1%), sitting (5.5%), standing (4.4%). The dataset’s description is detailed in [80].

An LSTM takes many input vectors to process them and output other vectors. In our case, the “many to one” architecture is used: we accept time series of feature vectors (one vector per time step) to convert them to a probability vector at the output for classification, as shown in Figure 4.3. As we can see from Figure 4.2, the inputs are raw

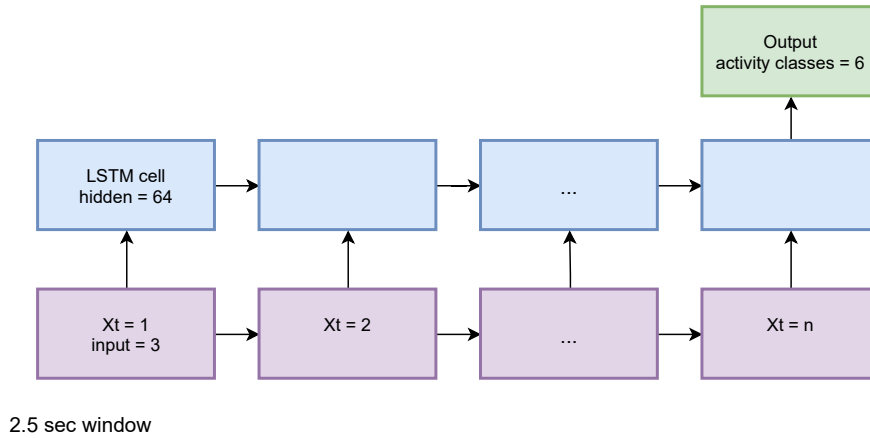


Figure 4.3 Many-to-one long short-term memory (LSTM) network architecture used for activity classification with six classes. n stands for the number of samples included in a 2.56 s window.

signals obtained from multimodal-sensors, which is a discrete sequence of equally spaced samples (x_1, x_2, \dots, x_T) , where each data point x_t is a vector of individual samples observed by the sensors at time t . These samples are segmented into windows of a maximum time index T and fed into LSTM-based deep learning model. Each generated sequence contains 200 training with 3 input parameters (3-axis accelerometer) per time steps. The model is trained for a maximum of 50 epochs by two fully-connected and two LSTM layers (stacked on each other) with 64 units each. We use rectified linear units (ReLU) for the hidden layers to increase the robustness of the model as well as remove any simple dependencies between the neurons preventing over fitting, and use the dropout technique to avoid overfitting in our model (Equation (4.1)), where a rectified linear unit has output 0 if the input is less than 0, and raw output otherwise.

$$\text{ReLU}(x) = \max(0, x). \quad (4.1)$$

Finally, the model outputs class prediction scores for each time step $(y_1^L, y_2^L, \dots, y_T^L)$, where $y_t^L \in R^C$ is a vector of scores representing the prediction for a given input sample

x_t and C is the number of activity classe, which are fed into the softmax layer to determine class membership probability.

Also, we use an optimization algorithm called Adam [47] to minimize the cost function by backpropagating its gradient and updating model parameter. The core hyper-parameters explored in this model are listed in Table 4.2.

Table4.2 Some core parameter definitions for the training.

Parameter	Value
LSTM layer	2 fully-connected
epochs	50
hidden layer units	64
output classes	6
input features per timestep	3 (acc_x, acc_y, acc_z)
timesteps per series	200
learning rate	0.0025
batch size	1024

For validating the trained model against test data, we apportion the data into training and test sets, with an 80–20 split. After each epoch of training, we evaluate the performance of the model on the validation set. We select the epoch that showed the best validation-set performance and apply the corresponding model to the test-set. As a result, we opt the final epoch that the accuracy and weighted F1-score both are reached over 97% (0.975 and 0.972, respectively) and loss is hovered at around 0.2. Note that the class distribution of the WISDM dataset has the sample imbalances among activity classes which can affect machine learning [77]. We could not collect more data that could balance our classes; however, we show the weighted F1-score for additional performance metric that is preferable if there is a class imbalance problem, not just only accuracy [121]. Since the smartphone is attached on the waist and each series to classify has just a 200 sample window, those predictions are extremely accurate. If we have a look at the confusion matrix of the model’ s predictions in Figure 4.4, we can see that our model performs really well. Although we can see some notable exceptions that there are difficulties in making the difference between walking, upstairs and downstairs, the model is almost always able to identify the movement type on a smartphone correctly. The visual insight of the results are presented in Figures 4.4 and 4.5.

4.3.2 Collect Sensor Data and Activity Labels

We requested participants to carry a waist-mounted Android smartphone (Wiko Tommy3 Plus (Android 8.1)) with embedded inertial sensors, install the mobile app on smartphones to select and record their daily life activities from the list of predefined labels. Information

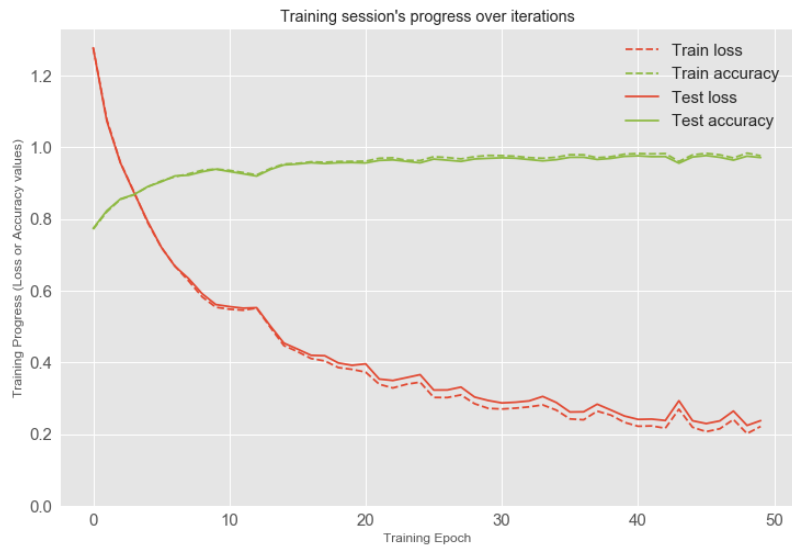


Figure4.4 Training session's progress over iterations.

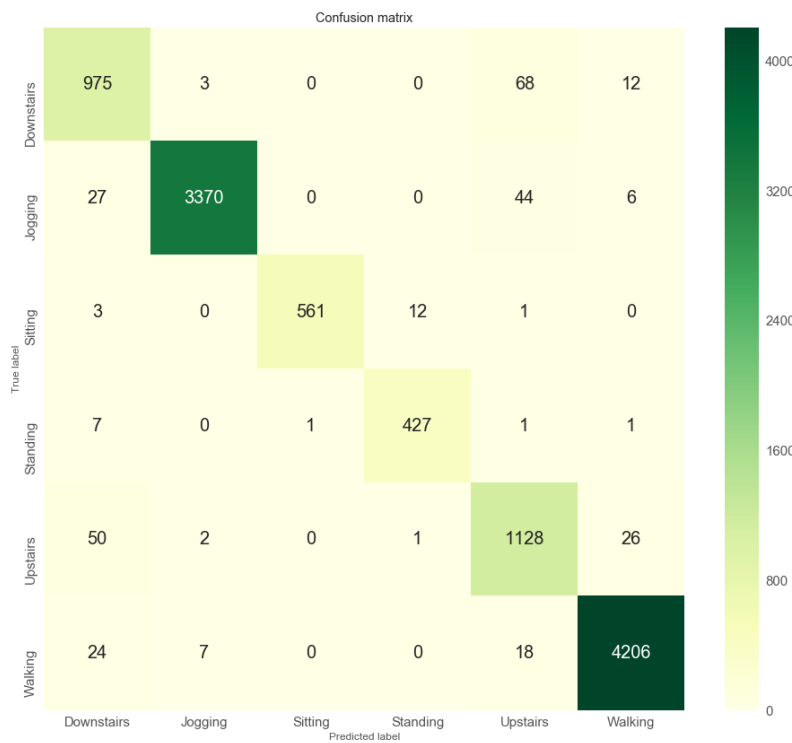


Figure4.5 The results for a classifier of the LSTM model.

about the demography of participants and the duration of the experiment are reported in Section 4.4. The mobile app is extended from our work [101] called “**FahLog**”, as shown in Figure 4.6, when a user selects the activity in Figure 4.6 a, the labels for each activity class will be put into the right column as shown in Figure 4.6 d. Then the user has to record it by pushing the button to start and stop recording while they are carrying out the activity by following the steps as shown in Figure 4.6 b–d. Each time the user

taps an activity label box, it will transition to before start (▶) → doing activity (⊙) → finish (✓) so a user can record the start and end of the activity. Since another activity may be performed while performing one activity, multiple activity labels can be started and ended in parallel. The activity labels can then be uploaded to the server when it is connected to the network. Otherwise, data will be stored on the smartphone until there is internet access. Moreover, we capture sensors and activity labels through smartphones to recognize activities using smartphone sensors continuously. Hence, we set the sampling rate of the app for the ‘standard’ settings of Android programming API, which is the slowest setting, where they are sampled 200 milliseconds when they are not busy, then we take one minute time windows for calculating time windows, it is enough of a sampling rate for such data collection. Note that the FahLog annotation tool will be unexpired and can be applied for other activity data collection experiments such as crowdsourcing [98] or nursing cares [66]. Also, it provides on Google Play openly^{*2}. It can alter multiple activity types from our server configuration connected to the app that has already been set up^{*3} as well as it can modify the user interface of the tool for each specific purpose (e.g., in this study, adding notifications for showing estimated activities as feedback). More features have been published in [101].

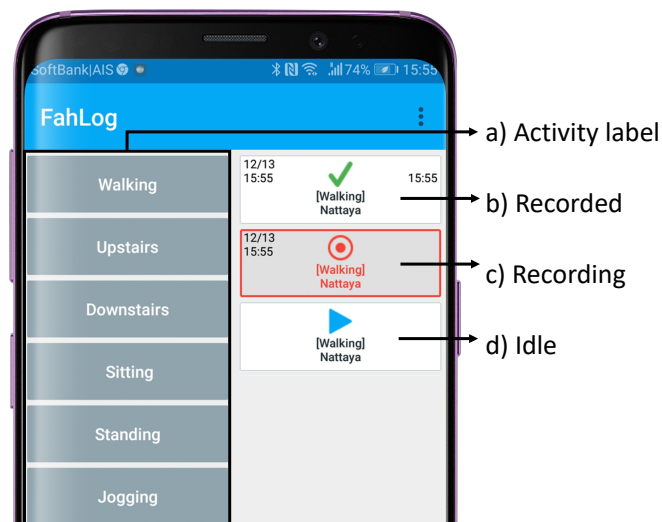


Figure4.6 FahLog: a mobile app for collecting sensor data and activity labels.

4.3.3 Provide Estimated Activities as Feedback

We interpret the results that retrieve from model inference. We use a list of probabilities that the model returned. We then meaningfully map them to relevant categories (activity classes) and present it on mobile notification to the user. Figure 4.7 presents an example of the results that are displayed on a notification. Note that to prevent excessive inter-

^{*2} <https://play.google.com/store/apps/details?id=jp.sozolab.fahlog&hl=en>

^{*3} <https://fahact.sozolab.jp>

ruptibility and to optimize resources, we stop activity reporting if the device has been still for a while, and use low power sensors to resume reporting when it detects changes in the user's activity (e.g., changing from walking to running) with mean inference time of 2846.0 ms. Also, when we put the deep learning model on the device and use a battery monitor application for Android smartphones to monitor the battery level, it increases energy consumption on its smartphone by 5% on average compared with the traditional manner without the on-device model. Therefore, showing that it works fast and does not waste a lot of energy.

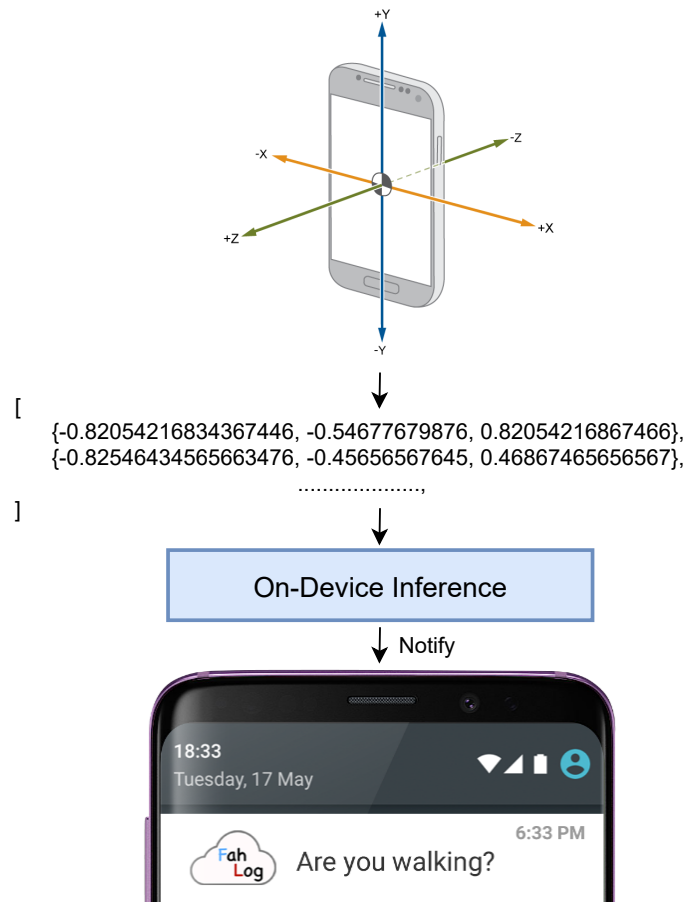


Figure4.7 Steps to show an estimated activity as feedback to a user.

4.4 Experimental Evaluation

In this section, we evaluate the proposed method using a standard activity recognition chain [10] by comparing its performance with the traditional method, as shown in Table 4.1. We describe the designed and conducted the experiments, described the dataset collected, pre-process the data collected, build the recognition model, and evaluate it.

4.4.1 Experimental Setup

The participants were required to carry a waist-mounted Android smartphone, install the FahLog app on the phones, to select and record their activities from the list of predefined labels (depicted in Figure 4.6), get notifications, and submit data to our server. Each participant performs the experiments for 6 days. Table 4.1 shows the detail of the proposed method and traditional. We propose that if we give estimated activities using on-device deep learning inference as feedback to users through smartphone notifications, they can improve activity data collection. Therefore, to compare our proposed method with the traditional method, we created notifications on smartphones that displayed two different versions. Each version only differed in the user interface where the proposed method showed estimated activities using on-device deep learning inference when the device detects changes in the user’s activity. On the other hand, the traditional method showed messages “What are you doing?”, without estimated activities once every 15 min. We also request the users click the push notifications sent to assure that the users have seen the notifications. Each participant received both conditions, each of which showed three days. We randomly displayed the conditions for each participant to ensure that they were not affected by the day of experiments for each term. The participants were instructed with detailed instructions on how to do all process step by step using the same protocol provided. During data collection, the dataset was collected in the “wild” because the subjects provided data from their daily lives.

4.4.2 Data Description

The dataset was collected between June 2019, from six subjects within an age bracket of 25–30 years, performing one of six regular activities (as shown in the left column of Table 4.3) while carrying a waist-mounted Android smartphone that recorded the movement data (accelerometers in smartphones). Note that we requested them to carry a smartphone in the same position as the WISDM dataset used to train the on-device recognition model. As a result, we gathered 713 activity labels from all participants.

4.4.3 Activity Recognition Using Smartphone Sensors

Since we propose a standard activity recognition chain and a supervised learning approach for evaluations, we first preprocess the dataset collected and then evaluate it.

Data Preprocessing

We put together the dataset by including three-axis accelerometer sensor data and the activity labels on the smartphones without clock and time synchronization because the sensor and the labeling system are both in the same device.

We used sliding windows of one minute with no overlapping. For each axis, aver-

Table4.3 The number of activity labels collected.

Activity Class	# labels
Walking	247
Jogging	1
Sitting	249
Standing	153
Downstairs	36
Upstairs	27
Total	713

age, standard deviation, maximum value and minimum value were extracted as features. An example of feature extraction is shown in Table 4.4. Before data proceeding, we excluded missing values. As a result, we obtained multivariate data of 9,129 samples with 12 variables for feature vectors.

Table4.4 An example of feature extraction.

Feature	Value
$mean_x$	num -0.82054216867469876 ...
max_x	num -0.622 ...
min_x	num -1.207 ...
sd_x	num 0.085123482931909022 ...
$mean_y$	num 1.3659708029197057 ...
max_y	num 5.468 ...
min_y	num -4.118 ...
sd_y	num 1.1740472194146572 ...
$mean_z$	num 9.819719626168224 ...
max_z	num 9.909 ...
min_z	num 9.742 ...
sd_z	num 0.894526836753883 ...

Figure 4.8 shows the activity labels distribution of the data samples in our dataset. It is worth noting that the distribution was highly skewed, where some classes appeared more frequently than others. Since the imbalanced dataset can negatively influence the generalization and reliability of supervised learning algorithms, we employed the SMOTE algorithm: synthetic minority over-sampling technique as presented in [24] (an oversampling technique that creates new synthetic data samples in the minority classes, varying the features values of the existing data points based on their k nearest neighbors in the feature space) in order to balance our dataset. By upsampling the size of training and

testing datasets separately.

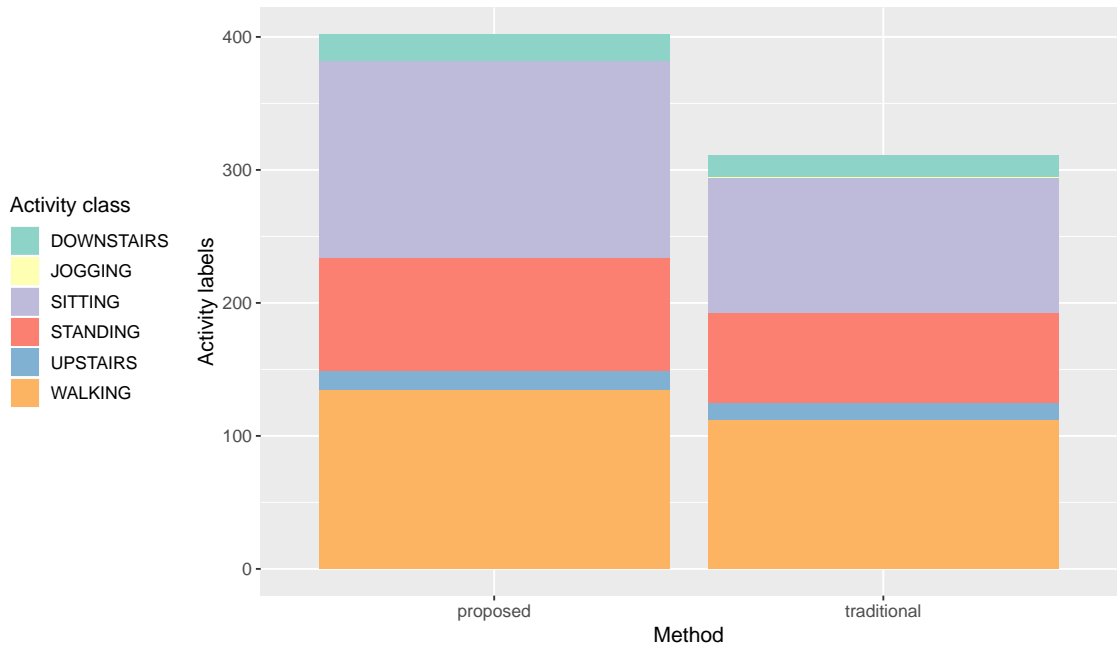


Figure 4.8 The number of activity labels for each method.

Evaluation Method

In this section, we present the effectiveness of the proposed method when we give estimated activities using on-device deep learning through smartphone notifications. The experiment was designed to test the performance of our classifier for a user-dependent scenario. In this case, the classifiers were trained and tested for each individual with her/his own data, and average accuracy and was computed. We show that the performance of several machine algorithms and LSTM have improvements with our method. We also show that the proposed method has improvements in the amount of data collected. To evaluate the proposed method using a technique of supervised learning algorithm for multiclass classification. We trained each participant separately using one deep learning classifier and several standard machine learning classifiers, including LSTM in the same way of the on-device model trained, logistic regression (LR) [61], linear discriminant analysis (LDA) [81], k-nearest neighbors (KNN) [71], decision tree (CART) [131], naive Bayes (NB) [130], support-vector machine (SVM) [145], and random forest (RF) [18].

To test the model’s ability we used stratified k-fold cross-validation. The folds are made by preserving the percentage of samples for each class to ensure each fold is a good representative of the whole. To account for label imbalance, the model performance was presented using the weighted average of precision, recall, F1-score of each class for the multiclass task. (i.e., averaging the support-weighted mean per label) So the average was weighted by the support, which was the number of samples with a given label. The “weighted” precision or recall score is defined in Equation (4.2). The same weighting is

applied to F1-score.

$$\frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \phi(y_l, \hat{y}_l) \quad (4.2)$$

- L is the set of labels
- \hat{y} is the true label
- y is the predicted label
- \hat{y}_l is all the true labels that have the label l
- $|\hat{y}_l|$ is the number of true labels that have the label l
- $\phi(y_l, \hat{y}_l)$ computes the precision or recall for the true and predicted labels that have the label l . To compute precision, let $\phi(A, B) = \frac{|A \cap B|}{|A|}$. To compute recall, let $\phi(A, B) = \frac{|A \cap B|}{|B|}$.

Note that weighted metrics is the performance of infrequent classes are given less weight since $|\hat{y}_l|$ will be small for infrequent classes. Therefore, weighted metrics may hide the performance of infrequent classes, which may be undesirable.

4.5 Results

Following the evaluation approach discussed above, we report our results of the validation together with a discussion of such results. We show the proposed method had improvements in data quality (the classification performance) compared to the traditional method. The average classification performance of all models results are shown in Figure 4.9. The F1-score performance results for each model are shown in Figure 4.10. The precision performance results for each model are shown in Figure 4.11. The recall performance results for each model are shown in Figure 4.12. The average classification performance results of all models for each user are shown in Table 4.5.

We also show that the proposed method has improvements in data quantity (the number of data collected) compared to the traditional method. Figure 4.8 shows the number of collected activity labels for both methods.

4.5.1 Quality of Collected Activity Data

Figure 4.9 shows F1-score, precision, and recall performance results of all machine learning models were improved with our proposed method compared to the traditional method. The F1-score was improved from 0.6240 to 0.7620 (**+0.138**) The precision was improved from 0.6440 to 0.7802 (**+0.136**) The recall of improved from 0.6366 to 0.7677 (**+0.131**).

Figure 4.10 shows F1-score performance results of all machine learning models were improved with our proposed method compared to the traditional method. The F1-score of CART was improved from 0.657 to 0.770 (**+0.113**) The F1-score of KNN was improved from 0.667 to 0.801 (**+0.134**). The F1-score of LDA was improved from 0.604

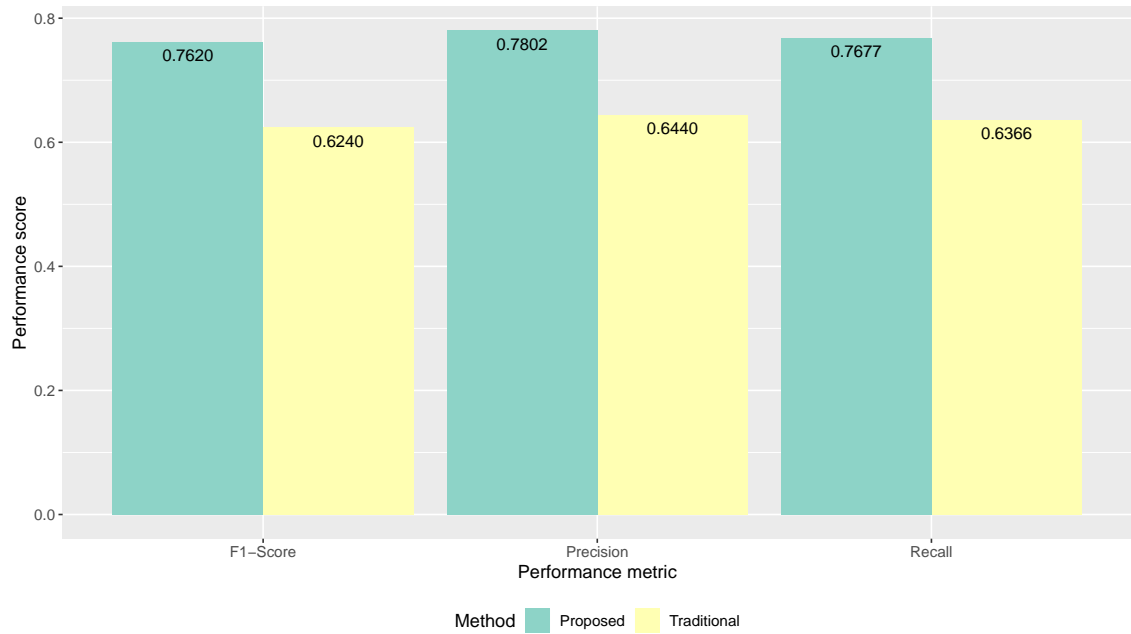


Figure4.9 The average classification performance of all models for each method.

to 0.766 (+**0.162**). The F1-score of LR was improved from 0.623 to 0.778 (+**0.155**). The F1-score of LSTM was improved from 0.657 to 0.783 (+**0.126**). The F1-score of NB was improved from 0.472 to 0.606 (+**0.134**). The F1-score of RF was improved from 0.694 to 0.815 (+**0.121**). The F1-score of SVM was improved from 0.623 to 0.775 (+**0.152**).

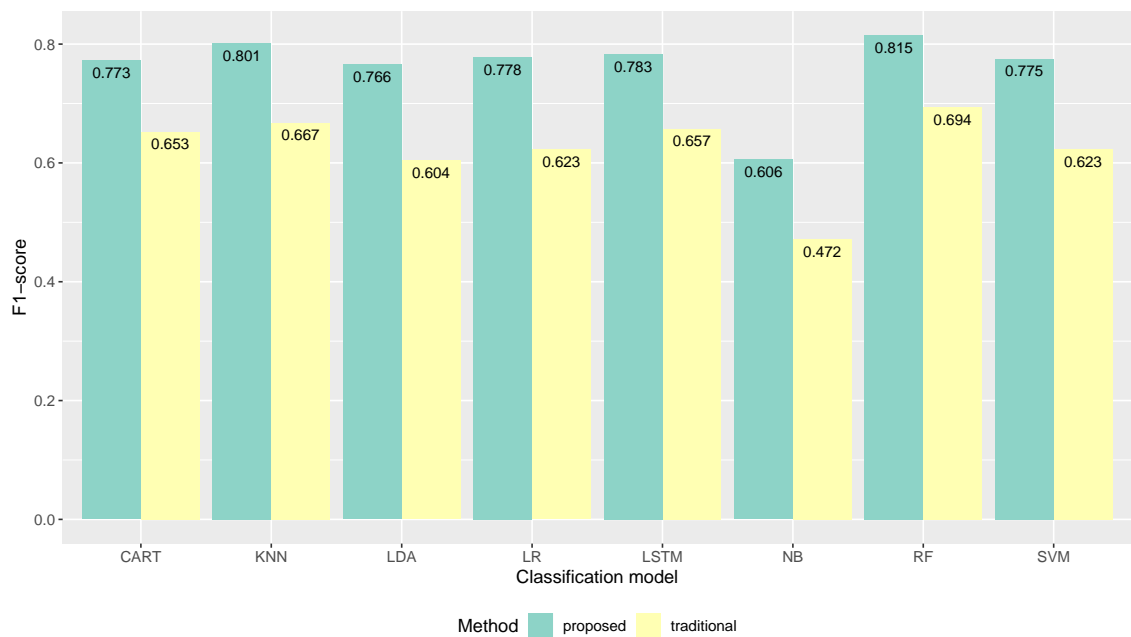


Figure4.10 The F1-score performance results of several machine learning models.

Figure 4.11 shows precision performance results of all machine learning models were

improved with our proposed method compared to the traditional method. The precision of CART was improved from 0.679 to 0.805 (+**0.126**). The precision of KNN was improved from 0.665 to 0.793 (+**0.128**). The precision of LDA was improved from 0.611 to 0.762 (+**0.151**). The precision of LR was improved from 0.616 to 0.759 (+**0.143**). The precision of LSTM was improved from 0.675 to 0.803 (+**0.128**). The precision of NB was improved from 0.593 to 0.757 (+**0.164**). The precision of RF was improved from 0.698 to 0.813 (+**0.114**). The precision of SVM was improved from 0.619 to 0.738 (+**0.119**). Figure 4.12 shows recall performance results of all machine learning models were improved

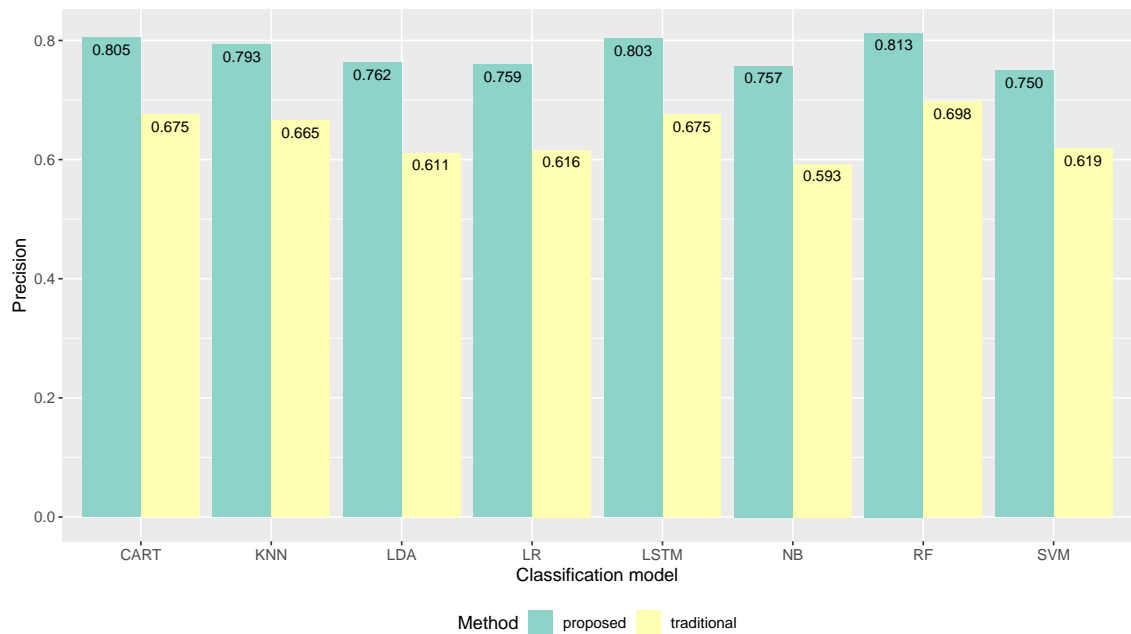


Figure4.11 The precision performance results of several machine learning models.

with our proposed method compared to the traditional method. The recall of CART was improved from 0.648 to 0.746 improve(+**0.098**). The recall of KNN was improved from 0.681 to 0.814 improve(+**0.133**). The recall of LDA was improved from 0.626 to 0.780 improve(+**0.154**). The recall of LR was improved from 0.657 to 0.806 improve(+**0.149**). The recall of LSTM was improved from 0.657 to 0.779 improve(+**0.121**). The recall of NB was improved from 0.459 to 0.556 improve(+**0.097**). The recall of RF was improved from 0.696 to 0.821 improve(+**0.137**). The recall of SVM was improved from 0.677 to 0.833 improve(+**0.156**).

Table 4.5 shows all users improve average F1-score, average precision, and average recall performances of all machine learning models with our proposed method compared to the traditional method.

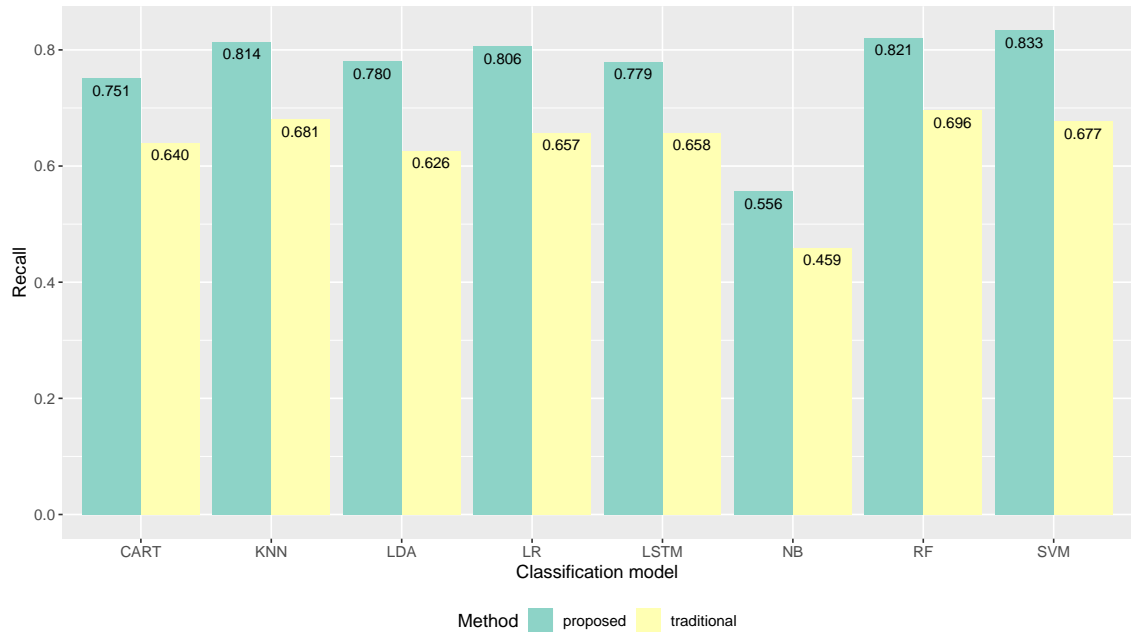


Figure 4.12 The recall performance results of several machine learning models.

4.5.2 Quantity of Collected Activity Data

Figure 4.8 shows the number of collected activity labels was increased with our proposed method. The number of activity labels increased from 311 to 402 (+91) compared to the traditional method.

Table 4.6 shows the number of labels of each activity class by comparing the proposed and traditional. While some activity classes have more labels with the proposed method, only one class has fewer labels with the proposed method.

The number of walking labels was increased from 112 to 135 (+23). The number of upstairs labels was increased from 13 to 14 (+1). The number of standing labels was increased from 68 to 85 (+17). The number of sitting labels was increased from 101 to 148 (+47). The number of downstairs labels was increased from 16 to 20 (+4). The number of jogging labels was decreased from 1 to 0 (-1).

4.6 Discussion and Future Directions

By evaluating the dataset and comparing with the traditional method, the results reflect that our proposed method has improvements in data quality (the performance of a classification model) for all machine learning models evaluated and data quantity (the number of labels collected) that indicate improvements in activity data collection. What we have found most interesting is that all users improve quality of activity data collection with the proposed method, as shown in Table 4.5. While RF achieves the highest F1-score

Table 4.5 The average

classification performance of all models for each user.

User	Method	F1-Score	Recall	Precision
98	proposed	0.7778	0.7756	0.7973
98	traditional	0.7127	0.7139	0.7391
98	Improvement	+0.0651	+0.0616	+0.0582
99	proposed	0.7009	0.7119	0.7156
99	traditional	0.4442	0.4830	0.4605
99	Improvement	+0.2567	+0.2289	+0.2551
101	proposed	0.8700	0.8774	0.8727
101	traditional	0.6449	0.6619	0.6701
101	Improvement	+0.225	+0.215	+0.203
103	proposed	0.7693	0.7663	0.7950
103	traditional	0.6490	0.6685	0.6584
103	Improvement	+0.120	+0.098	+0.137
104	proposed	0.7881	0.7954	0.8120
104	traditional	0.6333	0.6223	0.6705
104	Improvement	+0.155	+0.173	+0.142
105	proposed	0.6658	0.6794	0.6888
105	traditional	0.6600	0.6702	0.6654
105	Improvement	+0.006	+0.01	+0.023

at 81.5%, LDA has the most improvements by 16.2%. RF achieves highest the precision at 81.3%, NB has the most improvements by 16.4%. SVM achieves highest the recall at 83.3% and also has the most improvements by 15.6%. While this study enabled us to improve activity data collection effectively, there are some limitations that we would like to point out and reference in the future.

First, while we notified information about estimated activity when the user is currently doing the activity, it might be necessary to design both our mobile app and our recognition model to identify when a user starts or stops a particular activity, such as walking, biking, or driving (e.g., detect when users start and end an activity). For activity recognition systems, it is crucial to collect correct segments data. In other words, we need a labeled sequence of activities (i.e., the start and finish times of the events). Hence, if the app can be used to detect changes in the user's activity, we can also deliver this information as feedback to the user for better activity data collection. Researchers may consider this idea for other purposes, for example, an app subscribes to a transition in activities of interest and notifies the user only when needed (e.g., the app notifies driving when a user starts

Table4.6 The number of
activity labels of each activity class for each method.

Activity Class	Proposed	Traditional	Improvement
Walking	135	112	+23
Upstairs	14	13	+1
Standing	85	68	+17
Sitting	148	101	+47
Downstairs	20	16	+4
Jogging	0	1	-1
Total	402	311	+91

driving and mute all conversations until the user stops driving).

Second, we used the WISDM dataset to train our deep learning model. Hence, the smartphone’s position is limited for activity data collection in our experiment as we have to put the smartphone in the same position. If the smartphone’s position and/or orientation is discrepant from theirs, the on-device inference will not be correct. Consequently, considering to collect training dataset by ourselves will be vital. Also, we can collect more data to make the samples balance among activity classes. Furthermore, while we applied a three-axis accelerometer for training the recognition model and inferring on a smartphone device, other smartphone sensors would be useful for more accurate recognition. For example, adding gyroscope can help indicate orientation. We will leave this for future work.

Third, we run the trained model on a device without retraining. When designing activity recognition (machine learning) systems, it is crucial to understand how our data is going to change over time. A well-architected system should take this into account, and a plan should be put in place for keeping our models updated. There are several ways to retain the model, for example, manual retraining by training and deploying your models with fresh data using the same process you used to build your models in the first place or continuous learning by using an automated system to evaluate and retrain your models continuously (e.g., hosting a model on the cloud [33]). However, retraining the model to maintain machine learning systems would be challenging for research questions in future work, for example, how do we ensure our predictions continue to be accurate? Similarly, how do we keep our models up-to-date with new training data?

Fourth, as our proposed method can be applied for several algorithms, but the main on-device inference model that drove our work—that LSTM-based deep learning model. If a training model were evaluating using other deep learning methods, such as CNN, CNN + LSTM, then there would be value in expanding—why LSTM? What are the challenges that are different from other methods? Which method is best?

Finally, we plan to evaluate the method with long-term data collection and more diverse

samples, find data insights as well as find out the correlations between accuracy, the number of activity labels and classes to show whether and how strongly pairs of variables are related. For example, do notifications affect the number of activity labels or do notifications affect the number of activity classes? Answering these questions, it would also be helpful to understand user motivations and support activity data collection further. Likewise, we have seen that although the number of activity labels is increased with our method, not all activity classes (see in Table 4.6). Therefore, analyzing the data collected more deeply will be useful to understand correlation and causation.

Chapter 5

On-Device Deep Personalization for Activity Data Collection

5.1 Abstract

One of the biggest challenges of activity data collection is the need to rely on users and keep them engaged to continually provide labels. Recent breakthroughs in mobile platforms have proven effective in bringing deep neural networks powered intelligence into mobile devices. This study proposes a novel on-device personalization for data labeling for an activity recognition system using mobile sensing. The key idea behind this system is that estimated activities personalized for a specific individual user can be used as feedback to motivate user contribution and improve data labeling quality. First, we exploited fine-tuning using a Deep Recurrent Neural Network to address the lack of sufficient training data and minimize the need for training deep learning on mobile devices from scratch. Second, we utilized a model pruning technique to reduce the computation cost of on-device personalization without affecting the accuracy. Finally, we built a robust activity data labeling system by integrating the two techniques outlined above, allowing the mobile application to create a personalized experience for the user. To demonstrate the proposed model's capability and feasibility, we developed and deployed the proposed system to realistic settings. For our experimental setup, we gathered more than 16,800 activity windows from 12 activity classes using smartphone sensors. We empirically evaluated the proposed quality by comparing it with a baseline using machine learning. Our results indicate that the proposed system effectively improved activity accuracy recognition for individual users and reduced cost and latency for inference for mobile devices. Based on our findings, we highlight critical and promising future research directions regarding the design of efficient activity data collection with on-device personalization.

5.2 Introduction

In this study, we challenge the online and self-labeling scenarios using inertial sensors, such as accelerometers. Data labeling is labeled when the individual is performing the activity of concern. human labelers must start and stop the data capture process manually to label describing the on-going activity that needs to be assessed to avoid inaccurate timestamps, which requires high effort. Although participants show initial enthusiasm, they may lose interest and drop out over time. This situation leads to low-quality data collection and biased data. Indeed, it is hard to overcome the lack of motivation and sustained engagement without any artifice [97, 98, 103, 104]. Thus, our motivation is to create a strategy to keep the participants engaged with the labeling task to obtain high-quality labels. This challenge is well-motivated in user feedback studies. [59, 97, 102]. Prior work [102] utilized inference results as feedback to improve the quality and quantity of participant contributions. however, the datasets' models from different users lose accuracy when applied to a new user due to the diversity of users' behavior. This limitation can be addressed either by training a personal model on a cloud or a device. On-cloud training has the disadvantage of high computational cost and inability to scale when training the per-user model for millions of users. Contrarily, on-device training might give model training's inefficiency because of resource-constrained devices and insufficient user data on-device.

This study bridges this gap by introducing the proposed system, allowing activity recognition applications for smartphone sensor systems to achieve highly accurate training datasets based on three features. First, we employ *Fine-tuning deep neural networks* [165]: the technique widely used in transfer learning in the context of deep learning to overcome the lack of sufficient training data. We implement fine-tuning instead of full-training, called on-device personalization, helping models stay relevant to user behavior. Second, we propose *Magnitude-based weight pruning* [173]: an optimization technique to minimize the complexity of optimizing deep learning inference for on-device personalization. Finally, we integrate the abovementioned two features to build an efficient on-device personalization system. We utilize the inference results obtained from the on-device model as feedback to motivate user engagement and improve data labeling quality.

In short, the proposed system focuses on the accuracy of human contributions in achieving high-quality and consistent ground-truth labeling and, particularly, on the impact of the “on-device personalization system” and feedback under different conditions (See Table 5.1). To be entirely sure, the experimental setup was a within-subject design; the same person tests all the conditions where each participant receives both with- and without feedback. An overview of the proposed system is shown in Figure 5.1. The contributions of this work to the field are the following:

1. We introduce a system design of integrating on-device personalization and activity recognition, which allows activity recognition applications for smartphone sensor

systems to achieve highly accurate training datasets. We developed the proposed system based on three essential features: on-device fine-tuning, model optimization, and personalized feedback.

2. We deployed the proposed system to a realistic scenario demonstrating its capability and feasibility. We gathered more than 16,800 activity windows, each labeled with their corresponding activity class from 12 activity classes using smartphone sensors. We reviewed, analyzed, and used the obtained data for evaluations.
3. We empirically evaluated the proposed system’s quality by comparing the proposed condition with the baseline condition (see Table 5.1) using machine learning. The results indicate that the proposed system can achieve accurate and consistent labeling in activity datasets.

We discuss the results, challenges, limitations, and implications of this research on the design of efficient activity data collection methods with on-device personalization.

Table 5.1 Experimental design summary.

Method	Conditional detail
Proposed	Receive estimated-feedback notifications using on-device personalization.
Baseline	Receive estimated-feedback notifications using on-device inference [102].

5.3 Preliminaries

This section provides a brief overview of multiple learning paradigms, including mobile activity recognition with deep learning, transfer learning and fine-tuning, and, importantly, on-device personalization.

5.3.1 Mobile Activity Recognition with Deep Learning

This study relies on state-of-the-art mobile activity recognition using supervised learning, the input x is sensor data (regularly represented as a set of sensor input values around time t). We typically describe an example as a vector $x \in \mathbb{R}^n$, where each x_i of the vectors is a different feature. The output y is a numeric value classifying the activity class k in the given sensor data. The learning algorithm must produce a function $f : \mathbb{R}^n \rightarrow \{1, \dots, k\}$. When $y = f(x)$, the model assigns an input defined by vector x to a category k defined by numeric value y , where f can output a probability distribution over classes. Recent activity recognition is well-developed with deep learning [158] to overcome traditional algorithms’ failure on such recognition tasks. The deep learning strategy is to learn ϕ , where ϕ can be used as a provided set of features characterizing x or a new representation for x . In this strategy, we have a model $y = f(x; \theta, \omega) = \phi(x; \theta)^T \omega$. We have parameters θ that we apply to learn ϕ from a broad class of functions, and parameters ω that map

from $\phi\omega$ to the desired output. This is an instance of a common deep learning, where ϕ defining a hidden layer. We parametrize the representation as $\phi(x; \theta)$ and utilize the optimization algorithm to find the value of the parameters θ that result in the most useful function approximation.

The use of Convolutional neural networks (CNNs) and Recurrent neural networks (RNNs) have been the subject of study of many activity recognition applications [113]. Both kinds impose challenges when applied to practical applications owing to the complexity of their architecture. In this study, we deeply explore RNNs due to the suitability of temporal data for building the proposed system blocks. We describe a detailed RNN of the proposed system in Section 5.4.3.

5.3.2 Transfer Learning and Fine-Tuning

Transfer learning intends to apply earlier acquired knowledge to accelerate the learning of new tasks [111]. In this study, let D_0 and D_1 be domains with learning tasks T_0 and T_1 , respectively. The fundamental concept is to help enhance the learning of a predictive function $f(\cdot)$ in T_1 applying the learned knowledge extracted from D_0 and T_0 , where $D_0 \neq D_1$, and/or $T_0 \neq T_1$, suggesting that domains or tasks can be different. A pre-trained model is an accumulated network earlier trained on a massive dataset. We either adopt the pre-trained model or apply transfer learning to customize this model to a given task T_1 . In this paradigm, we classify the actions of humans employing transfer learning from a pre-trained network. There have been many proposed ways of customizing a pre-trained model, such as feature extraction and fine-tuning. The major variation between feature extraction and fine-tuning is that the former is done by instantiating the pre-trained model and supplementing a fully-connected classifier on top. In contrast, fine-tuning has a significant step to incrementally increase performance by repurposing the pre-trained models' top-level layers to the new dataset. In turn, it could also possibly lead to prompt overfitting. This study employs fine-tuning to build the proposed system. We refer an interested reader to [117] for a detailed review of transfer learning.

5.3.3 On-Device Personalization

In this learning setting, we employed a fine-tuning with deep learning technique to retrain an already trained model on the cloud (that carefully trained on high-quality datasets to be as generic and unbiased as possible) to adapt to a similar mobile activity recognition problem. We only focused on two disjoint datasets that are given and the task changed, i.e., $D_0 \cap D_1 = \emptyset$ and $Y_0 \cap Y_1 = \emptyset$. The target model (on-device fine-tuned model) replicates all model designs and their parameters on the source model (on-cloud pre-trained model), except the output layer, and fine-tunes these parameters based on D_1 . Contrarily, the output layer of the target model needs to be trained from scratch. In some exceptional cases, when fine-tuning is performed for D_1 , it can cover a part of the original

one D_0 . However, to simplify notations, we ignore that parts of D_1 can already be included in D_0 . Using this technique, we can create a personalized experience for the user on the device while overcoming limited training data and computational resources. For example, returning personalized estimation activities as feedback to individual devices.

5.4 Method

This section introduces the proposed system and its learning procedure. First, we introduce an overview of the methodology. Next, we describe the dataset used to train our pre-trained model. We then provide a detailed description of the network architecture and its implementation and classification performance. Finally, we discuss an optimization process for the model.

5.4.1 Overview

The objective of our work is to apply fine-tuning using RNNs to migrate the knowledge learned from the source dataset D_{src} on the cloud to the target dataset D_{tar} on the device for mobile activity recognition to deliver better personalized feedback to the user, as reflected in Figure 5.1.

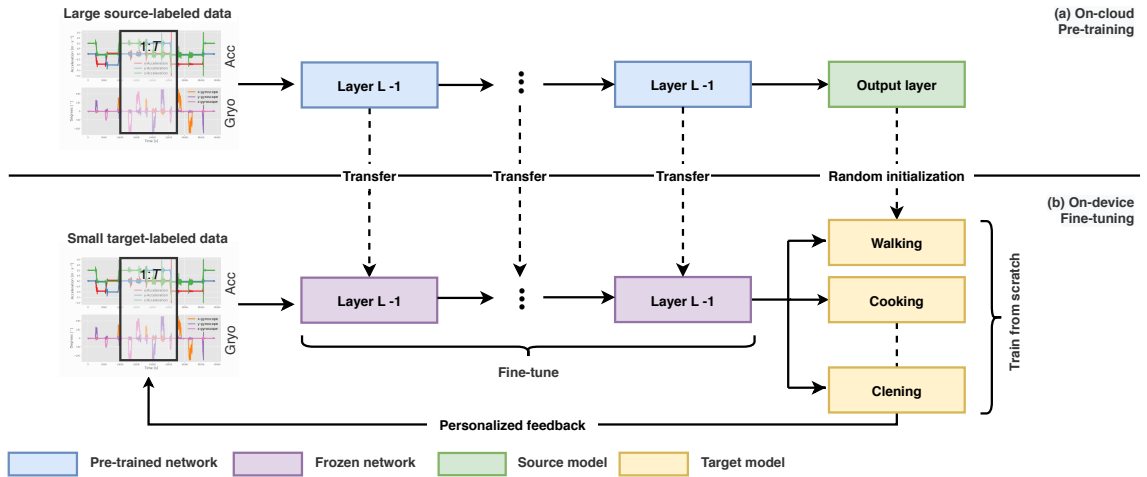


Figure 5.1 High-level overview of the proposed system. We train RNNs for activity recognition on an extensive labeled dataset Step (a). The learned features are transferred to the below activity recognition model on a device Step (b) to personalize individual devices’ prediction with a small labeled dataset. Next, the predicted activities are continuously returned as feedback for data labeling.

Although the activities in D_{src} are mostly unrelated to “walking”, models trained on this dataset can extract more general sensor features that can help identify acceleration and the rate of rotation of the device along the three sensor axes. These similar features may be equally effective for recognizing a “walking” class. Moreover, it takes less time

and requires less data than training a model from scratch. We simply selected a single fully-connected layer with softmax activation as M_{tar} in this experiment based on our preliminary study’s promising results [104]. However, we recommend researchers perform several experiments to see the effect of the number of layers to freeze and the number of layers to fine-tune before adopting. To build the proposed system, we implemented six steps:

1. Let M_{src} be the source model pretrained on the cloud; Let D_{src} be a source dataset (i.e., large-generic activity datasets); Let M_{tar} be the target model trained on individual devices; Let D_{tar} be the target dataset (i.e., small-personal activity datasets);
2. Build an input pipeline for M_{src} using RNNs. Then, pretrain M_{src} on D_{src} .
3. Create M_{tar} . This model replicates all model designs and their parameters on M_{src} , except the output layer. Assume that these M_{src} ’s parameters hold the knowledge learned from D_{src} ; this knowledge will be equally applicable to D_{tar} . Additionally, suppose that M_{src} ’s output layer closely resembles the labels of D_{src} and is consequently not used in M_{tar} .
4. Add an output layer with a specific output size (which is equal to the number of D_{tar} categories) to M_{tar} . Then, randomly initialize M_{tar} ’s parameters of this layer.
5. Train the output layer of M_{tar} on D_{tar} from scratch. The parameters of all remaining layers are fine-tuned based on M_{src} ’s parameters.
6. Execute M_{tar} to make predictions based on user’s input data (i.e., smartphone sensors and user-labeled data) to recognize activities and return estimated activities as feedback to the user.

5.4.2 Dataset

Large-scale datasets are prerequisites for the successful application of fine-tuning deep neural networks in a supervised learning manner. This study employed the dataset gathered from the real-world deployment on Amazon Mechanical Turk (MTurk) (<https://www.mturk.com/>) as D_{src} to build M_{src} . The procedure of labeling tasks of the dataset was similar to prior work [98]. The dataset has assessed the crowdsourced data’s validity to verify that the accuracy level is sufficiently high for application to real-world data. The experiments were carried out in January and February 2020 with 120 subjects (52 female, 68 male) between the ages of 22 and 57 years old (37.64 ± 9.37). Each person performed 19 activity classes carrying an application developed for an Android smartphone in their pockets. The dataset contains the readings of two embedded sensors commonly found in smartphones: accelerometer and gyroscope, sampled at a constant frequency rate of 20 Hz. We selected 12 activity classes from the entire categories: lying down, sitting, walking, standing, handwashing, cycling, eating, using a toilet, cleaning, in a vehicle, computer work, and cooking. Given this data, it is possible to create general-model

representations based on RNNs used as an initial model in the application.

5.4.3 Network Architecture and Implementation

Following our prior works [102,104], we optimally decided on the network architecture. Our preliminary findings found that RNN is incredibly well suited for sequential data because of handling arbitrary input/output lengths and the advantage of being less feature compatible when compared to other architectures such as CNN. Therefore, we employ RNNs to build the proposed system. This study explores two sequential feature models: a simple LSTM and CNN-LSTM model for performance reference.

Simple LSTM Model

We built RNNs as the source model M_{src} and prepared the sequence of vectors using a Long Short-Term Memory (LSTM) [144] layer to perform activity recognition using 3-axis acceleration sensor data available in the smartphone application as the direct input. An LSTM network is a developed RNN to solve input/output weight conflicts and avoid the vanishing gradient problem [64]. The key design of an LSTM network is to produce ways where the gradient can flow for long durations so that the time scale of combination can be modified dynamically based on the input sequence. Hence, this network has been observed remarkably successful in various activity recognition applications. Figure 5.2 shows our LSTM model architecture. We created RNNs. The 3-axis acceleration and gyroscope data

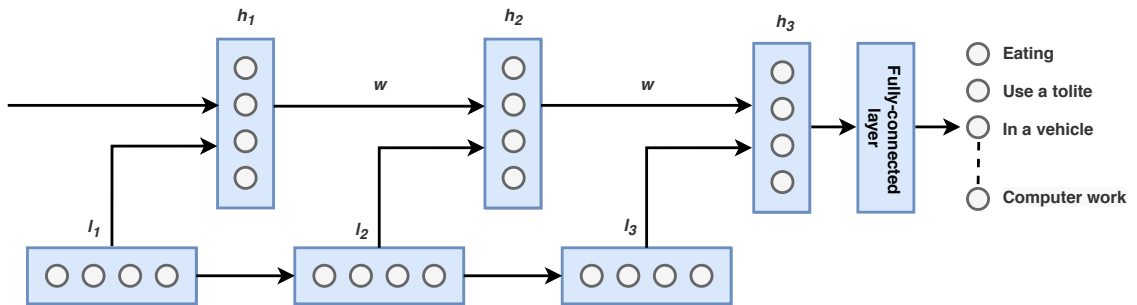


Figure 5.2 LSTM model for activity classification, where l is the input for each layer.

of each time corresponded to the dimensional input layer's size. The number of activity classes corresponded to the dimensional output layer's size. Each unit of each internal layer was an LSTM unit. We preprocessed the input signals since deep neural networks can learn to represent data directly from time-series data. We performed segmentation on the signals into fixed-size windows with 512 samples with a 1-second overlap. Instead of reading raw data immediately, we manually extracted valuable data from the raw sensor data. For each axis, the average and maximum and minimum values were selected as features. In sum, one representation of data had $512 \text{ time-steps} \times 18 \text{ features}$, or 9216 elements. A Rectified Linear Unit (ReLU) defined the activation function of whole layers, excluding the last fully-connected layer. A softmax function and a cross-entropy function

defined the output layer’s activation function and the error function. We set M_{src} holding a stacked-LSTM network that consists of two LSTM layers. This method potentially provides the hidden state at each level to perform at different timescales. They were followed by a dropout layer dedicated to reducing the model’s overfitting to the training data. The hidden layer dimension was assigned to 100. The neural network’s weight was learned using Adam [73] by setting cross-entropy as the loss function. The network was optimized by a batch size of 64 for a maximum of 15 epochs and a learning rate of 0.0001. Lastly, a fully-connected layer was adopted to describe the LSTM hidden layer’s features before a terminal output layer was employed to make predictions. The model’s output was a twelve-element vector including the probability of a given window belonging to each of the twelve activity classes.

CNN-LSTM Model

Convolutional layers can extract valuable knowledge and discover time-series data’s internal representation, while LSTM networks efficiently recognize short-term and long-term dependencies. Our proposed CNN-LSTM model’s approach is to consolidate the benefits of these deep learning techniques efficiently to achieve a remarkably accurate classification. To this end, we designed the CNN-LSTM architecture, consisting of two main components: the CNN architecture for feature extraction and the LSTM architecture for reading the features across time steps. Figure 5.3 shows our CNN-LSTM model architecture. We set

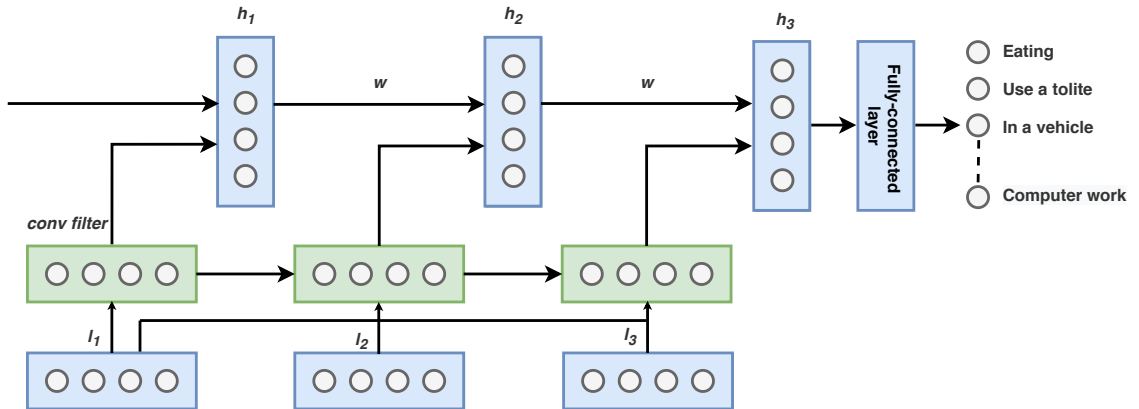


Figure 5.3 CNN-LSTM model. The input is first convolved, and fed to LSTM part.

the number of output, features, and window size using a similar parameter of the simple-LSTM model. We created the LSTM-CNN model that reads subsequences of the main sequence as blocks and selected features from an individual block, enabling the LSTM to understand the features extracted from each block. We divided each window of 512-time steps into four subsequences for the CNN model. As a result, the CNN model was defined to read in sequences with a length of 32-time steps and 18 features. We designed M_{src} as having two consecutive CNN layers followed by dropout and a max-pooling layer. The whole CNN model was wrapped in a Time Distributed (TimeDistributed layer class of

Keras API; this wrapper allows us to apply a layer to every temporal slice of an input) layer to enable the same CNN model to read in each of the four subsequences in the window. The extracted features were then flattened and provided to the LSTM model to read, removing its features before a final mapping to activity was constructed. The number of filters was set to 32, and kernel size was set to 3. Similar to the simple-LSTM model, the ReLU was used as an activation function for the CNN layer. The fully connected layer beside the softmax activation function was employed to classify the activity. The network was optimized with a learning rate of 0.0001 and a batch size of 64 for a maximum of 25 epochs. The weight of the neural network was learned using Adam by setting cross-entropy as the loss function.

The simple-LSTM and CNN-LSTM model were implemented in Python using Keras v2.4.0 (<https://keras.io/>) with TensorFlow Core v2.0.0-rc0 (https://www.tensorflow.org/versions/r2.0/api_docs/python/tf). Then, it was converted to work with TensorFlow Lite (<https://www.tensorflow.org/lite>) and was ready to use in our application. Model training was run with Tesla K80 GPU in Google Colab (<https://colab.research.google.com/notebooks/gpu.ipynb>).

5.4.4 Classification Performance

We carried out an analysis to quantify the performance of M_{src} to measure its generality before giving it to on-device. With the data prepared, we built a training and test dataset. The datasets contained different users to evaluate the robustness of the classifier to new users. We adopted the training dataset to build and validate the model and treated the test dataset as the unseen new data as if the model was in production. We used 80% for training and the remaining 20% of the data for validation. We used F-measure as a metric of accuracy.

Figure 5.4a presents the learning curves of recognition accuracy and loss by F-measure of the training and validation datasets over training epochs for the simple-LSTM model. The final epoch results show that the validation accuracy reached over 0.975 at the expense of only 0.075 validation loss. The test accuracy achieved an F-measure of 98.27%. Contrarily, Figure 5.4b presents the learning curves of recognition accuracy and loss by F-measure of the training and validation datasets over training epochs for the CNN-LSTM model. The final epoch results show the validation accuracy reached over 0.988 at a validation loss of only 0.046. The test accuracy achieved an F-measure of 98.78%. As a result, we can see that both models consistently perform well on the problem of accuracy, achieving an accuracy of about 98%. Overall, the results indicate that the recognition accuracy of the CNN-LSTM model was slightly higher than the simple-LSTM, with a difference of only 0.51% in F-measure for test accuracy. Additionally, Figure 5.4c summarizes each classifier's performance on a set of test data using a confusion matrix with normalization by class to support the size of training for the simple-LSTM and CNN-LSTM model. Both

matrices demonstrated better overall performance and could identify the movement type on a smartphone correctly. Note that we show one confusion matrix since the matrix results for the simple-LSTM are similar to that of the CNN-LSTM model.

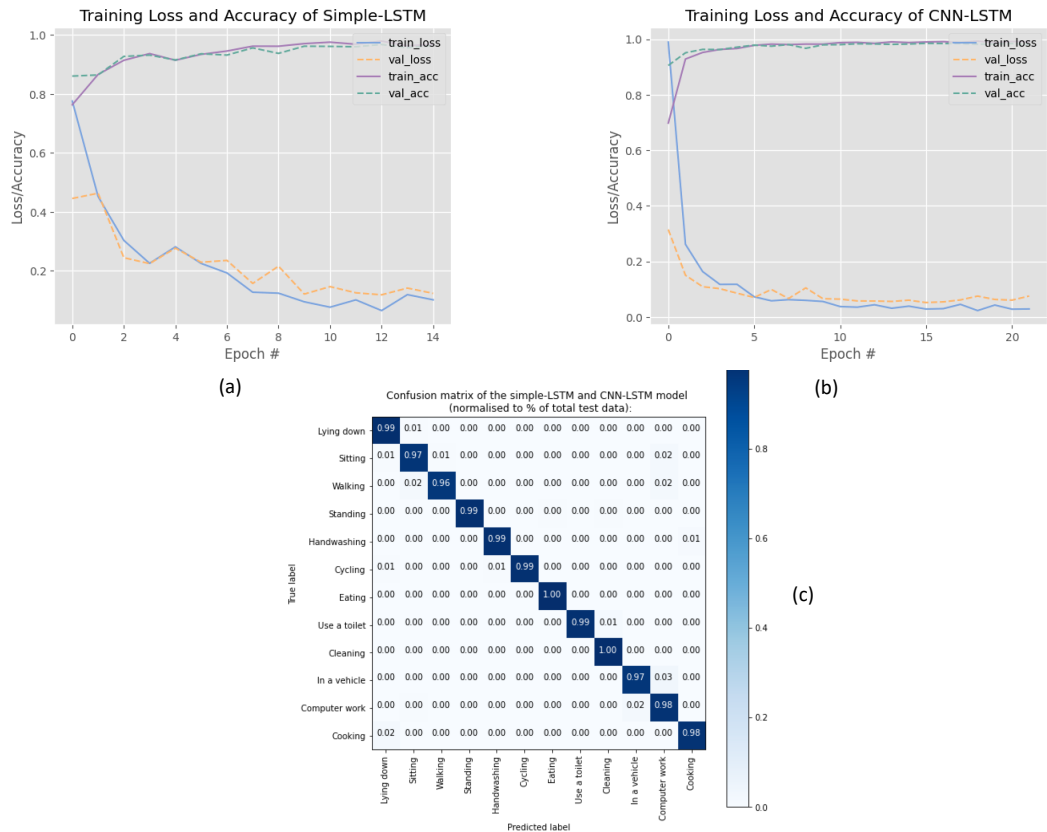


Figure 5.4 (a) A plot of accuracy and loss of the simple-LSTM model; (b) A plot of accuracy and loss of the CNN-LSTM model; (c) Normalized confusion matrix for the simple-LSTM and CNN-LSTM model.

5.4.5 Performance on a Smartphone

In real-world use, the training and inference time must be fast because our application requires immediate feedback to present to users who perform data labeling. The turned feedback should be personalized and given immediately after the task is completed. In this process, data labeling is more efficient because users' mistakes can be corrected more quickly. Thus, we estimated the inference and training time on the smartphone. Additionally, we assume the smartphone's resource usage such as battery damage, CPU, and memory usage is high. In that situation, it cannot be satisfactory for commercial service if its inference and training time is quick. Consequently, we examined the resources managed as inference and training performance on the smartphone.

We used Huawei P10 (Android 9.0, EMUI 9.1) for reference. The smartphone usage log was stored in the Android database. Each inference was performed at an interval of 5 min,

and the total number of executions was 10 if there is no detected change in user activity. Contrarily, if there is a detected change in user activity, the inference was performed immediately. Each training was performed at an interval of 15 min, and the total number of executions was 10. Note that the standard training time depended on several factors, such as the difficulty and complexity of models, the number of samples and parameters, and the task’s design. However, typically, the model can be trained from a few seconds to a few minutes. Our analysis trained the model until the validation loss decreased well, as expected. We estimated the time for preprocessing (feature generation), training time, and inference time using a machine learning model. The average preprocessing time was 0.054 s. Table 5.2 presents the mean inference and training time for each model. Because the simple-LSTM model is more simple than the CNN-LSTM model, the inference and training time of the simple-LSTM model was shorter than for the CNN-LSTM model. The training time was around 54 s and 126 s for the simple-LSTM model and the CNN-LSTM model respectively. The inference time was around 0.0106 s and 0.3941 s for the simple-LSTM model and the CNN-LSTM model. Consequently, the simple-LSTM model is acceptable in real-world applications, compared to the CNN-LSTM model. We

Table 5.2 Measurement results of inference and training time.

Model	Inference Time (second)	Training Time (second)
Simple-LSTM	0.0106	54
CNN-LSTM	0.3941	126

estimated the resource usage concerning battery consumption, CPU, and memory usage of the simple-LSTM for reference. Table 5.3 presents the estimation results. The full battery of the Huawei P10 is 3200 milliampere-hour (mAh). The average battery consumption for each inference was 0.02300 mAh. If our application uses 10% of the total battery, the total execution number is $3200 \times 0.1 / 0.02300 = 13,913.04$. Hence, if the inference is executed every 60 s, we can use the smartphone for $13,913.04 \times 60 = 834,782.4$ s = 231.884 h. The average battery consumption for each training was 0.05100 mAh. If our application uses 10% of the total battery, the total execution number is $3200 \times 0.1 / 0.05100 = 6274.50$. Hence, if the training is performed every 60 s, we can use the smartphone for $6274.50 \times 60 = 6189.3$ s = 104.575 h. The average CPU usage was 5.53%, and the average memory usage was 1.03 megabytes (MB) for model inference. The average CPU usage was 22.20%, and the average memory usage was 1752.45 MB for model training. Note that we estimated the performance when only our application was performed. Consequently, a variation of the corresponding performance in real-world practice is reasonable. Still, our results indicate that resource usage is inexpensive. In summary, the simple-LSTM model was much faster than the CNN-LSTM model, regarding the inference and training time. Moreover, the smartphone’s resource usage of the simple-LSTM model, such as battery

Table5.3 Measurement results of inference and training time.

Condition	Battery Consumption (%/times)	CPU Usage Rate (%)	Memory Usage (MB)
Inference	0.7	5.53	1.03
Training	1.6	22.20	1752.45

consumption, CPU, and memory usage, is inexpensive and acceptable in real-world use. Consequently, we mainly considered the simple-LSTM model for model optimization and evaluation, as described in the following subsections.

5.4.6 Performance Optimization with Model Pruning

Deep learning model inference can be considerably computation-intensive for mobile devices, even for small input data. This section describes a model pruning technique to reduce such computation overhead, delivering the proposed system feasibly on mobile devices. Model compression is an advised approach to decrease the model size and inference computations [28]. The proposed system attempts to apply the conventional compression algorithm to minimize the complexity of optimizing on-device deep learning inference. Various optimizations have been proposed to reduce complex layers, such as pruning [173], quantization [69], and clustering [56]. We selected the magnitude-based weight pruning that performs well on mobile devices based on a collection of experiments. Figure 5.5 overviews the compression pipeline of a weight pruning technique. Magnitude-based weight pruning works by extracting parameters within a model that have only an insignificant impact on its predictions. Pruning gradually diminishes the number of nonzero-valued parameters in the model throughout the training process to obtain model sparsity in a deep neural network’s different connection matrices. Thereby, sparse models are sufficient at compressing, and we can ignore the zeroes during inference for latency enhancements.

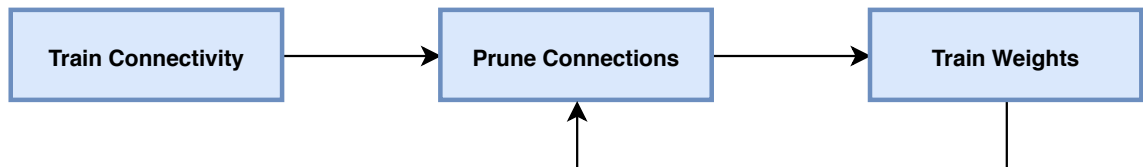


Figure5.5 An overview of weight pruning. The compression processes the original network by pruning synapses and neurons and sharing weights back to prune connections to eliminate redundant connections to make fewer weights in its model, resulting in a minimal loss in accuracy with a 10× reduction in model size.

This study extends the TensorFlow framework to prune the network’s connections

throughout training for the simple-LSTM. We followed a gradual pruning algorithm utilized in [173] in which sparsity is grown from an initial sparsity state s_i to a final sparsity state s_f during n pruning steps, beginning at training step t_0 and with pruning frequency Δt :

$$s_t = s_f + (s_i - s_f) \left(1 - \frac{t - t_0}{n\Delta t}\right)^3 \quad \text{for } t \in \{t_0, t_0 + \Delta t, \dots, t_0 + n\Delta t\} \quad (5.1)$$

The paired weight masks are updated each Δt steps as the network is trained to continuously enhance the network’s sparsity while allowing the network training steps to retrieve from any loss in accuracy after pruning. In our experiment, we started the model with 50% s_i (50% zeros in weights) and end with 80% s_f . Once the model reaches the target sparsity s_f , the weight masks are no longer updated. We computed the end step to finish pruning after 15 epochs. The network was optimized with a learning rate of 0.0001 and a batch size of 64. We split 10% of the training set for the validation set. We applied pruning to the whole model and see this in the model summary. Additionally, we created a helper function to compress the models via a standard compression algorithm using gzip (gzip is a file format and a software application used for file compression and decompression) and measured the zipped size after pruning.

As a result, there was a minimal loss in test accuracy after pruning compared to the baseline. Table 5.4 shows the baseline test accuracy and pruned test accuracy of our simple-LSTM model. We observed that by fully pruning a model with 80% sparsity, the pruned accuracy achieved the closest performance to the baseline accuracy with a difference of approximately 0.18% in test accuracy (an accuracy of 98.27% and 98.09% for the baseline accuracy and the pruned accuracy, respectively). On the other hand, the model size was significantly decreased up to 327,212.00 bytes from pruning. The model size was 520,224.00 bytes and 193,012.00 bytes for the gzipped baseline and gzipped pruned model, respectively.

Table 5.4 Loss in test accuracy and a smaller model after pruning, compared to the baseline.

Condition	Test Accuracy (%)	Model Size (byte)
Baseline test accuracy	98	520,224
Pruned test accuracy	98	193,012

5.5 Systems Implementation

In this section, we describe the system implementation and study design to evaluate the differences between the two conditions in Table 5.1. The simplified input–process–output model, including data labeling, model training, and model inference for our proposed system, is summarized in Figure 5.6. The algorithm’s key component concerning the design of returning personalized feedback using on-device personalization is found in **line 27**. Note

that each process is independent and can run simultaneously. In the following subsections, we detail the design rationale of each process.

Algorithm 1: The simplified input–process–output model for the proposed system.

```

Input : activity classification model with learned weights  $M$ , sensor data  $S$ ,
          activity labels  $l$ , input size  $I$ , batch size  $B$ 
Output: personalized estimated activity feedback  $O$ 
1 sensorBuffer  $\leftarrow$  allocateMinutesLongBuffer()
2 repeat
3    $S \leftarrow$  sensorReading()
4   sensorBuffer.insert( $S$ )
5   if  $S \geq I$  then
6     processInput(sensorBuffer)
7   while  $i \leq I$  do
8     inputSignal.insert(sensorBuffer)
9     /* Adds a new sample for training. */
10    if  $Mode.dataLabeling$  then
11       $M.insert(inputSignal, l)$ 
12      classInstance += 1
13    /* Fine-tunes the model on the previously added data samples. */
14    if  $Mode.training$  then
15      if classInstance  $\geq B$  then
16        while isTraining do
17          // Gradient-based update using Adam
18           $M.enableTraining((epoch, loss) -i \{$ 
19             $M.evaluate()$ 
20             $M.save()$ 
21             $M.disableTraining()$ 
22          })
23    /* Runs model inference on a given data. */
24    if  $Mode.inference$  then
25       $M.get()$ 
26      predictions  $\leftarrow M.predict(inputSignal)$ 
27      for  $i = 0$  to predictions.length do
28        if predictions[ $i$ ].getConfidence()  $\geq$  max then
29          idx  $\leftarrow i$ 
30          max  $\leftarrow$  predictions[ $i$ ].getConfidence()
31      // Returns feedback
32       $O \leftarrow$  predictions[idx].getClassName()
33    return  $O$ 
34  sensorBuffer.empty()
35 until exit;

```

Figure 5.6 The simplified input–process–output model for the proposed system.

To recognize activities on the device with fine-tuning, we need to collect supervised information on sensor data activities. We implemented the FahLog (<https://play.google.com/store/apps/details?id=jp.sozolab.fahlog>): an Android application, written in Java with AndroidX (AndroidX is a major improvement to the original Android Support Library), which is an improvement of [101]. This application can be used for the generated models in the previous section for data collection, fine-tune training, and inference. Furthermore, we implemented a cloud server (<https://fahact.sozolab.jp/>), which is an improvement of [98]. It enables us to manage large-scale data from the participants and use them for evaluations. In this work, we focus on implementing the application with the required functionality for performing the proposed system. For a detailed review of the cloud-server implementation, we refer an interested reader to the abovementioned paper. Software requirement analysis of the application includes the following:

- To efficiently collect smartphone sensor data and activity labels from user’s input for activity recognition.
- To automatically fine-tune the pre-trained model with small data on individual devices.
- To deliver estimated activities gained from on-device personalization as real-time feedback through notifications.
- To support offline-first to ensure that the application functionality is unaffected by intermittent lack of a network connection.

We itemize the requirement analysis resolution and software design as follows:

- **Data labeling and smartphone sensors:** Activities are temporal data with a specific duration; it is crucial to record both the start time and the end time. For this reason, we provided the labeling screen (Figure 5.7), which enables a user to perform activity data labeling tasks. We detailed a written guide and associated images of the application in a user manual (<https://github.com/nattafahm/supporting-materials-sensors20/blob/master/user-manual-fahlog.pdf>). The application can automatically collect smartphone sensors available on the mobile device. The sampling frequency is set at a 20-Hz, which is the standard and lowest setting. Since the participants in this study are using their smartphones, we cannot drain their battery. This configuration helped us optimize the sensing process to coordinate data generated and battery consumption, even if it had less frequent sensor readings.
- **Model fine-tuning:** Data instances keep adding their corresponding class IDs to the model cache if the data labeling is performed. Once training data is ready for use, it can be loaded into mini-batches, and the training can be initiated. In this state, data will not be immediately used for training. Instead, it will be buffered and used when the input samples’ size reaches a pre-defined batch size of the on-device model. Fine-tuning is automatically executed only every 15 min to avoid heavy computational workloads.

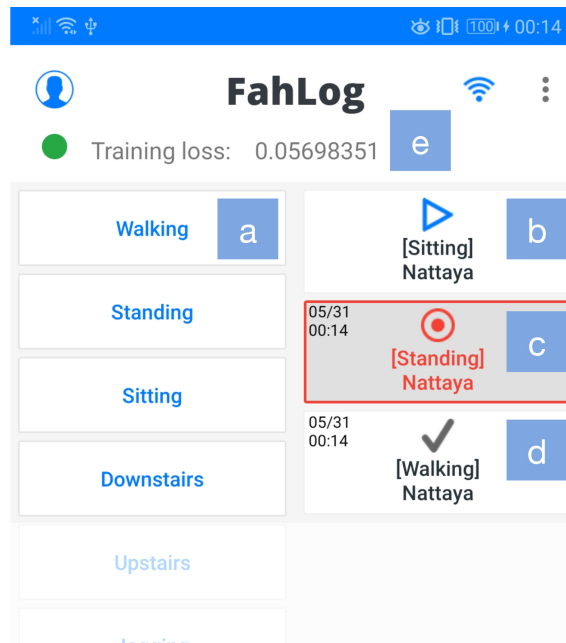


Figure 5.7 Data labeling screen.

Since the training is a simple indicator of model quality, it does not catch overfitting problems. We divided the dataset into development and test datasets and split 10% of the development set for the validation set. We then computed the loss over the validation set to ensure the model is learning what we want it to learn. Training is stopped when the validation accuracy no longer improves; the updated model overwrites the previous model. Only in this case, does it reach an accuracy percentage of over 70%. During the training process, the model is trained for a few minutes or seconds until loss decreases. The updated model is then used for inference before the next training is activated. We added functionality to show the training execution, as shown in Figure 5.7e. The \odot symbol is green if training is running; otherwise, it is gray. The loss values in the panel can be observed fluctuating as the network is trained.

- **Model inference:** We reused the saved model stored in the internal device for the inference process by considering the estimates' confidence bands. We observed the output probability of each class in a real-time manner. However, to prevent excessive interruptibility, the application stops notifying if the current activity is notified once. It resumes after 5 min or reports immediately if it detects changes in the user's action (e.g., users in the transition from "activity a" to "activity b"). By default, all sounds and vibrations are turned on and set as a high-priority notification to ensure that the application's notifications are notified to the user's smartphone. Figure 5.8 shows an example of estimated activities on a smartphone notification.
- **Offline first:** With an offline-first approach, data are written locally on the end user's device in the JSON format for model training and periodically uploaded to the cloud when the smartphone is connected via WiFi or mobile data for evaluations. Sensor

data and activity labels are uploaded to the server by the HTTPS protocol immediately if the on-device training is successfully executed to free up space on the device due to resource constraints. Additionally, data will be deleted from the phone's internal memory when the transmission is complete. This approach ensures that the application's core functionality will still work in the absence of a reliable network connection.

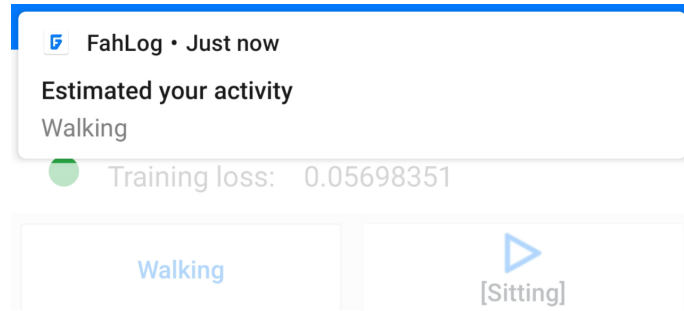


Figure 5.8 Notification displaying estimated activities.

5.6 Experiments

To verify the proper function of the protocol and data collection process and to assess the effect of the proposed method on data labeling, we performed a verification experiment. We recruited 8 volunteers who are students or alumni of a university in Thailand via social recruiting. Our post's objective directed participants to perform an activity labeling task for four days using the provided smartphone application. Participants were required to own an Android-based smartphone with at least 5.0 or more API levels. The device was placed in a trouser's pocket freely selected by the subject in any random orientation to simulate every phone usage. We employed a within-subject design in which all participants were exposed to every condition to help reduce errors associated with individual differences. Half of the participants were assigned to the proposed condition before they were assigned to the baseline condition. In contrast, the other half were assigned to the baseline condition before they were assigned to the proposed condition. They were asked to assign activities from the classes predefined in Figure 5.9 and spend 8 h per day at least (2 days per condition) on the application. The design choices and related user interface are detailed in Table 5.1.

Additionally, we requested participants to complete a pre-study questionnaire, focusing on demographic information and smartphone usage. We controlled for this variable by balancing participants across the two experimental conditions based on their response to minimize the learning effects across conditions. The study was conducted in early July 2020. Eight people (4 female, 4 male) between the ages of 24 and 27 years old participated in the study. A Welch's unequal variances t-test indicated no significant difference between conditions ($t = 0.65465$, $df = 5.069$, $p = 0.5412$).

5.7 Activity Recognition: Evaluation and Results

This section evaluates the proposed system in depth to verify whether it can improve data labeling. We applied the simple-LSTM algorithm using the labels and sensor data collected in Section 5.6 for activity recognition and compared the recognition accuracy results between two conditions using the F-measure. We followed a standard activity recognition chain using a supervised learning approach—data preprocessing, segmentation, feature extraction, training, and testing. The following research questions have been defined for this study:

- **RQ1:** Can the proposed system improve data labeling in each user?
- **RQ2:** Can the proposed system improve data labeling in each activity class?

5.7.1 Data Preprocessing

We accumulated three-dimensional periodic data that incorporate acceleration and gyroscope sensors on the smartphone, recording data every $1/20$ s. The axes' norm for each row dropping in the time slot was computed to aggregate the data. Therefore, discrepancies originating from various smartphone positions/orientations at the time of the reading decreased. We later combined the periodic sensor data and activity labels without time synchronization because both are positioned on the same device. Because deep neural networks are excellent at learning representations of data directly from time-series data, we only had to perform minimal preprocessing of the input signals for the system to work properly. The data kept only the activities that correspond to each subject to avoid any unexpected or invalid activity data from affecting results. The data were then linearly interpolated to account for missing data in some of the rows. We also discarded the first and last 10 s of each activity instance for each user to account for possible transient data that were incorrectly labeled as found in practice.

Next, we transformed the raw time-series data into examples. The resulting dataset after cleanup is quite unwieldy, and it is challenging to perform a feasible analysis directly. Consequently, we segmented the data using a sliding window of 5.12 s, which has been found to be an appropriate window of time to capture movement sequences. We then applied a 1 s displacement between consecutive windows and manually useful features from the raw sensor data to create a predictive model. For the accelerometer and gyroscope data, the average, maximum, and minimum values were extracted as features for each device's axis. We also included the participants' IDs for user-dependent training, as described in the next section. In total, one sample of data has (512 time-steps \times 19 features), or 9728 elements. The whole dataset is composed of 16,819 activity windows, each labeled with their corresponding activity id. Figure 5.9 shows the distributions of collected data.

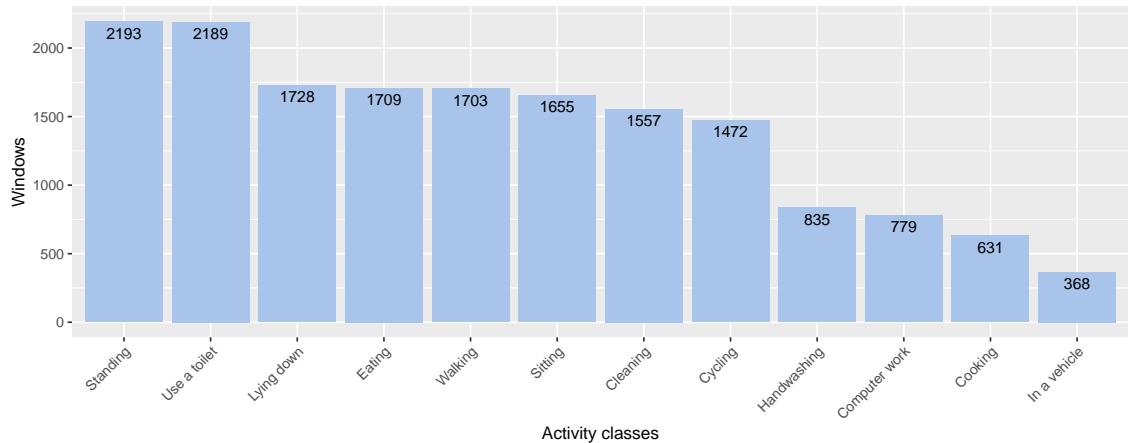


Figure 5.9 The distributions of collected data.

5.7.2 Evaluation Method

We developed and evaluated neural network models for multi-class classification problems. For the training algorithm, we divided the dataset into training and test sets. We used the training dataset to build and validate the model and treated the test dataset as the unseen new data. We used 20–30% of each user’s data from the beginning of the time-series and applied it for testing, and the next parts for training and validation. The training set users’ data was split into 80% for training the model and 20% for validation and hyper-parameter tuning.

Rather than applying the model to new users by comparing it with other users’ labels, we focused on the accuracy of human contributions in each condition (e.g., personal context and activities to be used by the user themselves) by comparing it with the machine’s knowledge. Hence, we applied user-dependent training to show accuracy improvements for each participant in each condition without considering side effects such as different sensor positions. We utilized the F-measure as a metric of accuracy. However, the real data are highly imbalanced, as shown in Figure 5.9. To address this issue, we handled imbalanced classes with upsampling using the SMOTE algorithm [24] by oversampling only on the training data; none of the information in the validation data was used to create synthetic observations to make them generalizable. We then utilized the F-measure after resampling to avoid the adverse effects of class imbalances to focus on true positive samples.

The models were trained using our simple-LSTM algorithm, as described in Section 5.4.3. Here, we utilized the same model configuration and window size based on an earlier investigation to keep experimental evaluation unbiased due to this hyper-parameters effect. Since neural networks are stochastic, while it gives the model its adaptive ability, it is impossible to assess the model’s skill from a single evaluation. To do so, we did a slightly more detailed assessment of the model. We repeated the model’s

evaluation a total of 10 times, then summarized the model’s performance across each of those runs. Additionally, we applied early-stopping during training to avoid over-fitting if the network fully converged on the training set.

5.7.3 Results

From the abovementioned research questions, we present the activity recognition accuracy results by F-measure of test data with user-independent training for two conditions from the viewpoints of (RQ1) activity recognition accuracy improvements in each user; (RQ2) activity recognition accuracy improvements in each activity class.

RQ1: Recognition Accuracy Improvements in Each User

Figure 5.10 shows the activity recognition accuracy by F-measure of user-dependent training for the test data. Overall, the data indicate that all participants’ recognition accuracy in the proposed condition was improved—the average recognition accuracy increased from 82% to 90% (+16%). When looking at the performance of individual users, we observed the use of the proposed method increased the average recognition accuracy of F-measure by +3% (from 84% to 87%) to +24% (from 80% to 56%). All participants in the proposed condition had improved recognition accuracy, sorting by descending order as follows: The participant ID (PID) 103 had recognition accuracy improvement of +24% in the F-measure.

Figures 5.11 and 5.12 summarizes the performance of each participant’s classifier on a set of test data using a confusion matrix with non-normalization of user-dependent training for the proposed and baseline condition, respectively. As a result, the proposed matrices were quite thick and demonstrated the overall results’ high accuracy score. In contrast, the baseline matrices were relatively sparse and explained the overall results’ low accuracy score.

RQ2: Recognition Accuracy Improvements in Each Activity Class

Figure 5.13 shows the activity recognition accuracy by F-measure of each activity for the test data. Overall, the data indicate that all activities’ recognition accuracy in the proposed condition was higher than the baseline. Regarding the test data’s activity recognition accuracy with user-dependent training, we observed that the proposed condition had the highest recognition accuracy improvement of +28% of the F-measure in the “walking” class. The proposed condition had the next-highest recognition accuracy improvement of +23% in the F-measure in the “handwashing” class, followed with the improvement of +19% in F-measure in the “in a vehicle” and “standing” class. The remaining activities had reasonable improvement of recognition accuracy in the proposed condition as follows: the “cooking” and “eating” class had a recognition accuracy improvement of +18% in the F-measure; the “cleaning” class had a recognition accuracy improvement of +15% in the F-measure; the “computer work” class had a recognition accuracy improvement of

+13% in the F-measure; the “use a toilet” class had a recognition accuracy improvement of +12% in the F-measure; the “lying down” class had a recognition accuracy improvement of +7% in the F-measure; the “cycling” class had a recognition accuracy improvement of +6% in the F-measure; the “sitting” class had a recognition accuracy improvement of +4% in the F-measure.

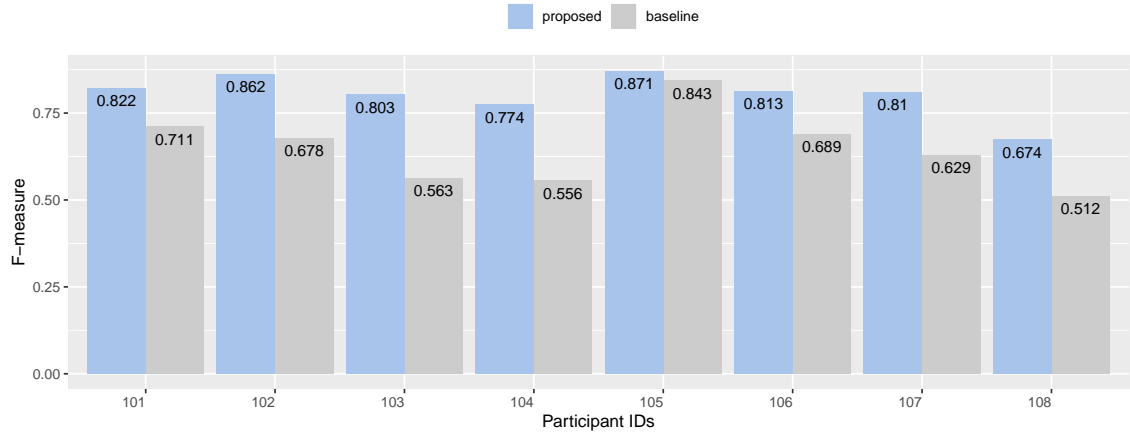


Figure 5.10 Recognition accuracy improvements in F-measure in each user.

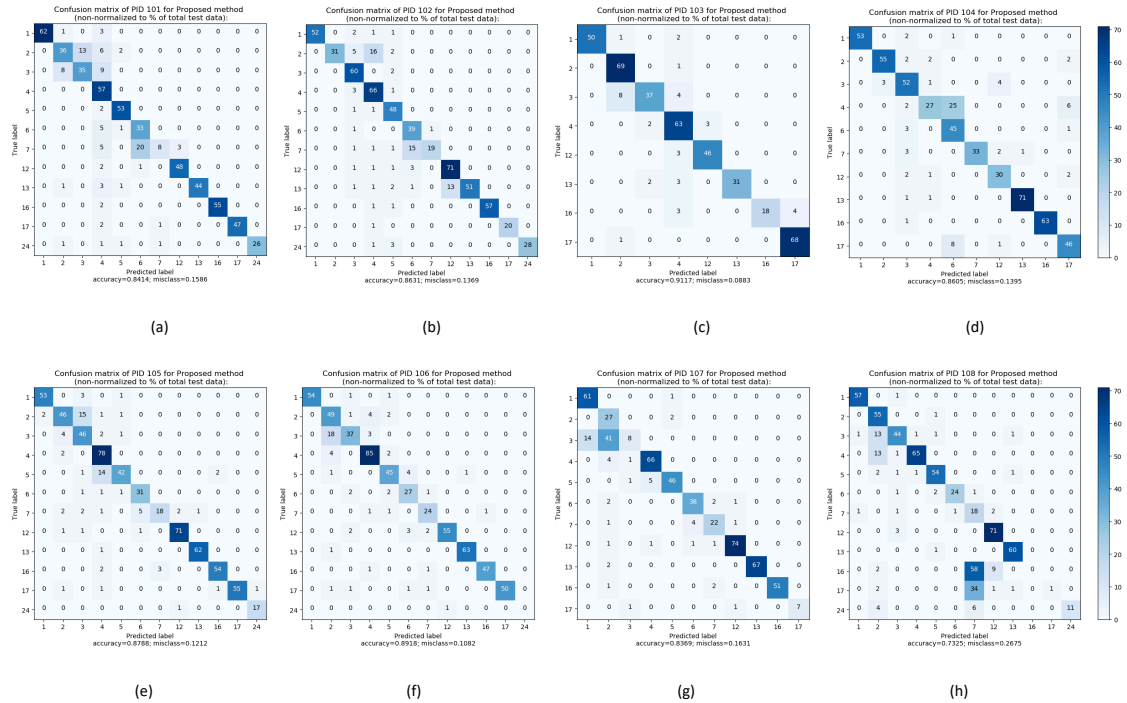


Figure 5.11 Non-normalized confusion matrices of each user for the proposed condition.

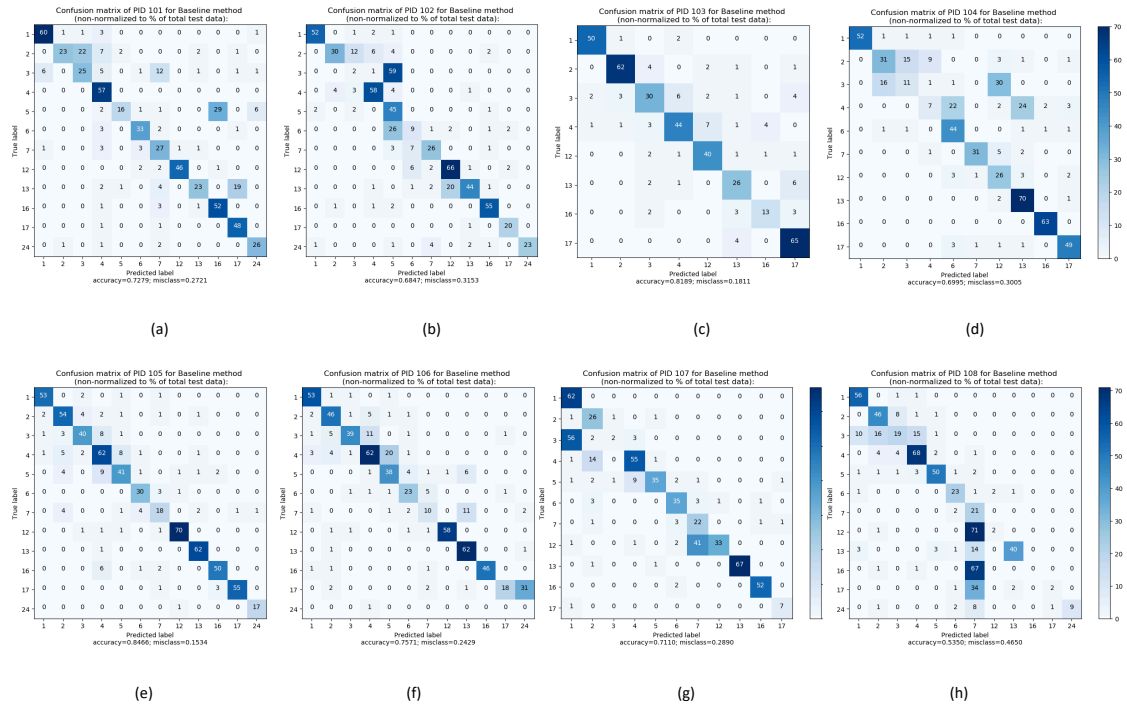


Figure 5.12 Non-normalized confusion matrices of each user for the baseline condition.

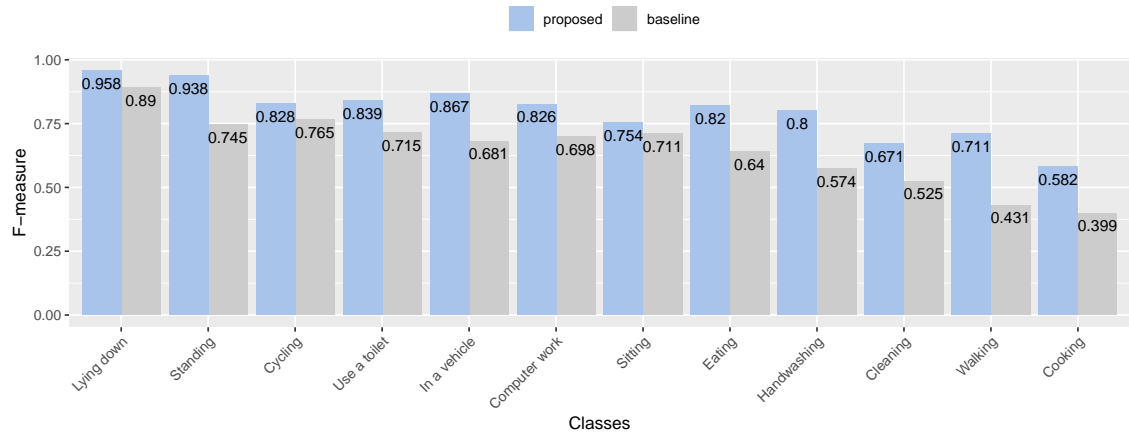


Figure 5.13 Recognition accuracy improvements in F-measure in each activity class.

5.8 Discussion and Future Directions

In this study, we introduced a method for activity data collection utilizing on-device personalization. Although our user research is carried out on a moderate scale and for a short-term duration, the trial results have already given promising evidence that RQ1 and RQ2 were fully supported. According to the current investigation of on-device machine learning inference [63] and the official web page of TensorFlow Lite, the current utilization mainly concentrates on imaging classification, object detection, speed recognition,

and natural language processing such as text classification, question answering, and smart reply. Contrarily, this research presents the application of activity recognition. We are confident that our study opens the door to an innovative application domain for on-device machine learning. Although the results are promising, there are still some weaknesses in our system. We outline remarkable limitations and discuss them below.

We assumed that the application is static or resource available for a given algorithm. However, budget resources for a specific application at runtime are not adjusted based on a predetermined estimate and can be dynamic on mobile operating systems, i.e., software platforms [40]. Thus, there is a need for research on algorithms incorporating resource-accuracy trade-off under a dynamic resource budget to choose the optimal algorithm that fits resource constraints. For instance, applying a greedy heuristic algorithm [16] to make the locally optimal choice at each stage with the intent of finding the best models or hyperparameters for multiple applications at runtime to maximize their performance jointly. This investigation can be explored in future work.

We predefined activity classes containing a fixed number of <activity, id> pairs. If the action that a user wants to input is out of the predefined list, it cannot be correctly predicted. Following prior work [65], the customizable activity class function developed is designed to be performed on the cloud and dynamically customized depending on the site server (e.g., an experimental group/facility) rather than individual users. The weak support of personalization can have a significant impact on model performance. Consequently, a customizable activity class function via the smartphone application remains to be carefully developed. However, the trade-off is the difficulty and complexity of the model design, which should be carefully considered. For example, suppose if the number of classes can change at runtime, we already need to thoroughly consider when we design the neural net's architecture and make its classification layer large enough.

The use of transfer learning may reduce the need for massive labeled data. However, the model's quality can be compromised if the device's acquired data is still insufficient, such as overfitting. Several preprocessing techniques can be considered to overcome when data are sparse, such as data augmentation [125]. Data augmentation is commonly used in deep learning, where the sample size is critical for model generalization. This process stimulates new data instances that maintain the correct labels to increase the sample size when limited labeled data are available. Data augmentation usually relies on linear transformations in the spatial domain and has mainly been implemented for image recognition. However, label-preserving augmentation for time-series is much more challenging since any transformation is complicated to determine without profound domain knowledge. We are confident that the impact of data augmentation on the performance deep neuron network will introduce new challenges to be explored in future research.

Additionally, we utilized on-device fine-tuning for personalization. However, this concept can be generalized to support many other activity recognition applications. Future work should attempt to explore the impact of generalization and the tradeoffs therein.

Similarly, while we employed a specific network for the two networks and achieved good training results, we may lose the optimal information if the parameter and meta parameter values are not appropriately selected. We believe that capturing several different network sizes and drawing conclusions will help achieve the greatest improvement. We intend to investigate this in future work. Further, while the accuracy level of the deployed model is sufficiently high for application to real-world data, the participants might still assign the wrong label if the model has made a few mistakes. Therefore, future research should further examine user errors that occur in such a scenario. For example, providing an accuracy percentage for participants to reduce user errors, but we need to avoid redundant information that may discourage participants. The other remaining limitations and challenges stimulate our future research; for example, we intend to attempt large-scale data collection, explore other types of optimization techniques, and further assess the usability of the proposed method with user studies.

Despite these limitations, we believe that our study is representative of a solution for the lack of accurate labels in data labeling and is an essential first step towards understanding on-device personalization in activity recognition.

Chapter 6

Discussion

This chapter reflects on the research contributions given in Article 1–6 and highlights how these contributions answer the research questions outlined at the start of this thesis. In the preceding chapters’ studies, we focus on both the quality and quantity of participant contributions in adopting smartphones as a research instrument. As researchers rely on these contributions for their study inferences, the collected data must be reliable and high-quality, and participant responses must be collected across various contexts adequately. In addition to considering the quality and quantity of participant responses, the included articles further examine aspects concerning participant motivation. Since mobile activity recognition studies require high effort, participant contributions must be highly motivated and sustained. We provide a summary of our contributions in Table 6.1.

This chapter starts with a discussion on the quality and quantity of participant data. First, we discuss how the quality and quantity of participant contributions can be improved. Following, we look at how methodological designs affect participants’ motivation. Subsequently, we reflect on smartphones’ usage as a research instrument, particularly for data collection in mobile activity recognition studies. Finally, we conclude this chapter with the limitations of the studies included in this thesis and present a road for future work in this domain.

6.1 Response Quality and Quantity

The considered studies’ empirical results show that participant responses’ quality and quantity in mobile activity data collection studies can be improved.

We argue that researchers should aim to improve data collection accuracy if they rely on participant contributions as a source for information to make reliable inferences. Researchers should also propose practical approaches to increase participant responses to gather sufficient data without high-cost and time-consuming for their strong inferences, particularly for a real application. Here, we discuss various methods for improving data quality and quantity in mobile activity recognition studies to answer the first research question:

Table 6.1 Summary of research objectives, experimental setup, and main contributions for Chapter 3-5.

Chapter	Research objectives	Experimental setup	Contributions
Chapter 3	Propose novel gamified active learning and anomaly detection for crowdsourced data labeling.	A total of 120 participants submitted 6,549 labels from 19 activities and engagement data to assess their participation.	Achieve high-quality crowdsourced datasets by overcoming three main issues: (1) lack of accuracy, (2) loss of motivation, and (3) inaccurate data.
Chapter 4	Propose novel on-device deep learning inference instead of cloud-based deep learning inference to alleviate the self-labeling effort .	A total of 6 participants participated in a six-day in-the-wild study. Participants submitted 734 labels from 6 activities.	Establish the possibilities of using on-device inference for efficient activity data collection and identify recommendations for optimizing data collection in future on-device studies.
Chapter 5	Propose a novel on-device personalization using resource-constrained devices and mobile optimization.	A total of 8 participants participated in a four-day in-the-wild study. Participants submitted 16,800 activity windows from 12 activities	Demonstrate the possibilities of on-device personalization in order to deliver relevant information to increase participant accuracy and identify recommendations for mobile optimization.

- **RQ1:** How can researchers improve the quality and number of participant contributions in data collection for mobile activity recognition studies?

In Chapter 3, we proposed a system for robust activity data collection with crowdsourcing. Crowdsourcing can be considered an efficient method or tool that human computation systems can use to distribute tasks through an open call. With crowdsourcing, the prior works evidence that it is efficient, cost-effective, and highly scalable for collecting human-labeled data. However, there are various challenges of crowdsourcing applications that exist today. One of the most critical challenges that we considered in this thesis is the

lack of accurate information. We set out the relevant questions as follows: (1) what task can be performed adequately by machine, consequently decreasing the need for human involvement?; (2) can we take advantage of the complementary abilities of machines to make computation more accurate and efficient?. There are many opportunities to apply machine intelligence to help improve the accuracy of human computation algorithms. To find solutions for those questions, we optimally selected intelligent techniques.

In Chapter 3, we employed “active learning” to reduce the cost of data labeling by human users. The results presented in this chapter indicate that the use of the proposed approach using active learning increased the accuracy of collected data. Similarly, In Chapter 4, we utilized on-device deep learning to make real-time activity recognition on smartphones. The proposed algorithm behind this chapter showed that the data obtained had improvements in both quality and quantity. Furthermore, In Chapter 5, we enhanced the proposed algorithm of Chapter 4 by utilizing on-device personalization to make better inferences, particularly for individuals. The results of this study indicated that the proposed system could as well improve activity accuracy recognition significantly.

Obviously, overall results reflect our systems that leverage machine intelligence as an optimizer tool can achieve better mobile activity data collection results. Crowdsourced data labeling can also be leveraged to benefit the researcher by providing information or solutions faster than traditional means. A summary of discussions in relation to **RQ1** presented in 6.1.

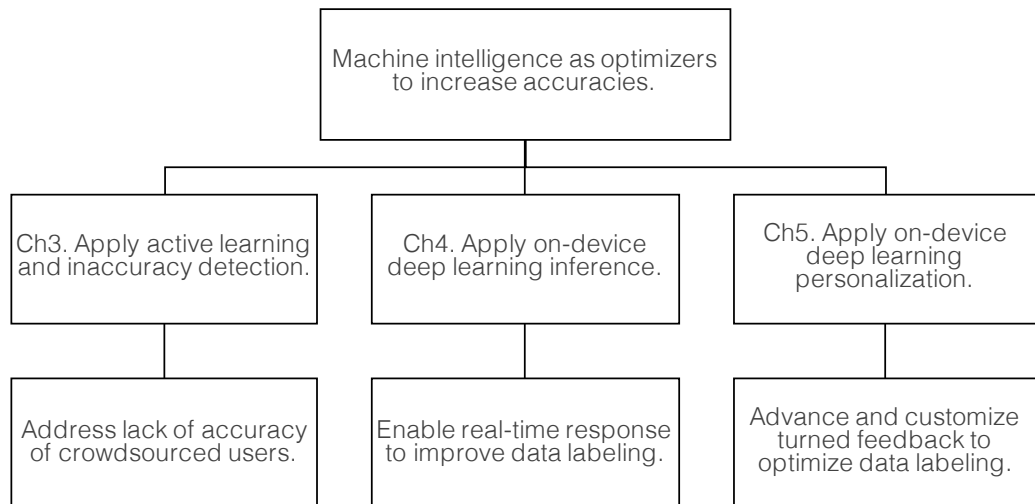


Figure6.1 A summary of discussions in relation to **RQ1**.

6.2 Participant Motivation

Given the nature of data labeling for mobile activity recognition studies, participant motivation is possible to decrease as time progresses and the response burden increases, as evidenced in empirical results. As a result, researchers might have to face a rise in partic-

ipant errors, the challenge of preventing and detecting inaccurate input, and the potential for participant dropout. As human computing systems are built to manage increasingly complex tasks done by increasingly larger crowds (e.g., to generate labeled activity data), we must use machine intelligence to coordinate people, make sense of, and display information to participants. Hence, we present applying machine intelligence pertains to the second question of design:

- **RQ2:** How can researchers motivate participants to participate and carry out data labeling tasks to their best abilities?.

We set out the relevant questions as follows: (1) how do we drive participants to have a long-term interaction with the systems by building an environment that satisfies their specific needs (e.g., to be relaxed and to have a sense of attainment)?; (2) how do we design game mechanisms [154] that encourage participants to report the truth (i.e., to generate accurate output)? In response to the challenges, in Chapter 3, we demonstrated the integration of gamification into active learning to overcome the lack of motivation and sustained engagement. We measured several dimensions of user engagement and received high scores on average. These scores indicate that the participants engaged with their labeling tasks. Moreover, we discovered that other engagement measurements were strongly correlated with activity recognition accuracy. These findings support our research goal; gamification could be used as a motivator to generate accurate output. In Chapter 4 and Chapter 5, we introduced the application of on-device inference to provide feedback to participants to help participants sense labeling performance faster. The mechanisms behind real-time updates provide data that allows participants to make more appropriate and informed decisions and then to take action based on that information. The study results indicate that our systems drive participants to interact with the system and encourage them to refrain from dropping out of a study as the quantity and quality of collected labels were significantly improved. While gamification enhances the quantity and quality of labels, the raw data quality can be compromised if the workers engage in dishonest behavior. To reduce this issue, we presented the inaccuracy detection algorithm. The result analysis indicates that data labeling with gamification can lead to more reliable participants. In other words, when enhancing motivation and sustained engagement with gamification, inaccurate data may also be reduced. A summary of discussions in relation to **RQ2** presented in 6.2.

6.3 Smartphones as a Research Instrument

The widespread availability of smartphones has taken exciting possibilities for researchers interested in studying humans in-the-wild. This technology has resulted in the increased use of smartphones as data collection tools for human activity recognition studies. Simultaneously, researchers have mentioned several concerns and challenges in the adoption of

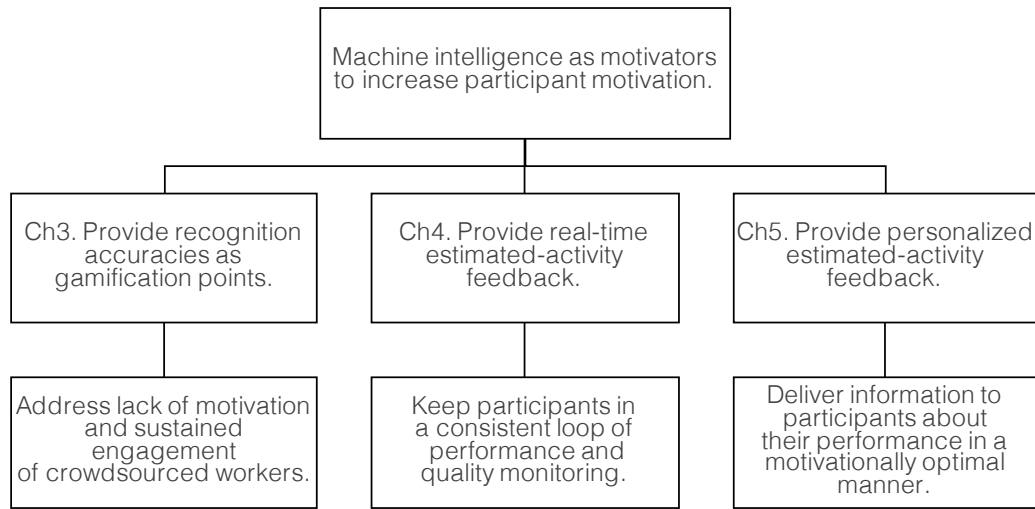


Figure6.2 A summary of discussions in relation to **RQ2**.

smartphones as a research instrument, as presented in Chapter 2. Therefore, answering the third research question:

- **RQ3:** How can researchers employ smartphones as an effective research instrument in data collection for mobile activity recognition studies?

is a crucial step towards increasing smartphone-based data collection efficiency.

In Chapter 3, we employed crowdsourced workers to perform data labeling tasks. We recruited 120 diverse workers and gathered valuable 6,549 activity labels in a shorter time. This collection indicates that the use of participants' smartphones as research tools had unlocked the potential for us for enriched data collection. Although the data was collected using different smartphone models from different vendors contribute to diversity challenges, it supports the generality of our approach and its usability in real-world conditions, which is impossible with paper-based methods. Moreover, using participants' smartphones increases the ecological validity of data contributions as participants carry a familiar device. On the other hand, we could not monitor their battery-powered. Therefore, we optimally set the most appropriate sampling rate of sensor data collection for the smartphone application. Our configuration allowed us to optimize the sensing process to balance data generated and battery consumption, even if this meant having less frequent sensor readings.

In Chapter 4, we exploited the advances in deep learning to detect and model human behaviors automatically on smartphones. However, these algorithms typically require massive amounts of computation for training and inference, making them inappropriate for deploying resource-constrained mobile devices. To keep our application resource-efficient, in Chapter 5, we introduced a list of practices and strategies to improve our model performance. First, we did performance evaluations of implicit smartphones and made a trade-off between model complexity and size. This result analysis provided us to optimally chose

the best model for the task. Second, we optimize our models using the proper technique to build smaller models that are generally faster and more energy-efficient for mobile devices. Our experimental results provided us to reduce the computation cost of on-device learning without affecting the accuracy. Finally, we applied transfer learning from a pre-trained network to address the lack of sufficient training data and minimize the need for training deep learning on mobile devices from scratch. The results implied that our algorithm behind Chapter 5 ushered in deep learning efficiently on resource-constrained mobile devices, making applications seamlessly run across diverse platforms.

In summary, smartphones are valuable scientific instruments for researchers to promptly collect a larger number of high-quality activity labels. However, the trade-off is an increase in challenges of resource-constrained mobile devices should be carefully designed and evaluated. A summary of discussions in relation to **RQ3** presented in 6.3.

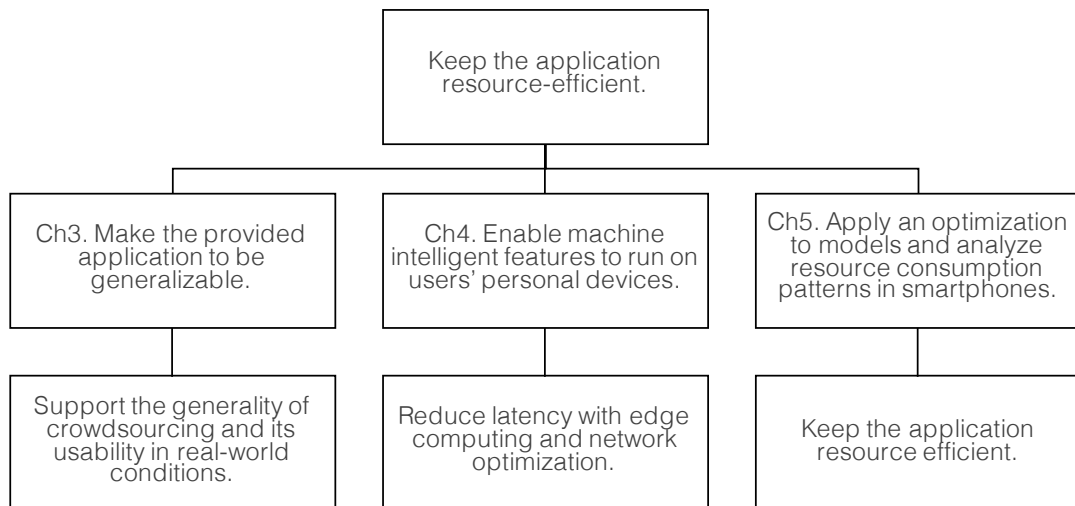


Figure6.3 A summary of discussions in relation to **RQ3**.

6.4 Future Directions

Following the discussion of our study concerning the research questions, we outline potential roads for future research in relation to this thesis topic. As the limitations of individual studies are already discussed in the respective articles, this section pays specific attention to additional certain recommendations and considerations that are the essence of this research and usually span across various studies.

The work introduced in this thesis is a preliminary step in quantifying the effect of methodological configurations on response quality and quantity and participant motivation in mobile activity data collection studies. Given the sheer number of methodological factors to consider in a mobile activity data collection study and the potential impact of these factors, future work is an opportunity to investigate different methodological configurations further. We consider two different directions directed to fulfilling this goal. First,

the most generally used strategy towards quantifying the effect of methodological designs is to analyze a standard study parameter as the subject of a study (e.g., the number of daily data labeling) [30]. This analysis allows deeply investigating an individual parameter to minimize the effect of any confounding factors as best as possible. Second, the individual studies reporting measurements that are expected to have some mistakes. Indeed, researchers cannot observe all error factors affecting response quality and quantity, and participant motivation (e.g., the differences among participants in digital literacy, smartphone usage, and inherent interest in a study). As such, we suggest employing a statistical analysis that combines the results of multiple mobile activity data collection studies. For example, researchers should consider adopting meta-analysis when multiple scientific studies are addressing the same question [94]. In addition to supplying an estimate closest to the common unknown truth, this results analysis can contrast results from various studies, recognize patterns among study results, provide references of disagreement, and identify multiple studies' relationships. Second, *the maintenance of academic software tools for researchers remains lacking*. One of the reasons that cause this issue is the rapid evolution of mobile technology. The technology speed and update rate requires both researchers and developers to maintain their software tools actively. However, there are factors in terms of software maintenance costs for academic research, such as the complexity of the software development, rapidly changing research interests, availability of funding, and most importantly, typically developing an academic software tool based on each study. Consequently, researchers often lack incentives to maintain it. In this thesis, we actively maintained our mobile application and applied it to multiple data collection studies. This continued software tool substantially reduces software development costs as our multiple scientific studies directed to the same proposal. Therefore, we advocate for researchers to actively maintain software tools [49] and support such essential software available and accessible to researchers worldwide [43]. As such, we recommend researchers releasing code as open source. We believe that open-source software is a necessity and in the interest of the broader research community. This enables reproducibility and collaboration, but maintainers need support to scale and build community.

Lastly, *a new paradigm for the future of data labeling exists*. In addition to active learning and transfer learning presented in Chapter 3–5, other machine learning areas exist [11, 172, 174] that are motivated by the demand for enhanced quantity and quality of labeled training data employing several high-level techniques to approach this need (e.g., weak supervision and semi-supervised learning). Based on our findings showing that alternatives to pure crowd-labeling are very cost-effective to acquire labeled data. We have demonstrated how these technological possibilities can be utilized to improve the quality, quantity, and motivation of participant contributions for mobile activity data collection studies. Accordingly, we are delighted to see the new research questions and methods that researchers incorporate those paradigms into their labeling workforces and solutions in the future of mobile activity recognition studies or other relevant areas. While these advanced

technological approaches will be increasingly able to help researchers collect data on study participants automatically and effectively, we argue that human input, observations, and reflections will play a crucial role in data labeling studies.

Chapter 7

Conclusion

This thesis investigates the quality, quantity, and motivation of participant contributions for mobile activity data collection studies. Furthermore, this thesis discusses actionable future steps to improve data collection and present recommendations to employ smartphones as a research instrument efficiently. Below is a summary of each research goal.

First, Chapter 3, we addressed the prevalent issues in crowdsourced data labeling, including lack of accuracy, lack of motivation and sustained engagement, and inaccurate input. The primary purpose of the study was the production of high-quality datasets for activity recognition using ubiquitous crowdsourcing. We evaluated the proposed system and accomplished the three study goals by collecting usage data from 120 diverse workers in a crowdsourcing platform. First, accuracy was substantially improved by using active learning. Second, motivation and sustained engagement were increased with gamification. Finally, inaccurate input was reduced by inaccuracy detection. Furthermore, the positive correlation between activity recognition accuracy and application of gamification provides evidence supporting our research approach. Second, Chapter 4, we introduced a recent breakthrough that enables machine-intelligent features to run on users' devices. We could optimize data collection for mobile activity recognition studies by exploiting on-device deep learning inference as real-time feedback. The experimental results showed that our proposed method had improvements in both data quality and data quantity. Third, Chapter 5, we addressed activity recognition using constrained mobile devices. We introduced a system design of integrating on-device personalization and activity recognition, which allows activity recognition applications for smartphone sensor systems to achieve highly accurate training datasets. We developed the proposed system based on three essential features: on-device fine-tuning, model optimization, and personalized feedback. The results indicated that the proposed system could achieve accurate and consistent labeling in activity datasets. Lastly, I confirmed that I achieved all of my research goals to construct a useful and intelligent data collection of human-labeled data in mobile activity recognition studies. Additionally, my thesis's new findings are also applicable to other research and systems, as described in future work.

Bibliography

- [1] Saeed Abdullah, Nicholas D Lane, and Tanzeem Choudhury. Towards population scale activity recognition: A framework for handling data diversity. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.
- [2] Khalid Abualsaud, Massudi Mahmuddin, Ramy Hussein, and Amr Mohamed. Performance evaluation for compression-accuracy trade-off using compressive sensing for eeg-based epileptic seizure detection in wireless tele-monitoring. In *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 231–236. IEEE, 2013.
- [3] Khalid Abualsaud, Massudi Mahmuddin, Mohammad Saleh, and Amr Mohamed. Ensemble classifier for epileptic seizure detection for imperfect eeg data. *The Scientific World Journal*, 2015, 2015.
- [4] Utku Günay Acer, Marc van den Broeck, Claudio Forlivesi, Florian Heller, and Fahim Kawsar. Scaling crowdsourcing with mobile workforce: A case study with belgian postal service. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–32, 2019.
- [5] Hande Alemdar, Tim LM van Kasteren, and Cem Ersoy. Using active learning to allow activity recognition on a large scale. In *International Joint Conference on Ambient Intelligence*, pages 105–114. Springer, 2011.
- [6] Shahriyar Amini and Yang Li. Crowdlearner: rapidly creating mobile recognizers using crowdsourcing. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pages 163–172, 2013.
- [7] Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, 1988.
- [8] Les E Atlas, David A Cohn, and Richard E Ladner. Training connectionist networks with queries and selective sampling. In *Advances in neural information processing systems*, pages 566–573, 1990.
- [9] Salikh Bagaveyev and Diane J Cook. Designing and evaluating active learning methods for activity recognition. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 469–478, 2014.
- [10] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *International conference on pervasive computing*, pages 1–17. Springer, 2004.

- [11] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.
- [12] Michael Barz and Daniel Sonntag. Gaze-guided object classification using deep neural networks for attention-based computing. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pages 253–256, 2016.
- [13] Anahid Basiri, Muki Haklay, Giles Foody, and Peter Mooney. Crowdsourced geospatial data quality: Challenges and future directions, 2019.
- [14] Sultan Basudan, Xiaodong Lin, and Karthik Sankaranarayanan. A privacy-preserving vehicular crowdsensing-based road surface condition monitoring system using fog computing. *IEEE Internet of Things Journal*, 4(3):772–782, 2017.
- [15] Martin Berchtold, Matthias Budde, Dawud Gordon, Hedda R Schmidtke, and Michael Beigl. Actiserv: Activity recognition service for mobile phones. In *International Symposium on Wearable Computers (ISWC) 2010*, pages 1–8. IEEE, 2010.
- [16] Paul E Black. Greedy algorithm. *Dictionary of Algorithms and Data Structures*, 2:62, 2005.
- [17] Dimitar Bounov, Anthony DeRossi, Massimiliano Menarini, William G Griswold, and Sorin Lerner. Inferring loop invariants through gamification. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.
- [18] Leo Breiman. *Classification and regression trees*. Routledge, 2017.
- [19] Hennie Brugman, Albert Russel, and Xd Nijmegen. Annotating multi-media/multi-modal resources with elan. In *LREC*, 2004.
- [20] Muhammed Fatih Bulut, Murat Demirbas, and Hakan Ferhatosmanoglu. Lineking: Coffee shop wait-time monitoring using smartphones. *IEEE Transactions on Mobile Computing*, 14(10):2045–2058, 2014.
- [21] Jeffrey A Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B Srivastava. Participatory sensing. 2006.
- [22] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Yacine Kessaci, Frédéric Oblé, and Gianluca Bontempi. Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, 2019.
- [23] Yung-Ju Chang, Gaurav Paruthi, and Mark W Newman. A field study comparing approaches to collecting annotated activity data in real-world settings. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 671–682, 2015.
- [24] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- [25] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, and Yuan Tang. Xgboost: extreme gradient boosting. *R package version 0.4-2*, pages 1–4, 2015.

- [26] Tianshi Chen, Zidong Du, Ninghui Sun, Jia Wang, Chengyong Wu, Yunji Chen, and Olivier Temam. Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning. *ACM SIGARCH Computer Architecture News*, 42(1):269–284, 2014.
- [27] Yu-Hsin Chen, Joel Emer, and Vivienne Sze. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. *ACM SIGARCH Computer Architecture News*, 44(3):367–379, 2016.
- [28] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.
- [29] Delphine Christin, Andreas Reinhardt, Salil S Kanhere, and Matthias Hollick. A survey on privacy in mobile participatory sensing applications. *Journal of systems and software*, 84(11):1928–1946, 2011.
- [30] Allan H Church. Estimating the effect of incentives on mail survey response rates: A meta-analysis. *Public opinion quarterly*, 57(1):62–79, 1993.
- [31] Federico Cruciani, Ian Cleland, Chris Nugent, Paul McCullagh, Kåre Synnes, and Josef Hallberg. Automatic annotation for human activity recognition in free living using a smartphone. *Sensors*, 18(7):2203, 2018.
- [32] Eduardo Cuervo, Aruna Balasubramanian, Dae-ki Cho, Alec Wolman, Stefan Saroiu, Ranveer Chandra, and Paramvir Bahl. Maui: making smartphones last longer with code offload. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 49–62. ACM, 2010.
- [33] Nur Çürükoğlu and Buse Melis Özyildirim. Deep learning on mobile systems. In *2018 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–4. IEEE, 2018.
- [34] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, pages 1269–1277, 2014.
- [35] Sebastian Deterding, Dan Dixon, Rilla Khaled, and Lennart Nacke. From game design elements to gamefulness: defining "gamification". In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*, pages 9–15, 2011.
- [36] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O’Hara, and Dan Dixon. Gamification. using game-design elements in non-gaming contexts. In *CHI’11 extended abstracts on human factors in computing systems*, pages 2425–2428. 2011.
- [37] Zhiguo Ding and Minrui Fei. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, 46(20):12–17, 2013.
- [38] Souad Djelassi and Isabelle Decoopman. Customers’ participation in product development through crowdsourcing: Issues and implications. *Industrial Marketing Management*, 42(5):683–692, 2013.

- [39] Olive Jean Dunn. Multiple comparisons among means. *Journal of the American statistical association*, 56(293):52–64, 1961.
- [40] Biyi Fang, Xiao Zeng, and Mi Zhang. Nestdnn: Resource-aware multi-tenant on-device deep learning for continuous mobile vision. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 115–127, 2018.
- [41] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.
- [42] Zachary Fitz-Walter and Dian W Tjondronegoro. Exploring the opportunities and challenges of using mobile sensing for gamification and achievements. In *UbiComp 11: Proceedings of the 2011 ACM Conference on Ubiquitous Computing*, pages 1–5. ACM Press, 2011.
- [43] Brian Fitzgerald. The transformation of open source software. *MIS quarterly*, pages 587–598, 2006.
- [44] Paul Föckler, Thomas Zeidler, Benjamin Brombach, Erich Bruns, and Oliver Bimber. Phoneguide: museum guidance supported by on-device object recognition on mobile phones. In *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, pages 3–10. ACM, 2005.
- [45] Jesús Fontecha, Fco Javier Navarro, Ramón Hervás, and José Bravo. Elderly frailty detection by using accelerometer-enabled smartphones and clinical information records. *Personal and ubiquitous computing*, 17(6):1073–1083, 2013.
- [46] Raghu K Ganti, Fan Ye, and Hui Lei. Mobile crowdsensing: current state and future challenges. *IEEE communications Magazine*, 49(11):32–39, 2011.
- [47] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [48] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6645–6649. IEEE, 2013.
- [49] Penny Grubb and Armstrong A Takang. *Software maintenance: concepts and practice*. World Scientific, 2003.
- [50] Tao Guan, Yunfeng He, Juan Gao, Jianzhong Yang, and Junqing Yu. On-device mobile visual location recognition by integrating vision and inertial sensors. *IEEE transactions on multimedia*, 15(7):1688–1699, 2013.
- [51] Bin Guo, Zhiwen Yu, Liming Chen, Xingshe Zhou, and Xiaojuan Ma. Mobigroup: Enabling lifecycle support to social activity organization and suggestion with mobile crowd sensing. *IEEE Transactions on Human-Machine Systems*, 46(3):390–402, 2015.
- [52] Bin Guo, Zhiwen Yu, Xingshe Zhou, and Daqing Zhang. From participatory sensing to mobile crowd sensing. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*, pages 593–598.

- IEEE, 2014.
- [53] Juho Hamari, Jonna Koivisto, and Harri Sarsa. Does gamification work?—a literature review of empirical studies on gamification. In *2014 47th Hawaii international conference on system sciences*, pages 3025–3034. Ieee, 2014.
- [54] Seungyeop Han, Haichen Shen, Matthai Philipose, Sharad Agarwal, Alec Wolman, and Arvind Krishnamurthy. Mcdnn: An approximation-based execution framework for deep stream processing under resource constraints. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, pages 123–136, 2016.
- [55] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A Horowitz, and William J Dally. Eie: efficient inference engine on compressed deep neural network. *ACM SIGARCH Computer Architecture News*, 44(3):243–254, 2016.
- [56] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- [57] Mark Andrew Hanson. *Wireless body area sensor network technology for motion-based health assessment*, volume 71. 2009.
- [58] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P Bigham. A data-driven analysis of workers’ earnings on amazon mechanical turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2018.
- [59] Kim Hartman. How do intrinsic and extrinsic motivation correlate with each other in open source software development?, 2011.
- [60] Yu-chen Ho, Ching-hu Lu, I-han Chen, Shih-shinh Huang, Ching-yao Wang, Li-chen Fu, et al. Active-learning assisted self-reconfigurable activity recognition in a dynamic environment. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1567–1572. IEEE Press, 2009.
- [61] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.
- [62] HM Sajjad Hossain, Md Abdullah Al Hafiz Khan, and Nirmalya Roy. Active learning enabled activity recognition. *Pervasive and Mobile Computing*, 38:312–330, 2017.
- [63] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European conference on computer vision (ECCV)*, pages 0–0, 2018.
- [64] Masaya Inoue, Sozo Inoue, and Takeshi Nishida. Deep recurrent neural network for mobile human activity recognition with high throughput. *Artificial Life and Robotics*, 23(2):173–185, 2018.
- [65] Sozo Inoue, Paula Lago, Tahera Hossain, Tittaya Mairittha, and Nattaya Mairittha. Integrating activity recognition and nursing care records: The system, deployment,

- and a verification study. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(3):1–24, 2019.
- [66] Sozo Inoue, Nattaya Mairittha, Tittaya Mairittha, and Tahera Hossain. Integrating activity recognition and nursing care records: the system, experiment, and the dataset. In *International Conference on Activity and Behavior Computing*, page To appear. IEEE, 2019.
- [67] Sozo Inoue and Xincheng Pan. Supervised and unsupervised transfer learning for activity recognition from simple in-home sensors. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 20–27, 2016.
- [68] Eiichi Iwamoto, Masaki Matsubara, Chihiro Ota, Satoshi Nakamura, Tsutomu Terada, Hiroyuki Kitagawa, and Atsuyuki Morishima. Passerby crowdsourcing: Workers’ behavior and data quality management. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(4):1–20, 2018.
- [69] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.
- [70] Ingook Jang, Hyunseok Kim, Donghun Lee, Young-Sung Son, and Seonghyun Kim. Knowledge transfer for on-device deep reinforcement learning in resource constrained edge computing systems. *IEEE Access*, 8:146588–146597, 2020.
- [71] James M Keller, Michael R Gray, and James A Givens. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, (4):580–585, 1985.
- [72] Wazir Zada Khan, Yang Xiang, Mohammed Y Aalsalem, and Quratulain Arshad. Mobile phone sensing systems: A survey. *IEEE Communications Surveys & Tutorials*, 15(1):402–427, 2012.
- [73] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [74] Michael Kipp. Anvil-a generic annotation tool for multimodal dialogue. In *Seventh European Conference on Speech Communication and Technology*, 2001.
- [75] J Konečný, HB McMahan, FX Yu, P Richtárik, AT Suresh, and D Bacon. Federated learning: Strategies for improving communication efficiency. arxiv 2016. *arXiv preprint arXiv:1610.05492*.
- [76] Jakub Konečný, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.
- [77] Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.
- [78] William H Kruskal and W Allen Wallis. Use of ranks in one-criterion variance

- analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.
- [79] Karthik Kumar, Jibang Liu, Yung-Hsiang Lu, and Bharat Bhargava. A survey of computation offloading for mobile systems. *Mobile Networks and Applications*, 18(1):129–140, 2013.
- [80] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.
- [81] Peter A Lachenbruch and M Goldstein. Discriminant analysis. *Biometrics*, pages 69–85, 1979.
- [82] Nicholas D Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, Lei Jiao, Lorena Qendro, and Fahim Kawsar. Deepx: A software accelerator for low-power deep learning inference on mobile devices. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–12. IEEE, 2016.
- [83] Nicholas D Lane, Sourav Bhattacharya, Akhil Mathur, Petko Georgiev, Claudio Forlivesi, and Fahim Kawsar. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing*, 16(3):82–88, 2017.
- [84] Nicholas D Lane, Petko Georgiev, and Lorena Qendro. Deeppear: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 283–294, 2015.
- [85] Nicholas D Lane, Emiliano Miluzzo, Hong Lu, Daniel Peebles, Tanzeem Choudhury, and Andrew T Campbell. A survey of mobile phone sensing. *IEEE Communications magazine*, 48(9):140–150, 2010.
- [86] Nicholas D Lane, Ye Xu, Hong Lu, Shaohan Hu, Tanzeem Choudhury, Andrew T Campbell, and Feng Zhao. Enabling large-scale human activity inference on smartphones using community similarity networks (csn). In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 355–364, 2011.
- [87] Maxwell G Lay et al. Public infrastructure: An historical perspective. *Road & Transport Research: A Journal of Australian and New Zealand Research and Practice*, 22(2):62, 2013.
- [88] Tak Yeon Lee, Casey Dugan, Werner Geyer, Tristan Ratchford, Jamie Rasmussen, N Sadat Shami, and Stela Lupushor. Experiments on motivational feedback for crowdsourced workers. In *Seventh International AAAI Conference on Weblogs and Social Media*, 2013.
- [89] Janette Lehmann, Mounia Lalmas, Elad Yom-Tov, and Georges Dupret. Models of user engagement. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 164–175. Springer, 2012.
- [90] Joe Lemley, Shabab Bazrafkan, and Peter Corcoran. Deep learning for consumer devices and services: Pushing the limits for machine learning, artificial intelligence, and computer vision. *IEEE Consumer Electronics Magazine*, 6(2):48–56, 2017.

- [91] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR' 94*, pages 3–12. Springer, 1994.
- [92] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE Network*, 32(1):96–101, 2018.
- [93] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.
- [94] Mark W Lipsey and David B Wilson. *Practical meta-analysis*. SAGE publications, Inc, 2001.
- [95] Jinwei Liu, Haiying Shen, and Lei Yu. Question quality analysis and prediction in community question answering services with coupled mutual reinforcement. *IEEE Transactions on Services Computing*, 10(2):286–301, 2015.
- [96] Brent Longstaff, Sasank Reddy, and Deborah Estrin. Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, pages 1–7. IEEE, 2010.
- [97] Nattaya Mairittha and Sozo Inoue. Gamification for high-quality dataset in mobile activity recognition. In *International Conference on Mobile Computing, Applications, and Services*, pages 216–222. Springer, 2018.
- [98] Nattaya Mairittha and Sozo Inoue. Crowdsourcing system management for activity data with mobile sensors. In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 85–90. IEEE, 2019.
- [99] Nattaya Mairittha and Sozo Inoue. Improving annotation for activity recognition with active learning and gamification. *研究報告高齢社会デザイン (ASD)*, 2019(5):1–8, 2019.
- [100] Nattaya Mairittha and Sozo Inoue. Robust activity data collection with on-device recognition using long short-term memory. *研究報告高齢社会デザイン (ASD)*, 2019(12):1–8, 2019.
- [101] Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. A mobile app for nursing activity recognition. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, pages 400–403, 2018.
- [102] Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep learning inference for efficient activity data collection. *Sensors*, 19(15):3434, 2019.
- [103] Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Optimizing activity data collection with gamification points using uncertainty based active learning. 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*, pages 761–767, 2019.

- [104] Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. Improving activity data collection with on-device personalization using fine-tuning. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 255–260, 2020.
- [105] Nattaya Mairittha, Tittaya Mairittha, and Sozo Inoue. On-device deep personalization for robust activity data collection. *Sensors*, 21(1):41, 2021.
- [106] Nattaya Mairittha, Tittaya Mairittha, Paula Lago, and Sozo Inoue. Crowdact: Achieving high-quality crowdsourced datasets in mobile activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(1):1–32, 2021.
- [107] Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Başar, and Jean-Pierre Hubaux. Game theory meets network security and privacy. *ACM Computing Surveys (CSUR)*, 45(3):1–39, 2013.
- [108] Jane McGonigal. *Reality is broken: Why games make us better and how they can change the world*. Penguin, 2011.
- [109] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agueray Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
- [110] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.
- [111] Francisco Javier Ordóñez Morales and Daniel Roggen. Deep convolutional feature transfer across mobile activity recognition domains, sensor modalities and locations. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, pages 92–99, 2016.
- [112] Mohamed Musthag, Andrew Raij, Deepak Ganesan, Santosh Kumar, and Saul Shiffman. Exploring micro-incentive strategies for participant compensation in high-burden studies. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 435–444, 2011.
- [113] Francisco Javier Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.
- [114] Ekin Ozer, Maria Q Feng, and Dongming Feng. Citizen sensors for shm: Towards a crowdsourcing platform. *Sensors*, 15(6):14591–14614, 2015.
- [115] Heather L O’ Brien, Paul Cairns, and Mark Hall. A practical approach to measuring user engagement with the refined user engagement scale (ues) and new ues short form. *International Journal of Human-Computer Studies*, 112:28–39, 2018.
- [116] Maria V Palacin-Silva, Antti Knutas, Maria Angela Ferrario, Jari Porras, Jouni Ikonen, and Chandara Chea. The role of gamification in participatory environmental

- sensing: A study in the wild. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.
- [117] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 10(22):1345–1359, 2010.
- [118] ANANT Pande, K Sivakumar, S Sathyakumar, R SURESH Kumar, JA Johnson, Samrat Mondol, and Vinod B Mathur. Monitoring wildlife and their habitats in the southern ocean and around indian research stations in antarctica. *Proceedings of the Indian National Science Academy*, 83(2):483–496, 2017.
- [119] Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419, 2010.
- [120] Al-Sakib Khan Pathan. *Crowd Assisted Networking and Computing*. CRC Press, 2018.
- [121] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.
- [122] Martin Pielot, Karen Church, and Rodrigo De Oliveira. An in-situ study of mobile phone notifications. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*, pages 233–242, 2014.
- [123] Ming-Zher Poh, Kyunghee Kim, Andrew D Goessling, Nicholas C Swenson, and Rosalind W Picard. Heartphones: Sensor earphones and mobile application for non-obtrusive health monitoring. In *2009 International Symposium on Wearable Computers*, pages 153–154. IEEE, 2009.
- [124] Sihang Qiu, Ujwal Gadiraju, and Alessandro Bozzon. Improving worker engagement through conversational microtask crowdsourcing.
- [125] Khandakar M Rashid and Joseph Louis. Times-series data augmentation and deep learning for construction equipment activity recognition. *Advanced Engineering Informatics*, 42:100944, 2019.
- [126] Reza Rawassizadeh, Elaheh Momeni, Chelsea Dobbins, Joobin Gharibshah, and Michael Pazzani. Scalable daily human behavioral pattern mining from multivariate temporal data. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):3098–3112, 2016.
- [127] Reza Rawassizadeh, Timothy J Pierson, Ronald Peterson, and David Kotz. Nocloud: Exploring network disconnection through on-device data analysis. *IEEE Pervasive Computing*, 17(1):64–74, 2018.
- [128] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):1–27, 2010.
- [129] Attila Reiss and Didier Stricker. Creating and benchmarking a new dataset for physical activity monitoring. In *Proceedings of the 5th International Conference on*

- Pervasive Technologies Related to Assistive Environments*, pages 1–8, 2012.
- [130] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.
- [131] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.
- [132] Guido Sautter and Klemens Böhm. High-throughput crowdsourcing mechanisms for complex tasks. *Social Network Analysis and Mining*, 3(4):873–888, 2013.
- [133] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.
- [134] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [135] Burr Settles. *Active Learning*, volume 18. Morgan & Claypool Publishers, 2012.
- [136] Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.
- [137] Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.
- [138] Minh Shin, Cory Cornelius, Apu Kapadia, Nikos Triandopoulos, and David Kotz. Location privacy for mobile crowd sensing through population mapping. *Sensors*, 15(7):15285–15310, 2015.
- [139] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016.
- [140] Sarabjot Singh, Mikhail Geraseminko, Shu-ping Yeh, Nageen Himayat, and Shilpa Talwar. Proportional fair traffic splitting and aggregation in heterogeneous wireless networks. *IEEE Communications Letters*, 20(5):1010–1013, 2016.
- [141] Maja Stikic, Kristof Van Laerhoven, and Bernt Schiele. Exploring semi-supervised and active learning for activity recognition. In *2008 12th IEEE International Symposium on Wearable Computers*, pages 81–88. IEEE, 2008.
- [142] Riny Sulistyowati, Hari Agus Sujono, and Ahmad Khamdi Musthofa. A river water level monitoring system using android-based wireless sensor networks for a flood early warning system. In *Proceedings of Second International Conference on Electrical Systems, Technology and Information 2015 (ICESTI 2015)*, pages 401–408. Springer, 2016.
- [143] Jingchao Sun, Rui Zhang, Xiaocong Jin, and Yanchao Zhang. Securefind: Secure and privacy-preserving object finding via mobile crowdsourcing. *IEEE Transactions on Wireless Communications*, 15(3):1716–1728, 2015.
- [144] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for

- language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.
- [145] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.
- [146] Nadeem Ahmed Syed, Syed Huan, Liu Kah, and Kay Sung. Incremental learning with support vector machines. 1999.
- [147] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.
- [148] Arvind Thiagarajan, Lenin Ravindranath, Katrina LaCurts, Samuel Madden, Hari Balakrishnan, Sivan Toledo, and Jakob Eriksson. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *Proceedings of the 7th ACM conference on embedded networked sensor systems*, pages 85–98, 2009.
- [149] Emma L Tonkin, Alison Burrows, Przemysław R Woznowski, Pawel Laskowski, Kristina Y Yordanova, Niall Twomey, and Ian J Craddock. Talk, text, tag? understanding self-annotation of smart home data from a user’s perspective. *Sensors*, 18(7):2365, 2018.
- [150] Francisco J Valverde-Albacete and Carmen Peláez-Moreno. 100% classification accuracy considered harmful: The normalized information transfer factor explains the accuracy paradox. *PloS one*, 9(1):e84217, 2014.
- [151] Niels Van Berkel, Jorge Goncalves, Simo Hosio, and Vassilis Kostakos. Gamification of mobile experience sampling improves data quality and quantity. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):1–21, 2017.
- [152] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. *arXiv preprint arXiv:1610.05202*, 2016.
- [153] Gilles Virone, A Wood, Leo Selavo, Quihua Cao, Lei Fang, Thao Doan, Zhimin He, and J Stankovic. An advanced wireless sensor network for health monitoring. In *Transdisciplinary conference on distributed diagnosis and home healthcare (D2H2)*, pages 2–4. Citeseer, 2006.
- [154] Luis Von Ahn and Laura Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.
- [155] Maja Vukovic, Rajarshi Das, and Soundar Kumara. From sensing to controlling: the state of the art in ubiquitous crowdsourcing. *International Journal of Communication Networks and Distributed Systems*, 11(1):11–25, 2013.
- [156] Julie Wallace, Denis Corr, Patrick Deluca, Pavlos Kanaroglou, and Brian McCarry. Mobile monitoring of air pollution in cities: the case of hamilton, ontario, canada. *Journal of Environmental Monitoring*, 11(5):998–1003, 2009.
- [157] Xiangpeng Wan, Hakim Ghazzai, and Yehia Massoud. Mobile crowdsourcing for in-

- telligent transportation systems: Real-time navigation in urban areas. *IEEE Access*, 7:136995–137009, 2019.
- [158] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 119:3–11, 2019.
- [159] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beau-fays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.
- [160] Yufeng Wang, Xueyu Jia, Qun Jin, and Jianhua Ma. Quacentine: a quality-aware incentive mechanism in mobile crowdsourced sensing (mcs). *The Journal of Super-computing*, 72(8):2924–2941, 2016.
- [161] Yufeng Wang and Jianhua Ma. *Mobile social networking and computing: A multi-disciplinary integrated perspective*. CRC Press, 2014.
- [162] Yu Xiao, Pieter Simoens, Padmanabhan Pillai, Kiryong Ha, and Mahadev Satya-narayanan. Lowering the barriers to large-scale mobile crowdsensing. In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, pages 1–6, 2013.
- [163] Mengwei Xu, Feng Qian, Qiaozhu Mei, Kang Huang, and Xuanzhe Liu. Deeptype: On-device deep learning for input personalization service with minimal privacy concern. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(4):1–26, 2018.
- [164] Fei Yan, Josef Kittler, David Windridge, William Christmas, Krystian Mikolajczyk, Stephen Cox, and Qiang Huang. Automatic annotation of tennis games: An integra-tion of audio, vision, and learning. *Image and Vision Computing*, 32(11):896–903, 2014.
- [165] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.
- [166] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. A survey of crowdsourcing systems. In *2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing*, pages 766–773. IEEE, 2011.
- [167] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. In *Advances in neural information processing systems*, pages 685–693, 2015.
- [168] Liyue Zhao, Gita Sukthankar, and Rahul Sukthankar. Robust active learning using crowdsourced annotations for activity recognition. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.
- [169] Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep au-toencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on*

- Knowledge Discovery and Data Mining*, pages 665–674, 2017.
- [170] Huan Zhou, Jiming Chen, Jialu Fan, Yuan Du, and Sajal K Das. Consub: Incentive-based content subscribing in selfish opportunistic mobile networks. *IEEE Journal on Selected Areas in Communications*, 31(9):669–679, 2013.
- [171] Zhenyu Zhou, Haijun Liao, Bo Gu, Kazi Mohammed Saidul Huq, Shahid Mumtaz, and Jonathan Rodriguez. Robust mobile crowd sensing: When deep learning meets edge computing. *IEEE Network*, 32(4):54–60, 2018.
- [172] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1):44–53, 2018.
- [173] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.
- [174] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.
- [175] Mengdie Zhuang and Ujwal Gadiraju. In what mood are you today? an analysis of crowd workers’ mood, performance and engagement. In *Proceedings of the 10th ACM Conference on Web Science*, pages 373–382, 2019.