Collaborative Crowdsensing at the Edge

Yifan Du

Sorbonne University

École Doctorale Informatique, Télécommunications et Électronique de Paris
Inria Paris - MiMove Team

Collaborative Crowdsensing at the Edge

By **Yifan Du**

Doctoral Thesis in Computer Science

Under the supervision of Valérie Issarny and Françoise Sailhan

Presented and defended publicly on July 21, 2020

In front of a jury composed of:

Prof. Christine Julien (University of Texas at Austin, US)	Reviewer
Prof. Mirco Musolesi (University College London, UK)	Reviewer
Dr. Aline Carneiro Viana (Inria Saclay, FR)	Examiner
Dr. Hamed Haddadi (Imperial College London, UK)	Examiner
Prof. Pierre Sens (Sorbonne University, FR)	Examiner
Dr. Valérie Issarny (Inria Paris, FR)	Advisor
Dr. Françoise Sailhan (CNAM Paris, FR)	Co-advisor

Abstract

Mobile crowdsensing is a powerful mechanism to contribute to the ubiquitous sensing of data at a relatively low cost. With mobile crowdsensing, people provide valuable observations across time and space using sensors embedded in/connected to their smart devices, e.g., smartphones. Particularly, opportunistic crowdsensing empowers citizens to sense objective phenomena at an urban and fine-grained scale, leveraging an application running in the background. Still, crowdsensing faces challenges: The relevance of the provided measurements depends on the adequacy of the sensing context with respect to the phenomenon that is analyzed; The uncontrolled collection of massive data leads to low sensing quality and high resource consumption on devices; Crowdsensing at scale also involves significant communication, computation, and financial costs due to the dependence on the cloud for the post-processing of raw sensing data.

This thesis aims to establish opportunistic crowdsensing as a reliable means of environmental monitoring. We advocate enforcing the cost-effective collection of high-quality data and inference of the physical phenomena at the end device. To this end, our research focuses on defining a set of protocols that together implement *collaborative crowdsensing at the edge*, combining:

- *Inference of the crowdsensor's physical context characterizing the gathered data*: We assess the context beyond geographical position. We introduce an online learning approach running on the device to overcome the diversity of the classification performance due to the heterogeneity of the crowdsensors. We specifically introduce a hierarchical algorithm for context inference that requires little feedback from users, while increasing the inference accuracy per user.

- *Context-aware grouping of crowdsensors to share the workload and support selective sensing*: We introduce an *ad hoc* collaboration strategy, which groups co-located crowdsensors together, and assigns them various roles according to their respective contexts. Evaluation results show that the overall resource consumption due to crowdsensing is reduced, and the data quality is enhanced, compared to the cloud-centric architecture.

- *Data aggregation on the move to enhance the knowledge transferred to the cloud*: We introduce a distributed interpolation-mediated aggregation approach running on the end device. We model interpolation as a tensor completion problem and propose tensor-wise aggregation, which is performed when crowdsensors encounter. Evaluation results show significant savings in terms of cellular communication, cloud computing, and, therefore, financial costs, while the overall data accuracy remains comparable to the cloud-centric approach.

In summary, the proposed collaborative crowdsensing approach reduces the costs at both the end device and the cloud, while increasing the overall data quality.

Key Words Context Inference, Crowdsensing, Middleware, Mobile Sensing, Context Awareness, Environmental Monitoring, Ubiquitous Sensing, Pervasive Computing, Data Aggregation, Opportunistic Relay.

Résumé

Le *crowdsensing* mobile permet d'obtenir des données sur l'environnement à un coût relativement faible. De fait, les personnes peuvent collecter et partager des observations spatio-temporelles au moyen de capteurs intégrés dans les appareils intelligents comme les smartphones. En particulier, le *crowdsensing opportuniste* permet aux citoyens de détecter des phénomènes environnementaux à l'échelle urbaine grâce à une application dédiée s'exécutant en arrière plan. Cependant, le *crowdsensing* est confronté à différents défis : la pertinence des mesures fournies dépend de l'adéquation entre le contexte de détection et le phénomène analysé ; la collecte incontrôlée de données entraîne une faible qualité de détection ainsi qu'une forte consommation des ressources au niveau des appareils ; le *crowdsensing* à large échelle induit des coûts importants de communication, de calcul et financiers en raison de la dépendance au *cloud* pour le traitement des données brutes.

Notre thèse vise à rendre le *crowdsensing opportuniste* comme un moyen fiable d'observation de l'environnement urbain. Pour ce faire, nous préconisons de favoriser la collecte et l'inférence du phénomène physique au plus proche de la source. À cet effet, notre recherche se concentre sur la définition d'un ensemble de protocoles complémentaires, qui mettent en œuvre un *crowdsensing collaboratif* entre les noœuds mobiles en combinant :

- *L'inférence du contexte physique du crowdsensor*, qui caractérise les données recueillies. Nous évaluons un contexte qui ne se limite par à une simple position géographique et nous introduisons une technique d'apprentissage du contexte qui s'effectue au niveau de l'appareil afin de pallier l'impact de l'hétérogénéité des *crowdsensors* sur la classification. Nous introduisons spécifiquement un algorithme hiérarchique pour l'inférence du contexte qui limite les interactions avec l'usager, tout en augmentant la précision de l'inférence.

- *Le groupement contextuel des crowdsensors* de manière à partager la charge et effectuer une captation sélective. Nous introduisons une stratégie de collaboration *ad hoc* qui vise à affecter différents rôles aux *crowdsensors*, afin de répartir la charge entre les *crowdsensors* proches en fonction de leur contexte respectif. L'évaluation de notre solution montre une réduction de la consommation globale des ressources et une meilleure qualité des données, comparée à celle basée sur une architecture *cloud*.

- *L'agrégation nomade de données* pour améliorer les connaissances transférées au *cloud*. Nous introduisons une solution à l'agrégation des données observées qui distribue l'interpolation sur les *crowdsensors*. Les résultats de son évaluation montrent des gains importants en termes de communication cellulaire, de calcul sur le *cloud* et donc de coûts financiers, tandis que la précision globale des données reste comparable à une approche centralisée.

En résumé, l'approche collaborative proposée pour le *crowdsensing* réduit les coûts à la fois sur le terminal et sur le *cloud*, tout en augmentant la qualité globale des données.

Mots clés Inférence de contexte, Contexte de détection, *Crowdsensing*, *Middleware*, Détection mobile, Surveillance de l'environnement, Détection omniprésente, Calcul ubiquitaire, Agrégation de données, Relais opportuniste.

Acknowledgments

The three years I spent on my PhD thesis has definitely been one of the most precious memories in my entire life. I have learned the means to carry out research studies in a systematic manner and expanded my horizons. It is not only through my own effort but the support of many people that I was able to achieve this outcome.

To begin with, I would like to thank my supervisors Valérie Issarny and Françoise Sailhan for welcoming me to their team and for offering their generous guidance throughout my time as a PhD candidate. They were very patient and attentive with helping me even I made some fundamental mistakes. I also want to express my gratitude to all my colleagues during my studies at Inria Paris, France - MiMove Team, particularly Nikolaos Georgantas, Nathalie Gaudechoux, Rachit Agarwal, Georgios Bouloukakis, and Patient Ntumba.

I am indebted to my reviewers, Christine Julien and Mirco Musolesi for their invaluable suggestions and feedback. I would also like to thank the rest of my thesis committee, Aline Carneiro Viana, Hamed Haddadi and Pierre Sens, for their participation in my jury.

Second, I am grateful to Jiazi Yi and Thomas Heide Clausen, my supervisors for the master and internships at Ecole Polytechnique, France. If not for their guidance on academic work and research, I would not have chosen to pursue a PhD degree.

Finally, I express my deepest gratitude to my parents Jihe Du and Meng Luo for their unconditional support and love ever since I came into this world. I would not have been able to accomplish any of this without them. I would also like to thank my best friend Frank Howell and my four-paw furry friend Koffee, who have accompanied and supported me during these years.

Yifan Du
July 2020
Paris, France

Contents

Acronyms

AI	Artificial Intelligence
ALS	Alternating Least Square
AP	Access Point
D2D	Device to Device
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
GCP	Google Cloud Platform
GPR	Gaussian Process Regression
GPS	Global Positioning System
GSM	Global System for Mobile Communications
IBk	Instance Based k-nearest neighbor
IoT	Internet of Things
KNN	K-Nearest Neighbor algorithm
LWL	Locally Weighted Learning
MANET	Mobile ad hoc Network
MAPE	Mean Absolute Percentage Error
OLS	Ordinary Least Squares
P2P	Peer to Peer
RBF	Radial Basis Function
RMSE	Root Mean Square Error
RSSI	Received Signal Strength Indicator
SGD	Stochastic Gradient Descent
WLAN	Wireless Local Area Network
WSN	Wireless Sensor Network

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Crowdsensing: A Ubiquitous Data Source

Mobile crowdsensing [67] is a sensing paradigm that empowers ordinary people to contribute with data sensed from or generated by their sensor-enhanced mobile devices. In other words, crowdsensing introduces a new shift in the way we collect data by permitting to acquire local knowledge through the smart devices attached to people [100], such as smartphones, tablets, smartwatches [72]. As we all know, sensing is an essential method for acquiring information or data about the phenomena. Recently, both academia and industry have witnessed the rise of AI (Artificial Intelligence), including machine learning, deep learning, and reinforcement learning techniques. One of the key enabling bases of such an AI (r)evolution is the big data because data is the raw input of AI, and the intelligence results from the processed data; namely, most AI learns from the data and works on data. The demand of meaningful data is always a necessity to support the increasing AI development and application. Although the rapid growth of IoT (Internet of Things) devices has generated a huge amount of data around the world, it is still costly because many of them require manual deployment in local-scope and are based on short-range networks, e.g., smart home, smart parking, smart building solutions. Alternatively, mobile

crowdsensing is a very competitive technique because it is naturally a mobile data source for big data [114]. Furthermore, crowdsensing supports pervasive computing over observations, because it is not only a simple "Thing" as in the general definition of IoT; the device can be equipped with multiple network interfaces, computing capability, and even linked to human intelligence. In short, mobile crowdsensing does not only stand as a ubiquitous data source but is also a ubiquitous mini-computer.

The widespread availability of smartphones/tablets/smartwatches and the ever-increasing number of their embedded sensors have greatly participated in the adoption of mobile crowdsensing solutions. The crowdsensing device does not only include conventional sensors like motion sensors, position sensors, environment sensors, but also multimedia sensors like cameras, microphones, and even virtual sensors like social networks. There has been an outstanding growth of mobile crowdsensing proposals from both academia and industry in the decade [216]. Crowdsensing has numerous practical applications [141], such as monitoring the public service (e.g., time attendance to services, delivery tracking, traffic conditions, and roads, safety perception), monitoring the environment (e.g., noise and ambiance, atmospheric conditions, garbage, air quality, classify galaxy pictures), and enrichment of social media, just to name a few. Various types of applications have been developed to realize the potential of mobile crowdsensing.

In general, four aspects should be considered in the design of a mobile crowdsensing solution: the task (e.g., type, data, scale), the participation (e.g., involvement, location, knowledge), the data collection (e.g., routing, transmission, networking) and the processing (e.g., recruitment, workflow, analysis). At the same time, numerous options are available inside each [157]. For instance, the noise monitoring of a city block is a sensing task; the participants are ordinary citizens in that block; the sensing data is sent to a cloud, and the server generates a city noise map. Depending on the type of phenomenon being measured or mapped, the application of mobile crowdsensing can be divided into three categories [126]: (a) environmental, (b) infrastructural, and (c) social.

We illustrate in Figure 1.1.1 the high-level architecture of a typical mobile crowdsensing system. At the base layer, the contributor corresponds to ordinary people, e.g., citizens contributing to the city, as part of their daily life. People may have smart devices on-hand or in-pocket, they may stay inside buildings or in open space, and they may take subways or walk on the street. At the middle layer, mobile devices like smartphones/tablets are multimedia carriers that generate the sensing data and act as a proxy providing people access to the Internet. Mobile devices are usually belongings bound to people; these mobile devices and people are together called *"crowdsensors"*. The crowdsensing application is deployed on crowdsensors that embed sensors (or connected to personal devices that can be used to sense) or has access to multimedia data. At the top layer of the system, there is the cloud, where a server with a database is responsible for the processing, analysis, and storage of the resulting sensing data. It can also be a cloud service such as e.g., Google Cloud Platform, Microsoft Azure, or Amazon Web Services. Crowdsensors are con-

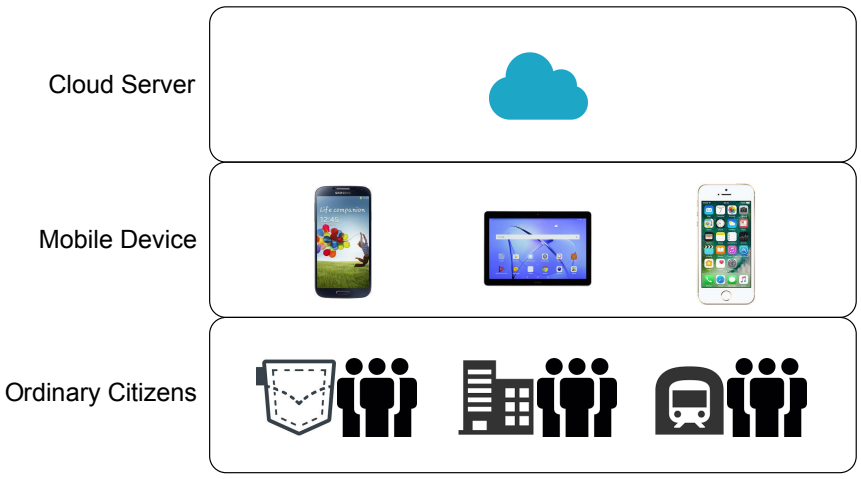


Figure 1.1.1: Crowdsensing system architecture

nected to the Internet via cellular networks or Wi-Fi access points such that they can transmit their sensing data to the cloud server.

In general, a mobile crowdsensing system implements a five-phases workflow [219], as described below:

1. *Service registration* - each crowdsensor needs to register its available services and location periodically to the cloud server such that it can be recruited to perform a crowdsensing task as needed.

2. *Task creation* - a crowdsensing task is defined at the cloud explicitly (i.e., by the client) or implicitly (i.e., automatically) to indicate the topic of interest and the attributes (e.g., sensing duration, location, and sensor types).

3. *Task assignment* - the cloud server selects the crowdsensors that should perform the crowdsensing task and sends the related requests. Thus, crowdsensors can be activated for given sensing works.

4. *Individual task execution* - the active crowdsensors that have been recruited carry out the sensing data collection (in a participatory or opportunistic manner), temporarily cache the sensing data and upload the data to the cloud server periodically.

5. *Data integration* - the cloud server processes and aggregates the received sensing data, and the (raw and/or analyzed) data is stored in the database for future queries by clients.

In short, mobile crowdsensing appears to be an effective way of observing a large-scale phenomenon based on the sensing data provided by people that are distributed ubiquitously.

1.1.1 Why is Crowdsensing not as Efficient as Expected?

Originally, the concept of crowdsensing was viewed as a specific type of crowdsourcing tasks [98], in which people were recruited and paid to perform some work. It was initially named as "*participatory sensing*" and participants were awarded money or virtual benefits. For example, if someone reports on Google Maps the traffic condition that she/he experienced, she/he may get traffic conditions of the entire city. In such participatory sensing, people need to actively engage in sensing activities by manually determining how, when, what, and where to sense. Obviously, getting enough participants on-board is the key to the ubiquitous sensing, and a critical aspect is the proactive efforts required from people. In order to get participatory sensing work effectively, a supporting incentive/recruitment mechanism is needed. Plenty of research works have been conducted to improve the participatory sensing approach, with the introduction of novel incentives [165, 60, 87], recruitment strategies [71, 118, 123, 14, 117, 213, 188], recruitment optimization [206, 189, 198], as well as security and privacy-preserving mechanisms [130]. The approaches mentioned above neutralize the burden put on participants by encouraging them.

What if people are not willing to be disturbed by crowdsensing even with altruism? Here comes a question: is it possible to lower or even eliminate the crowdsensing burden on people? In such a case, people would be less/not involved and thereby unaware of the sensing, and crowdsensing would become opportunistic. The answer to this question is affirmative. In practice, considering the user involvement, mobile crowdsensing supports two sensing strategies: *participatory crowdsensing* [68, 98, 157] and *opportunistic crowdsensing* [59, 135, 38]. The difference between these two schemes is intuitive [111]:

- *Participatory crowdsensing* requires the proactive involvement of individuals who consciously contribute with sensing data, by, e.g., taking a picture, reporting a road closure, noting a place, answering a questionnaire.
- *Opportunistic crowdsensing* collects sensing data in the background and does not necessitate any explicit action from the user, by, e.g., sampling noise continuously, capturing a photo randomly, tracking location, recording signal strength.

Since opportunistic crowdsensing does not require user actions, what it senses is usually a physical phenomenon (rather than a subjective view), and the application is also widely deployed [133, 105, 143, 172, 26, 38]. We believe that opportunistic crowdsensing is the best option to support ubiquitous sensing that can be running anytime and anywhere because it does not disturb the user. Meanwhile, the embedded sensors like temperature, pressure, humidity, proximity, and ambient light, as well as other more conventional components such as accelerometer, gyroscope, microphone, compass, and GPS (Global Positioning System) can be all used opportunistically to get measurement data [113]. Each new measurement can be uploaded with both spatial (e.g., location coordinates) and temporal information (e.g.,

timestamps). Especially, opportunistic crowdsensing appears to be a scalable and cost-effective alternative to the deployment of static WSN (Wireless Sensor Network) for the dense coverage of large areas, which is ideal for monitoring the urban environment.

However, every technique comes with a trade-off; the same applies to the comparison between participatory and opportunistic crowdsensing. Opportunistic crowdsensing shifts the burden of gathering high-quality data from people to the application and cloud server. The critical aspect is that opportunistic crowdsensing is uncontrolled: sensing activities are fully automated, usually running in the background, and do not involve the user. Thus, the system needs to be autonomous and intelligent because the user should be almost unaware of the crowdsensing procedure. How to effectively collect and process data that is sensed opportunistically is the most critical problem. In our opinion, opportunistic crowdsensing faces two major challenges: (i) the low data quality due to the diversity and accuracy of mobile sensors, and the uncontrolled sensing in the background; (ii) the high resource -including financial- cost due to the massively collected raw data from crowdsensors, and the reliance on cloud for post-processing. In particular, we elaborate on these two challenges as:

- **Low data quality -** First, an opportunistic crowdsensing system provides wide availability but is also characterized by low data accuracy [166]. Unlike a WSN, which has a specific function and is regularly calibrated by experts, heterogeneous crowdsensors often introduce high bias in sensor measurements. Obviously, one reason is the variety and heterogeneity of crowdsensing devices. However, the bias is not only due to the low accuracy/lack of calibration of the contributing sensors, but also due to the diversity of the sensing contexts. For instance, noise measurements collected from a device placed in a pocket do not correlate with the one collected out of pocket; temperature measurements collected out-door differ from the ones collected in-door (at the same location). Significant errors can be introduced when uncorrelated crowdsensing data are aggregated. Combining uncorrelated crowdsensing data introduces significant errors in the resulting knowledge.

  In short, the lack of crucial context information, about both accuracy assessment and device placement, may introduce significant errors in the aggregation of the crowdsensed measurements. In order to properly analyze the heterogeneous sensing data coming from crowdsensors, context information is necessary for understanding the contributed observations.

- **High sensing cost -** The second critical issue arising with opportunistic crowdsensing relates to its cost for both the individual crowdsensor, especially in terms of battery consumption and network traffic fees, and also for the infrastructure, regarding the uploading to – and further processing on – the cloud of massive raw data. Moreover, such a cost includes the one associated with the useless gathering of low-quality data. On the user side, crowdsensing

devices are typically battery-operated with limited access to energy, and they are occasionally plugged to power. We note that the device positioning using GPS and data uploading over cellular networks are the two main sources of power consumption for a crowdsensing task. At the same time, the one associated with the operation of the embedded environmental sensors is relatively negligible [29]. The continuous positioning and data transmission to the cloud leads to a dramatic usage of the energy and communication radio. In addition to power consumption, sensing data uploaded to the cloud is often charged by the operator according to the traffic amount. Even if users do not need to take action proactively, they do not like to be disrupted by the crowdsensing task, and the sensing cost significantly discourages users from contributing. On the infrastructure side, processing, analysis, and storage of sensing data on the server are charged as cloud services, according to the usage of resources depending on computational complexity and data amount. The cloud service provider also charges the connection and network flow for crowdsensing.

Overall, providing low-quality and a massive amount of raw measurements does not improve the crowdsensing utility but notably increases the sensing costs.

This thesis addresses the efficiency of opportunistic mobile crowdsensing with respect to these two criteria: *data quality* and *sensing cost*.

1.2 From Cloud to Edge: Powerful Crowdsensors

As aforementioned, opportunistic crowdsensing may continuously generate a massive amount of sensing data, which consumes many resources (e.g., bandwidth, energy, and cache) on end devices and may provide worthless information. There may be significant redundancy in the content of the sensing data gathered ultimately on the cloud [126]. In order to extract useful information from opportunistic crowdsensing, the traditional cloud-based crowdsensing system post-processes the vast amount of raw sensing data (through, e.g., filter, aggregation, data mining) received from ubiquitous crowdsensors [82]. Cloud-based crowdsensing often suffers from the high resource consumption on the crowdsensing device due to raw data collection and high operational cost on the centralized server due to data post-processing, hence the poor overall efficiency and scalability.

In recent years, the proliferation of IoT and the success of cloud computing and services have pushed forward the edge computing [174, 169], which is to process the data closer to end devices. In the IoT context, it has been shown that edge computing enhances mobile computing from the standpoint of scalability, battery life constraint, bandwidth cost-saving, as well as data safety and privacy [18]. The benefit of edge computing results in its adoption in, e.g., cloud offloading, smart home, smart city. Although the "Edge Computing" terminology refers to the deployment of edges server at the network border, we believe that the computing can be offloaded

to the very edge, that is, the end devices (crowdsensors). Nowadays, smartphones can perform many complex computing tasks such as activity recognition, photo enhancement, augmented reality, 3D online games, and even machine learning. Since the power of mobile devices like smartphones is significantly increasing, it is our perspective that in the future, any end device can play the role of an edge server. In such a situation, the edge computing can refer to collaboration among several end devices [91, 46]. We consider this type of in-network collaboration as a branch of edge computing.

Inspired by the rise of edge computing, we argue that combining the power of mobile edge computing and the large-scale opportunistic crowdsensing can realize higher efficiency and better coverage for the sensing of smart cities. Promoting collaboration at the very edge with opportunistic crowdsensing is beneficial with regards to three aspects:

- **Computing perspective:** The cloud-computing resource consumption can be offloaded to the edge. The smartphone's computing power has evolved at a very high speed, thanks to the semiconductor technology. For example, activity recognition using traditional machine learning [2, 66], and even deep learning (e.g., TensorFlow Lite), can run on smartphones [112, 81]. Future trends show that more energy, processing, and storage will be given to mobile smartphones that will deal with complex analysis. Thus, collaboration among mobile devices would be more contributory to the future mobile system. The collaboration will lead to very early data processing before uploading it to the cloud.

- **Networking perspective:** The data transmission to and reception on the cloud can be reduced. With the evolution of high-performance multi-network smartphones (e.g., Bluetooth, Wi-Fi, 5G interfaces), it has been shown that cellular networks can benefit from D2D (Device-to-Device) communications between smartphones [12, 13, 122]. Meanwhile, the MANET (Mobile Ad hoc NETwork) [1] has been studied for decades and has already been partly implemented in industry. It allows a decentralized type of wireless network does not rely on a pre-existing infrastructure, such as routers in wired networks or access points in managed wireless networks. For instance, Wi-Fi Direct [7] is a protocol that standardizes the direct communication between Wi-Fi-enabled mobile devices. D2D communication is an enabling technology to support ad hoc collaboration at the edge.

- **Human perspective:** The behavior and social interaction among people can be studied and leveraged. Citizens have habits, and their activities and interactions follow some patterns [56, 167, 180]. In reality, human mobility is not at all random, and the dynamics of citizen interactions mostly depend on social ties [40, 41]. The property of human mobility primarily supports collaboration among crowdsensors via D2D communication. For instance, in the

crowd computing [31], when an application is willing to perform a D2D task, it chooses the best peer with which to collaborate and sends the task to the latter. The collaboration requires discovering the nearby devices and evaluating the goodness of collaborative peers, based on the device and human/device holder capabilities.

We believe that the optimal method to achieve opportunistic crowdsensing efficiency is by enforcing low cost and – at the same time – high-quality data collection at the very edge. That is, maximizing the effectiveness of opportunistic crowdsensing rather than gathering massive raw sensing data at the cloud server (which requires costly post-processing to produce meaningful knowledge). The above leads us to argue that *"collaborative crowdsensing at the edge"* [46], which leverages D2D communication and human interactions, may significantly contribute to enabling cost-effective, high-quality opportunistic crowdsensing by moving part of the processing of the sensing data closer to the end devices.

1.3 Thesis Contribution and Outline

This thesis aims to raise opportunistic mobile crowdsensing to a reliable means of observing phenomena, focusing on urban environmental monitoring. That is, we address the challenges associated with *opportunistic crowdsensing in the context of urban environmental monitoring*. The mobile crowdsensors contribute measurements related to the physical environment (e.g., ambient temperature, air pressure, ambient humidity, ambient light, sound level, magnetic field.) using the embedded/connected sensors on smart devices.

For the purpose of increasing crowdsensing data quality while decreasing the crowdsensing cost both at end devices and on the cloud – thereby enhancing the overall efficiency of opportunistic mobile crowdsensing – we specifically promote the context-aware collaboration among crowdsensors. To this end, this thesis contributes to defining a set of protocols – from design to prototype implementation and evaluation – that together support *"context-aware collaborative mobile crowdsensing at the edge"* [46], by combining three complementary features:

- ***ContextSense*: personalized context inference -** The inference of physical context is essential to characterize the gathered crowdsensing data. Indeed, users may contribute valuable observations across time and space using sensors embedded in their smart devices. However, the relevance of the provided measurements depends on the adequacy of the sensing context with respect to the analyzed phenomena. While the meaning of *context* is broad, we concentrate more specifically on assessing the sensing context when gathering observations about the physical environment beyond its geographical position in the Euclidean space, that is, whether the sensor is in-/out-pocket, in-/out-door and upper-/under-ground.

The challenge is to devise a classifier that accounts for the diversity in the characteristics of contributing devices, the behaviors of contributing users, and even the usage scenarios. For instance, the inference needs to cope with the availability of features depending on the device and user preference. For this purpose, we personalize the classifier to overcome the disparity of the classification performance. We introduce an online learning approach to support the local inference of the sensing context that can evolve according to the environment in which it takes place. While the personalized inference of the sensing context is running on the device, feedback is requested to the end user to assess the correctness of the inference result and update the current classifier accordingly. Our approach features a hierarchical algorithm explicitly for the inference that limits the number of opportunistic feedback required to the user, while increasing the accuracy of the context inference for each user.

- ***BeTogether*: group-based crowdsensing -** The objective of the collaborative group is to leverage D2D communication so that co-located crowdsensors cooperate (e.g., filter/aggregate sensing data) and share the available resources (e.g., Internet access, location). Traditional cloud-based crowdsensing requires that each crowdsensor provides the spatio-temporal sensing data to the cloud server, which processes it. D2D wireless networks such as Bluetooth and Wi-Fi Direct have brought the ability to collaborate using short-range communication.

  In order to support cooperation among crowdsensors, we introduce a context-aware and cloud-less collaboration strategy in which crowdsensor groups are maintained in an autonomous and distributed way to monitor a physical phenomenon of interest. It divides a crowdsensing task into sub-tasks following the principle of teamwork. The context such as user activity (e.g., static *vs* mobile) and the physical environment (e.g., in-door *vs* out-door) are used to create homogeneous groups constituted of crowdsensors that tend to stay together. Indeed, grouping together the crowdsensors that are co-located and that behave alike facilitates the collection of measurements that relate to a shared physical phenomenon and that henceforth can be aggregated locally. In order to optimize the task assignment and to ensure that only relevant group members perform sensing, the crowdsensor contexts such as in-/out-pocket, the sensor accuracy, and remaining power are used to estimate the utility of the crowdsensor, which defines to which extent a member can provide the service. Overall, our group-based strategy introduces a context-aware clustering strategy and task allocation scheme that enhances the local and global crowdsensing efficiency. We provide an evaluation of the proposed solution using analytic evaluation, implementation, and simulation on dataset. The results support the adoption of context-aware collaboration for opportunistic crowdsensing by achieving quality/cost-efficiency compared to the cloud-based architecture.

- ***IAM*: data processing at the edge -** The collaborative data processing at the edge leverages smartphones' computing power to enhance the knowledge before transmitting to the cloud. The final step of this thesis is focused on the pre-processing of the crowdsensing data to reduce the data uploading and resource consumption on the cloud. Indeed, large-scale crowdsensing usually involves significant communication, computation, and financial costs due to the dependence on the cloud for the collection and post-processing (i.e., fusion, interpolation, and aggregation) of a vast amount of raw sensing data.

  As an alternative, we investigate a distributed data processing approach to pre-process the sensing data at the very edge, thereby reducing the resource consumption and enhancing the relevance of the knowledge collected on the cloud. As a result, part of the work that is traditionally carried out by the cloud server is transferred to the ubiquitous crowdsensors. We introduce a distributed interpolation-mediated aggregation approach running on the crowdsensors. To achieve so efficiently, we model interpolation as a tensor completion problem on each crowdsensor and propose a tensor-wise aggregation following an opportunistic relay process. The aggregation is based on opportunistic meetings, and the relay decision is made based on the quality of the inferred data. The solution results in shifting the cloud-centric approach to a distributed interpolation and aggregation on the move based on crowdsensor P2P (Peer-to-Peer) meetings. Not only the cloud resource is saved but also D2D relays significantly replace the crowdsensor uploading. The evaluation using real-world crowdsensing datasets shows that our solution allows significant savings in terms of cellular communication and cloud computing, and therefore financial costs. Furthermore, the overall data accuracy remains comparable to that of the cloud-centric approach.

This thesis follows the following structure: Chapter 2 surveys the background and existing work of efforts on increasing the quality and decreasing the cost, while only few work have proposed collaborative crowdsensing; Chapter 3 presents our contribution to user-centric context inference for crowdsensing and shows the accuracy enhancement; Chapter 4 introduces our work on context-aware collaborative crowdsensing group and shows the quality/resource benefits of such collaboration; Chapter 5 illustrates our proposal on collaborative interpolation and aggregation at the edge. It shows the benefits for the crowdsensor and the cloud; finally, Chapter 6 concludes this thesis, points out remaining challenges, and outlooks the future work.

1.3.1 Publications

The contributions of this thesis are published in the following papers:

- *User-Centric Context Inference for Mobile Crowdsensing.* **Yifan Du**, Valerie Issarny, and Francoise Sailhan. In Proceedings of the ACM/IEEE Interna-

tional Conference on Internet of Things Design and Implementation (IoTDI 2019).

- *When the Power of the Crowd Meets the Intelligence of the Middleware: The Mobile Phone Sensing Case*. **Yifan Du**, Valerie Issarny, and Francoise Sailhan. ACM SIGOPS Operating System Review (OSR), July 2019 Issue.
- *Let Opportunistic Crowdsensors Work Together for Resource-efficient, Quality-aware Observations*. **Yifan Du**, Francoise Sailhan, and Valerie Issarny. In Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom 2020).
- *In-network Collaborative Mobile Crowdsensing*. **Yifan Du**. In Proceedings of the IEEE International Conference on Pervasive Computing and Communications Ph.D. Forum (PerCom 2020).

Furthermore, the following paper is under submission:

- *IAM: Interpolation and Aggregation on the Move - A Collaborative Crowdsensing Approach for Spatio-temporal Phenomena*. **Yifan Du**, Francoise Sailhan, and Valerie Issarny.

Chapter 3	*User-Centric Context Inference for Mobile Crowdsensing*; *When the Power of the Crowd Meets the Intelligence of the Middleware: The Mobile Phone Sensing Case*
Chapter 4	*Let Opportunistic Crowdsensors Work Together for Resource-efficient, Quality-aware Observations*
Chapter 5	*IAM: Interpolation and Aggregation on the Move - A Collaborative Crowdsensing Approach for Spatio-temporal Phenomena*
Overall thesis	*In-network Collaborative Mobile Crowdsensing*

Table 1.1: Thesis chapters and related papers

We note that Chapters 3 to 5 may be read independently, presenting the related work and from design to prototype implementation and evaluation of the specific contributions. As such and as shown in Table 1.1, the contributions presented in Chapters 3 to 5 are also available through the originally published/submitted papers mentioned above.

Chapter 2

Background

Contents

2.1 What does Crowdsensing Sense?

What crowdsensing can sense is the key factor that determines its added value. We classify the sensing data gathered from crowdsensors into two types: (i) the subjective knowledge requested from the people, e.g., their views, opinions, and observations; (ii) the objective phenomena existing around the device, e.g., sensor readings, device status, and user contexts. Sensing both people and things is essential for the development of smart cities.

2.1.1 Knowledge from the Crowd

This type of crowdsensing scheme requires people to give their own opinions and to report some information about their observations, as the sensing data. Typical applications include [25]:

- Prevention of emergencies, which requires the volunteers to provide a summary of the situation. For example, the volunteer may take some pictures of waterways and report the amount of water in the riverbed to aid water management programs [104].
- E-commerce, which requires the clients to look up the products and share the product information. For instance, people can take a picture, report the fuel price, and receive a live-comparison among gas stations in turn [23].
- Well-being improvement, which needs people to exchange their suggestions or views about life. For example, people can record and share their experiences with diets and fitness [164].
- Social networks recommendation that extracts information from what people post on the Internet. For instance, the objective may be to share experiences and media (e.g., photo and video) among users with similar interests [16].
- Public safety evaluation, which requires volunteers to assess and report the safety situation. For example, citizens can check and share the level of crimes for their residential area [15].
- City waste management that requires citizens to monitor and help waste-recycling operations. For instance, gathering real citizen needs (e.g., amount of trash) for waste collection routing [19].

Although some of the applications mentioned above can be recognized as no burden for the people, they still need to take some actions, e.g., people post on Twitter at their will. People are involved in this kind of sensory. Obviously, participatory crowdsensing is a suitable way to collect knowledge from the crowd. People contribute to these sensing data through some incentive mechanism or because they are willing to share information or just altruism. The pros of this kind of sensing scheme are that the intelligence of people is leveraged. The cons are that the sensing at a large scale can be a challenging burden since people should be rewarded to ensure active participation.

2.1.2 Phenomena around the Crowd

This type of crowdsensing scheme automatically collects the data generated by the sensors embedded in the mobile devices attached to people. The collected data are further processed on the cloud without involving the user. Unlike the knowledge

from the crowd, the burden is usually put on the platform (application and server) rather than on people to mine the data. Typical applications include [25]:

- Post-disaster management, which leverages the embedded sensors on mobile devices to report and infer the situation. For example, the motion sensors of many people reporting the same vibration indicate an earthquake [58].
- Monitoring environmental conditions, using the sensor embedded in or attached to mobile phones. For instance, air pollution requires an external sensor and the noise pollution uses the microphone only [74].
- Health care systems, which infer the physical conditions of people based on their embedded sensors readings. For example, human activity recognition and monitoring are useful for remote feedback and diagnosis [76].
- Indoor localization and navigation using location-dependent fingerprints in environments lacking GPS. For instance, leveraging the received signal strength of Wi-Fi, magnetic strength, or luminous conditions [208].
- Intelligent transportation systems, using features extracted from mobile phone sensors, locations, and available network status. For example, estimating trajectory and travel time along the route as a public transport service [175].
- Interaction with unmanned vehicles, which requires the help of ubiquitous sensors from the crowd. For instance, the collection of sensing or location data from mobile devices for driver-less aerial vehicles [223].
- Experience-based decisions for urban planning that leverages embedded sensors on mobile devices. For example, monitoring the bridge vibrations using some smartphones' accelerometer on the move [140].
- Urban network characterization, which is based on the network property of mobile devices. For instance, Wi-Fi coverage mapping leverages interference power, wireless spectrum, and received signal strength [57].

In general, people are unconscious of this sensing, which is autonomously running in the background. The device observes autonomously without requiring any actions from people. Sensing data is provided by the device readings rather than human and is thereby objective. Even the behaviors of people can be considered as phenomena. This method leads to better scalability for ubiquitous sensing because it does not put a burden on the end users; thus, more people are expected to participate. The pros of this kind of sensing scheme are that people do not need to be requested or disrupted. The cons are that the raw sensing data requires analysis or mining; thus, post-processing is the key to extract valuable information.

2.1.3 Contribution to Smart City

It has been shown in the previous section that most of the phenomena that are observed by the crowd are relevant to smart cities. Particularly, we believe that crowdsensing, especially opportunistic crowdsensing, is the right way to support a variety of application domains, such as, e.g., environmental monitoring, healthcare, urban transportation, location services, social recommendation [216].

Ubiquitous sensing service is an essential enabler of the smart city concept. Nowadays, smart city development requires the involvement of citizens to enhance the transport and services delivered to the community as well as the health and well-being of citizens [89]. The sensing service has become a crucial part of smart city development. The IoT paradigm has been generating a lot of sensed information to support the city's administration and services for the citizens [217, 155]. Smart cities should be equipped with numerous kinds of sensors and actuators to monitor various aspect (e.g., the structural health of buildings, air quality, noise, traffic congestion, city energy consumption), detect and manage waste, enable smart parking, smart lighting as well as automation and salubrity of public buildings, just to name a few. Some of these sensing applications require deploying specific and professional IoT devices (sensors and gateways), while the rest can be performed through crowdsensing, especially those phenomena associated with people. Ubiquitous sensing enabled by WSN and crowdsensing technologies together offers the ability to measure, infer, and understand environmental and urban phenomena [65].

The urban environment is an essential characteristic of the smart city. Opportunistic crowdsensing naturally supports such monitoring as the target is an objective phenomenon. The urban environment knowledge informs actions regarding issues as diverse as, e.g., public health, city planning, intelligent transport system. Environmental monitoring benefits from the concept of crowdsensing, since it produces a huge amount of sensing data on an urban scale pervasively [145]. In practice, opportunistic crowdsensing is a scalable and cost-effective alternative to deploying static WSN for dense sensing coverage across large areas [178]. With the increasing variety of sensors embedded in smart devices, crowdsensing also becomes a convenient platform that enables a broad range of environmental monitoring. For example, (small) external sensors connected to smartphones are useful to monitor air pollution [51, 74, 176]. Similarly, specific external equipment supports low-level radiation detection [204]. Meanwhile, the smartphone's sensors may also enable a variety of applications. For instance, the smartphone's camera and its flash provide valuable information to particulate matter dosimeter [22]. The quality of the GPS signal that penetrates the ionosphere permits tomographic analysis of the global ionosphere [150].

Unlike temperature or air quality sensor, the microphone is available on all mobile phones. Thus, a body of research work has focused on leveraging crowdsensing for monitoring noise pollution over large-scale urban areas [221]. In order to act as a sound level meter, the microphone records a short sound clip, and A-weighting is usually applied so as to provide a sound level pressure expressed in dB(A). For

instance, the work in [82] introduces a mobile crowdsensing application that monitors urban noise and reports on the empirical analysis of this crowdsensing application from both technical and social perspectives face of a large and highly heterogeneous population of participants. Other efforts propose to monitor the city noise using crowdsensing smartphones [210, 64, 159, 170, 162, 177, 137]. Notably, we note that opportunistic crowdsensing enables (i) the monitoring of the noise pollution, and, (ii) the generation of the noise maps of the city. The work in [132] goes one step further; it exploits an audio event classification system to label the sounds that are known and discover novel types of sound.

In summary, mobile crowdsensing has many potential applications in various domains. The different features of participatory *vs* opportunistic crowdsensing determine their respective targets, which lay in collecting people's opinion *vs* sensing the objective phenomena. Opportunistic crowdsensing does not put a burden on the people as it often senses the phenomenon using the device readings. Opportunistic crowdsensing is a scalable and pervasive sensing mechanism that supports environmental monitoring, which is an essential service to be delivered within smart cities. In this thesis, we mainly focus on the environmental monitoring scenario to support smart city development.

2.2 The Way to Efficient Crowdsensing

The efforts undertaken to develop participatory and opportunistic crowdsensing applications focus mainly on three aspects: context-awareness to provide meta-information for more insight about the sensing data (Section 2.2.1); the efficient management of crowdsensors, such as participants recruitment (Section 2.2.2); the processing of the crowdsensed data, such as relay-based uploading (Section 2.2.3).

2.2.1 Context-awareness

Context-awareness for mobile devices relates to applications like localization, movement and activity tracking, and environmental sensing [27]. Context-awareness is essential for mobile crowdsensing because it provides the capability of being conscious of physical environments or situations around the crowdsensor. Thus, it allows the crowdsensors to work proactively and intelligently based on such awareness. For instance, the context can be used to predict a crowdsensor's responsiveness to notifications for greater worker engagement [95]. As an example of context, activity recognition is the most studied topic for crowdsensing [215, 218]. First of all, the context must be defined, and it depends on the application that is the target of crowdsensing. Then, the context of crowdsensing needs to be extracted, usually conducted using data mining techniques. Finally, the context information needs to be leveraged in some way to enhance crowdsensing efficiency.

Formally, the various contexts of mobile crowdsensors include:

Temporal contexts	Day, week, month, season, year, etc.
Spatial contexts	Area of interest, cell ID, latitude, longitude, altitude, etc.
Physical contexts	Placement, temperature, noise level, light intensity, traffic conditions, etc.
User contexts	Profile, activity, behavior, neighbors, social network, etc.
Device contexts	Internet connectivity, sensor accuracy, battery, communication cost, computing power, etc.

Table 2.1: Contexts of mobile crowdsensors

Among the content of Table 2.1, the first two contexts are usually essential as parts of the sensing data. In environmental monitoring, the physical context is the sensed phenomenon. The other contexts are often used to support the decision making of sensory: crowdsensors are naturally associated with heterogeneous devices and diverse people; knowing these contexts does not only gives more insight to classify the observations but also helps to allocate the resource. The same applies to IoT systems for which the sensor selection is a critical requirement [154]. Generally speaking, selective sensing holds when there is access to a large number of sensors with overlapping and redundant functionalities.

By leveraging context information, the crowdsensing application can be enhanced in terms of both the quality of the delivered data and the cost of sensing:

- *Context-awareness to improve the data quality.* The paper [161] classifies the sensing contexts of crowdsensors into two groups: on-hand and in-pocket (or bag), such that the crowdsensor senses only when the device is on-hand. The work in [128] trains a context-data quality classifier, which estimates the data quality from the context (e.g., user activity), such that only high-quality crowdsensors are recruited. The paper [50] also considers contexts (e.g., proximity and phone call state of the crowdsensor) to decide whether the environmental measurement (noise) is inaccurate and should be discarded. In general, device placement is the aspect considered to filter out inaccurate crowdsensing measurements.

- *Context-awareness to reduce the sensing cost.* The work in [28] considers context like geographical, temporal, demographics, and user activity for task assignment, that is, to allocate the right tasks to the right users in the right circumstances to preserve mobile device resources. The on-demand and selective sensing in [153] allows clients to combine queries for their interests, leveraging contexts like location and user activity, to reduce the energy consumption, network communication, and storage requirements. The paper [75] considers more contextual parameters, including the spatio-temporal aspect (e.g., location and time), user information (e.g., gender, age, and activity) and

device characteristics (e.g., battery level, built-in sensors). It smartly recruits crowdsensors and subsequently allocates tasks to improve energy efficiency. Furthermore, the work in [190] defines a semantic context model for query to select appropriate crowdsensors, not only to help task creator to define recruitment constraints but also to select labors more likely undertake a crowdsensing task. The key fact is that context-awareness allows performing crowdsensing tasks in an explicit and selective manner.

This thesis is inspired by previous work, and we believe that context-awareness should be delivered to collaborative crowdsensing as well. The reason is that the context information beyond location and timestamp does not only give more insight into the crowdsensing data but also is useful to determine which crowdsensors should collaborate and how crowdsensors can collaborate with others. In participatory crowdsensing, context is mostly used to support semantic querying and quantitative reasoning [98], i.e., contextual information related to each sensor is used to select suitable participants. Similarly, in opportunistic crowdsensing, the context plays a crucial role in achieving efficient collection, including sensor selection and task assignment. In particular, context-awareness is important for opportunistic crowdsensing because the sensing data collection is fully autonomous and is done without (or with minimal) active user interaction. While it goes with collaboration, context can determine how the collaboration should be further performed.

The term "*context*" is very general, and the context categorization may include many factors. The impact of context depends on the application scenarios and the selection of it as well. Unlike previous work that uses context for a semantic query, we aim to support the context-aware and opportunistic crowdsensing of the urban environment. Precisely, we take into account three types of context: the device attribute (e.g., network connectivity, communication cost, and computing resources), the user characteristic (e.g., neighbors, activities, behaviors), and the physical environment (e.g., in-pocket, in-door, under-ground). Some context information comes from machine learning inference, and we specifically deal with the quality/accuracy of context inference (see Chapter 3). Then, the context is leveraged for determining the collaboration behaviors among several crowdsensors at the very edge.

2.2.2 Crowdsensing Management

One reason why crowdsensing is very challenging is that it is not the same as an infrastructural IoT system, where wireless sensors are manually deployed and controlled by a server/host, and sensors are professional quality. Instead, crowdsensing end devices are naturally heterogeneous and uncontrolled. Crowdsensing management involves three aspects: user recruitment, task allocation, and incentive mechanism. The objective is to perform a sensing campaign while minimizing the cost, which necessitates ensuring a spatio-temporal sensing coverage and accomplishment of the related crowdsensing tasks under (minimal) budget constraints, while effectively incentivizing the crowdsensor participation (in the case of participatory

crowdsensing). In the following, we detail crowdsensing management:

- *User recruitment* selects crowdsensors according to the sensing tasks. By leveraging the probabilistic mobility model that estimates the displacement of crowdsensors, the work in [71] limits the participation of redundant crowdsensors depending on the required service. The paper [213] predicts the users' mobility so as to select a subset of participants that should visit some points of interest. Based on predicting call probability, the work in [118] recruits the minimal number of participants necessary to get a certain level of coverage and support real-time, spatio-temporal crowdsensing tasks. The paper [123] also selects crowdsensors based on the desired quality of information (that quantifies the data granularity) as well as the number of tasks that are necessary while achieving energy efficiency by selecting minimal participants. The recruitment model [14] can also select once the most appropriate group of participants using criteria related to the user's area-of-interest and device characteristics. While the crowdsensing tasks are heterogeneous and dynamically performed, the recruitment method [117] proposes both offline and online algorithms based on the prediction of call probability, to minimize the number of participating crowdsensors and maintain a satisfying level of coverage. The work in [188] selects a subset of users on the social network as initial seeds and allocate crowdsensing tasks to them. Then, influenced users should accept and propagate it to friends, and the ultimate goal (which is to maximize the coverage) will be met.

- *Task allocation* aims at assigning the tasks that are executed on crowdsensors. Although the existing work of participatory crowdsensing often addresses the task allocation and participant recruitment problems together (i.e., they recruit users to perform tasks), task allocation can still be handled as a separate problem. A task allocation framework may aim at maximizing the overall system utility in a centralized way by coordinating the allocation of multiple heterogeneous tasks while considering crowdsensor-side factors, including user bandwidth, user availability, devices' sensor configuration, task completion likelihood, and mobility pattern [189]. When crowdsensors move over time and tasks arrive in a stochastic way, the location-based task allocation becomes dynamic. Fairness is required to maintain system stability and achieve a sensing utility close to the optimum [198]. For the piggyback crowdsensing approach, which is power-saving, the crowdsensing task allocation for participants also introduces an optimization problem subject to incentive budget and spatio-temporal coverage constraints [206].

- *Incentive mechanism* controls the trade-off between contributing crowdsensors and their rewards. There are diverse strategies that have been proposed to provide incentives for stimulating users to participate in mobile crowdsensing applications [220]. The work in [152] estimates the quality of sensing data and

pay the participants a reward based on their useful contribution, to motivate the rational participants to perform data sensing efficiently. Leveraging the game-theoretical reverse auction mechanism, the paper [165] designs a framework that encourages the crowdsensing user participation with uncertain mobility. Based on reverse combinatorial auctions from game theory, the work in [87] incorporates the quality of information on crowdsensors as metric and builds an incentive framework for mobile crowdsensing systems. The paper [60] randomly recruits crowdsensing participants and uses a quality-aware incentive mechanism to maximize the amount of high-quality sensing data under a limited task budget.

Incentive mechanism is particularly important in participatory crowdsensing since people need to take action and to should be involved in the sensing. Participants can be paid with money (payments for their contributions), with a form of entertainment (sensing tasks are turned into playable games to attract participants), or by a service given in exchange (for mutual benefits). In order to encourage crowdsensors to contribute and limit the budget, most incentive approaches are based on game theory. The incentive mechanism is the key to achieve the effectiveness of participatory crowdsensing, especially with regards to the budget.

User recruitment and task allocation attract comparatively more attention in opportunistic crowdsensing, where participants are self-motivated and altruistic. Since opportunistic crowdsensing shifts the burden of collecting sensing data from users to the application, we especially need to deal with the crowdsensor selection and task assignment more effectively. Generally speaking, depending on the mathematical model devised for the system, tasks assignment to (recruited) users is usually an NP-hard optimization problem, in which a cost function associated with the crowdsensing tasks is minimized, subject to some constraints. Overall, the proposed solutions differ in terms of cost functions and constraints, while assuming that the cloud maintains a global knowledge about each crowdsensor to select appropriate crowdsensors and assign them some crowdsensing tasks.

This thesis addresses the opportunistic crowdsensing efficiency challenge. Although the similar principles of crowdsensor selection (user recruitment) and task allocation may apply, they need to be adapted. We propose to perform crowdsensing management at the edge, rather than on the cloud. It is worth mentioning that our collaborative crowdsensing management will be fully distributed (see Chapter 4). In our case, the crowdsensor selection is essential to determine who will collaborate with whom. The task allocation determines the task(s) that every crowdsensor should conduct as a collaborative member. The collaboration requires an efficient resource negotiation during the sensing and reporting phases, considering various criteria such as, e.g., sensor accuracy, network occupancy, computational capabilities.

2.2.3 Data Processing

Following crowdsensing management, sensing data processing is also a pivotal aspect of effective opportunistic crowdsensing, as it aims to extract valuable information from the raw data. In practice, ubiquitous crowdsensors generate a massive amount of raw data, and the cloud is usually responsible for gathering and analyzing the data using, e.g., data mining techniques.

Efficient crowdsensing data processing often focuses on three aspects: data collection, data forwarding, and data analysis. The aim is to collect sensing data while minimizing the cost of crowdsensors, to upload the data to the cloud in the cheapest way, and to analyze the data for knowledge extraction. We detail the efficient data processing below:

- *Data collection* targets minimizing the sensing cost of the crowdsensor: For this purpose, the work in [133] applies sensors admission control and on-demand processing on the device to trade-off the performance need of application and the resource demand of continuous sensing on the opportunistic crowdsensors. The idea is to adapt the sensor pipeline, which judiciously triggers power-consuming stages, taking into account the user's mobility and behavioral patterns to reduce energy costs. Considering data collection utility and smartphones' potential sensors to gather information, the cloud-based mobile crowdsensing system presented in [26] minimizes the cost of both sensing and reporting, while maximizing the quality of contributed information. The well-known publish/subscribe model is also introduced in cloud-based crowdsensing that continuously selects the k-best sensors for a sensing task [139]. It controls the collection by eliminating redundant sensor activities while satisfying sensing coverage requirements and sensing quality, thus consequently reduces the overall energy consumption.

- *Data forwarding* aims to reduce the Internet communication cost for the crowdsensor. The work in [110] performs piggyback crowdsensing to save the energy consumption. It uploads sensing data to the cloud by exploiting and predicting those times when the crowdsensor uses mobile applications or makes phone calls. Then, to save the energy and budget associated with the data uploading, the paper [196] adapts the uploading scheme, considering different network types and predicting the future user's position. It chooses the appropriate condition to either offload data to Bluetooth/Wi-Fi gateways, or to unlimited data plan users. The work in [33] only uses a Wi-Fi network for uploading, which requires to forecast when the Wi-Fi network is available, and when no front-end applications are using it, to minimize the overall energy consumption and mobile data cost. To reduce the mobile data fee, the paper [195] distinguishes pay-as-you-go and unlimited mobile data plan to take (delay-tolerant) uploading and forwarding decisions. It predicts the mobility pattern of crowdsensors and estimates the size of the sensed data to partition the users into the two types.

- *Data analysis* extracts out useful information from raw sensing data: The work in [105] collects large-scale continuous crowdsensing data about human behaviors in a non-intrusive manner. It uses multiple embedded sensors to analyze modalities, such as social interaction and spatial behaviors, which is a productive means to study location attributes. The paper [143] uses continuous crowdsensing data to determine "what is going on" around the user in real-time, by exploiting multiple sensor readings. Thus, it explores living points of interest of the city by determining real-time hot-spots from sensor data, and automatically estimates user's points of interest. The work in [38] exploits opportunistically captured locations, user trajectories, images, and audio clips together from crowdsensors to classify visited places within place categories, for automatic and scalable semantics of places. The paper [172] even runs real-time data stream mining like a lightweight classification algorithm running on mobile devices. It correlates sensing data with social media and provides accuracy comparable to those on the cloud while reducing the amount of data uploaded and energy usage.

Overall, a majority of work adapts the data collection and forwarding for opportunistic crowdsensing. So far, the proposed collection approaches tune the sampling frequency, allocate multiple sensing tasks, and even filter data according to the sensor type. The forwarding scheme is mostly delay-tolerant and thereby relies on caches to temporarily store the sensing data. In contrast, for real-time uploading, the sensing data needs to be transmitted immediately. These sensing mechanisms are application-specific; they should apply customized parameters depending on the sensing target. For instance, the sensing resolution in both time and space needs to be carefully defined for environmental monitoring. Also, data collection and data forwarding should be controlled by the crowdsensing platform, which allocates the sensing tasks and dynamically adapts the uploading strategy.

Most importantly, the data analysis mentioned above consists of applying data mining methods to the sensing data collected to extract various information related to, e.g., citizen behaviors and points of interest, or even social networks. Such data analysis is typically performed on the mobile device and on the cloud, which processes a vast amount of raw crowdsensing data. In particular, the application determines the sensors that are enabled on mobile devices as well as the computing resource allocated on devices and clouds. Since we focus on environmental monitoring, the ultimate aim is to create a spatio-temporal map of the city environment based on the observations provided by crowdsensors. The map requires storing the spatio-temporal crowdsensing data and analyze the data on the cloud server periodically.

This thesis focuses on opportunistic crowdsensing for environmental monitoring, which shifts the burden from people to the middleware/application and requires data analysis afterward. Traditional opportunistic crowdsensing relies on the cloud server to process the ubiquitous and raw sensing data. We promote to process the data at a very early phase before the cloud, i.e., on the end devices together, not only to

control the collection, to determine the forwarding but also to perform data analysis among multiple crowdsensors as in a collaborative manner (see Chapter 5). To the best of our knowledge, distributed data processing for opportunistic crowdsensing is an open research question. In particular, we need to address the collaborative data analysis at the very edge.

2.3 Collaborative Crowdsensing

Most efforts on the development of crowdsensing systems assume that a cloud server monitors and, in some cases, controls every crowdsensor. Every crowdsensor relies on the cloud server for registration, sensing and uploading. The crowdsensor may collaborate with, e.g., a cloud server, an edge server, an intelligent gateway, or with other crowdsensing devices. We classify the collaborative crowdsensing approaches into two types: (i) infrastructure-based (Section 2.3.1) or (ii) crowdsensor-centric (Section 2.3.2). Furthermore, the collaboration can be managed either by the cloud or can be totally cloudless. In this thesis (Section 2.3.3), our solution refers to performing the collaboration with other crowdsensors via D2D communication, while assuming the cloud only provides essential backend support if needed.

2.3.1 With the Infrastructure

The collaboration with an infrastructure still assumes that each crowdsensor works individually for sensing, location tracking and data uploading, but is managed by and reports to the infrastructure. Such a collaboration necessitates a local gateway/access point connected to the crowdsensors that may provide access to an edge/fog server.

Collaboration with the edge requires the local server/gateway to be available. For the collaborative uploading, the mobile device senses and caches the sensing data, which is opportunistically transmitted when the device gets into the communication range with a nearby Wi-Fi access point. Such a crowdsensing solution supports delay-tolerant reporting of the sensed data [136]. The uploading can be scheduled in advance by forecasting the Wi-Fi access points that may be encountered by the crowdsensor, to avoid using cellular networks [33]. Besides, an edge/fog server can be deployed to assist mobile crowdsensors. Such a server then recruits local/nearby crowdsensors, collects and filters the duplicated sensing data, and besides pre-processes the data [148]. In the work [158, 11, 10], an edge server manages the crowdsensor resources and acts as a broker of the publish/subscribe system. The broker at the edge filters and aggregates the sensing data published by the nearby crowdsensors and forwards the aggregated data to a back-end subscriber (i.e., server on the cloud). Brokers can be organized into a hierarchy constituted of nano-, local-, and public-cloud servers. The hierarchy allows the crowdsensors to collaborate with the mobile broker that aggregates the information from local region [168].

Overall, such collaboration involves gateway/access point and an edge/fog server that uploads and pre-processes the data, which are provided by the crowdsensors that work individually (i.e., sense, keep track of the context and report). The primary advantage of leveraging edge/fog servers is to provide low latency and fast response to end devices while offloading cloud resource consumption. Most edge computing proposals support a collaboration with the edge infrastructure but very few promote a direct and autonomous collaboration among crowdsensors without e.g. edge/fog server, which is referred as edge computing in a distributed and *ad hoc* manner.

2.3.2 With other Crowdsensors

Collaboration with other crowdsensors finds two different implementations, depending on whether it is managed (i) by the cloud, or (ii) in a Device-to-Device way:

- *Cloud-based Collaboration*: This kind of collaboration among multiple crowdsensors assumes that the cloud controls/manages/assists every crowdsensor. Typically, collaboration occurs between people who are geographically close to each other or who hold a close relationship and thereby belong to the same community/organization. If the moving trajectory of each crowdsensor is known in advance, the cloud server can arrange the sensing scheduling of multiple crowdsensors, to avoid duplicate sensing in the same area of interest and reduce the overall sensing energy consumption [171]. The work in [109] detects the people associated with a social network to form crowdsensing communities, which are composed of people sharing similar tastes, preferences and making the same choices, and which are henceforth likely to cooperate. The existence of nearby crowdsensors can even be leveraged to analyze the phenomenon. In the work [201], probe smartphones scan neighboring smartphones and report the result to the cloud, which estimates the crowd density using several features (e.g., number of scanned neighbors, Bluetooth signal strength). Another work [39] uses the microphone of the smartphone to determine the acoustic context of several adjacent crowdsensors. Then, data mining algorithms running on the cloud identify groups of people that speak and interact.

- *D2D-based Collaboration*: This kind of collaboration is distributed and works at the very edge. Most D2D collaborations among crowdsensors involve a delay-tolerant data relay and/or a D2D network. Collaboration requires a physical proximity (i.e., crowdsensors should be within D2D communication range). Proposed approaches are opportunistic: pedestrians walk together, opportunistically form a D2D network and start the collaboration. In order to reduce the upload rate of each pedestrian, the work in [182] proposes to elect a leader that acts as manager and as a network proxy that uploads crowdsensing data. Groups of pedestrians are detected in [78] based on the history of radio

connectivity between the mobile devices, and, clusters of nearby members are maintained for efficient information diffusion. By leveraging the predictability of encounters and mobility of people, the papers [194, 196] propose an adaptive uploading scheme that makes the D2D forwarding decision following fixed data uploading cycles. Crowdsensors and other mobile devices from a local community may share network access and transfer data for each other [44], by selecting and exploiting multiple mobile hot-spots in the vicinity. The work in [187] estimates the probability distribution of user arrival at points of interest and the inter-user contact probability to recruit mobile crowdsensors that will relay the data of others, for the purpose of minimizing the uploading cost. Crowdsensors can even collaborate with wireless sensor networks. The paper [181] alleviates the load of the most queried WSN nodes by offloading a portion of the traffic to nearby related crowdsensors to save energy. The positioning can also be shared collaboratively. The work in [108] aims to minimize the total number of location measurements for a given fairness criterion to finally reduce the average power consumption of mobile devices. In the P2P-based crowdsensing architecture proposed in [86], the sensing data is locally stored in and processed by mobile devices and is shared among users in a quality-aware data sharing market, to avoid high operational cost on the centralized server for storing and processing massively. Collaborative crowdsensing framework may capture and share sensed data between multiple distributed applications and users, to facilitate the development and deployment of opportunistic sensing applications [84].

The collaboration via the cloud requires tracking all users and updating in real-time the related information while people move. Such a cloud-based management of collaboration provides a global view on the crowdsensors anytime, anywhere, to support an optimal but resource-intensive control of the crowdsensors.

The D2D (or P2P) collaboration, which is naturally in-network and at the edge, is more distributed and scalable. Still, collaboration strategies have to be adapted to the specific application and framework. For example, the D2D sensing data offload has already been widely studied in WSN. Differently, in the case of crowdsensing, human factors, e.g., human encounters, device heterogeneity, and incentive mechanism, must be accounted for. While more factors should be considering and leveraged, some functionalities are missing: internet traffic offloading, location-sharing and data processing are needed. We should address them together, considering the characteristics of crowdsensors, especially the increasing computing power of mobile devices.

2.3.3 Our Solution at the Very Edge

We have stated that opportunistic crowdsensing is an effective way to achieve ubiquitous sensing for phenomena around the people, such as environmental monitoring of smart cities. We have shown that efforts have been made to improve the

crowdsensing efficiency in three aspects, including context-awareness, crowdsensing management, and data processing. We have found that although collaborative crowdsensing has been proposed to enhance the crowdsensing efficiency, it remains constrained since most works are still centralized or semi-distributed and focus on energy fairness.

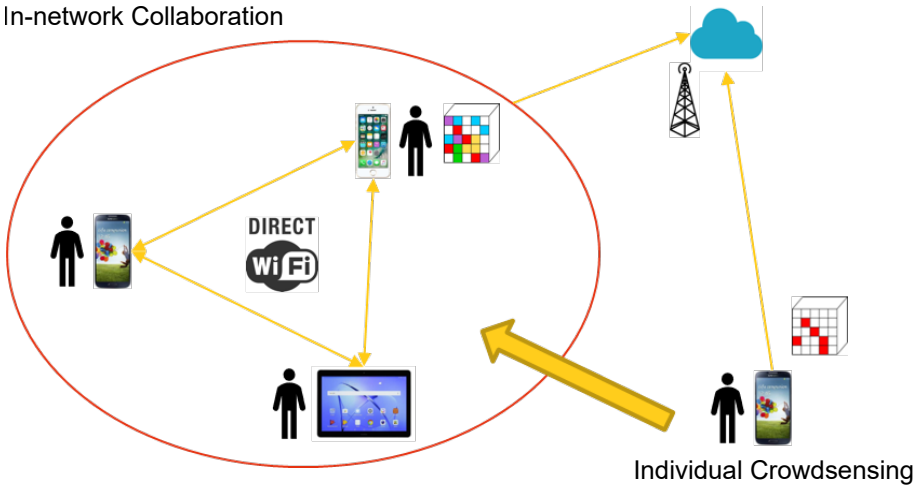


Figure 2.3.1: From cloud-centric to collaborative crowdsensing at the edge

In this thesis, we propose a collaborative crowdsensing system that opportunistically operates at the very edge (Figure 2.3.1). Such collaboration takes place among mobile crowdsensors leveraging opportunistic encounters. From the management standpoint, the architecture is ad hoc, fully distributed, and cloudless (in particular, the cloud does not allocate tasks). It neither requires the additional deployment of edge/fog servers to monitor and manage the local areas.

We address the challenge of efficient opportunistic crowdsensing through such a collaboration, for which our solution distinguishes from the state of the art on crowdsensing for environmental monitoring by featuring three complementary contributions:

- *On-device context-awareness* - The context beyond the temporal and spatial information of each crowdsensor is useful to 1) explicitly describe the sensing data, 2) manage the collaborative group and 3) support task assignment, among several nearby crowdsensors. Meanwhile, the heterogeneous context inference accuracy of each crowdsensor is taken into account. *ContextSense* provides such crowdsensing context.

- *Context-aware and distributed crowdsensing management* - The collaboration mechanism is: 1) based on context-awareness, and 2) autonomous and distributed among multiple crowdsensors that opportunistically interact with each other. The collaboration is cloudless and independent of the infrastructure, with no involvement of any edge server. *BeTogether* establishes such ad hoc collaborative groups.

- *Server-less sensing data processing for spatio-temporal analysis* - Rather than only performing sensing, forwarding and uploading to the cloud, more complex collaborative processing is performed among crowdsensors during opportunistic encounters. More data analysis is shifted from the cloud onto many crowdsensors, allowing the cloud to execute only simple functions. *IAM* deals with such collaborative data processing.

In summary, this thesis investigates a set of collaboration mechanisms among crowdsensors that interact opportunistically, with the aim of increasing data quality while reducing the overall computation and communication cost of crowdsensing. The initial concern of this research is addressed by the multi-party and multi-hop calibration in [166], which introduces an opportunistic group calibration system that pervasively compensates the reading error of un-calibrated crowdsensors. This method of calibration is automated and operates in the background. Having taken inspiration from this work, we now propose our context-aware collaborative crowdsensing at the very edge.

Contribution \Target	End device data quality	End device sensing cost	Cloud data quality	Cloud sensing cost
ContextSense	✓	-	-	-
BeTogether	-	✓	✓	-
IAM	-	-	✓	✓

Table 2.2: The problems addressed in three contributions

As shown in Table 2.2, each contribution provides solutions to the corresponding problems of opportunistic crowdsensing, and together they support distributed collaborative crowdsensing at the end devices level. *ContextSense* provides contextual information that enhances the knowledge on sensing data at end devices. *BeTogether* distributes the workload among several crowdsensors to save resources at end devices and performs selective sensing to improve the knowledge gathered on the cloud. *IAM* offloads the data analysis from cloud to many end devices, thus reducing the cloud's resource demand and enhancing knowledge on end devices.

It is worth mentioning that *BeTogether* and *IAM* are utilized in different scenarios. Our collaborative crowdsensing at the edge builds upon short-range communication and human interactions. We classify user interactions as either *group-wise* or *pair-wise*. During daily life, people typically encounter others following two types of patterns. Some people stay near each others for an extended period of time; this is the case when e.g., several people work in the same building area during working days. Other encounters may occur very briefly, e.g., someone passes by another person walking on the street. A robust collaboration solution must take into consideration both situations. In order to deal with the two user behaviors, the collaboration schemes in *BeTogether* and *IAM* differ in terms of three following aspects:

- Crowdsensing phase: When several crowdsensors stay close together for a while, they sense and monitor the same phenomenon. In such a case, we apply group-wise collaboration among the co-located crowdsensors during the data collection phase (i.e., as part of the sensing process) so as to keep to a minimum the resources involved by the group members to e.g. sense, aggregate and upload the measurements. If two crowdsensors make only momentary contact, we may assume that the two crowdsensors have been sensing and monitoring different urban areas until they meet. Thus, we apply pair-wise collaboration during the data processing phase so as to support a post-analysis of the data that have been gathered during the (previous) data collection phase.

- Crowdsensing functionalities: The above perspective leads to the application of different crowdsensing functionalities. Collaboration during the data collection phase involves frequent reporting/uploading to the cloud, while collaboration during the data processing phase requires temporarily data caching for a subsequent analysis.

- Crowdsensing scale: Group-wise collaboration is utilized for sensing a dense and community-scale area, i.e., a group of crowdsensors are located in the same area of interest and monitor the same phenomenon. Pair-wise collaboration copes with sensing a sparse, urban scale area, i.e., crowdsensors are traveling across the city following different trajectories and continue monitoring.

In short, *BeTogether* benefits the overall end-device efficiency when operating with *a dense crowd on a community scale*. In other words, it is suitable for scenarios in which *many crowdsensors are co-located simultaneously*. *IAM* reduces the cloud cost of the crowdsensing platform which operates with *a sparse crowd at an urban scale*. Meaning that it is suitable for scenarios where *crowdsensors only encounter each other very infrequently*.

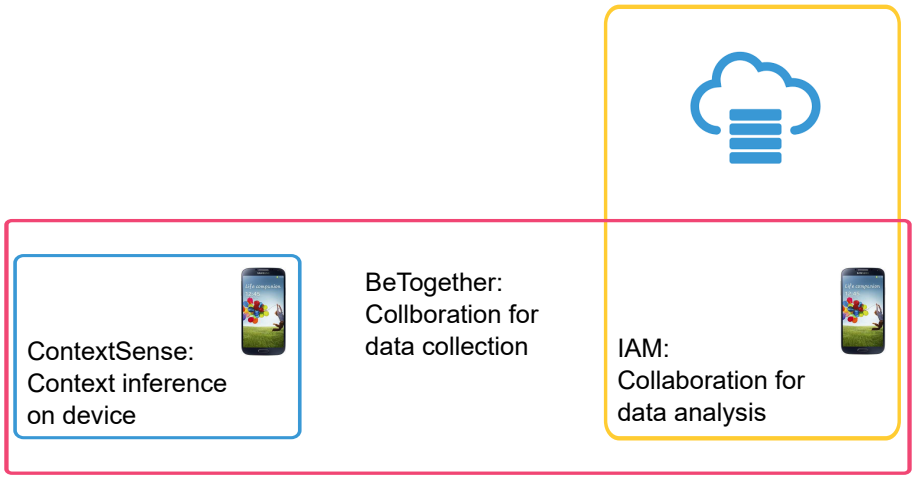


Figure 2.3.2: The three contributions complement each other

Before presenting the details of each contribution, we have illustrated in Figure 2.3.2 how the three contributions complement each other in terms of overall design and implementation: *ContextSense* is integrated as a component of the *BeTogether* middleware layer solution to provide parts of the necessary contextual information. In *BeTogether*, the aggregation of sensing data is a simple task, and each aggregation is followed by the uploading of data. On the other hand, more complex aggregations are addressed by *IAM* through the analysis of cached data. Note that while *IAM* still builds upon the framework of *BeTogether* in order to discover and exchange data, the management (group-wise *vs* pair-wise) and aggregation component need to be adapted.

Chapter 3

ContextSense: Knowing the Context of Crowdsensors

Contents

3.1 Context Inference

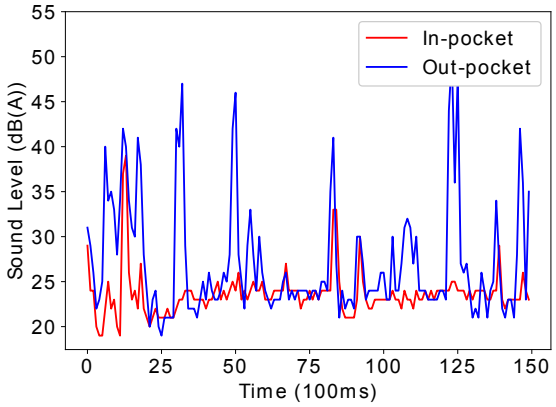
We argue that opportunistic crowdsensing empowers ordinary citizens to contribute to the environmental monitoring of smart cities [67, 126] because: (i) it allows ac-

quiring hyperlocal knowledge at scale, thanks to the proliferation of smartphones and the ubiquity of wireless connections, and (ii) numerous sensor types embedded in today's smart devices can provide quantitative observations about the urban environment (e.g., sound level, temperature, atmospheric pressure, humidity, illuminance). The observations further come with the related spatial and temporal information, which supports the analysis of overall environmental knowledge. Nonetheless, the major challenge facing effective crowdsensing is undoubtedly being able to collect data of sufficient quality, starting with the ability to characterize the provided observations. The quality of the contributed measurements challenges the aggregation of environmental knowledge.

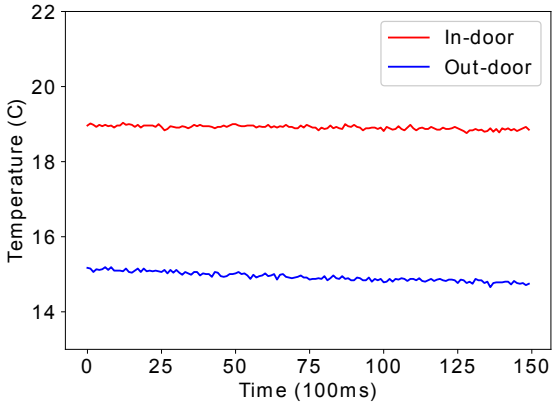
The data quality depends on both the *accuracy* [116] of the contributing sensors and the adequacy of the *sensing context* [215]. Addressing the former requires calibration [166], and the latter requires a supporting inference mechanism, which is the focus of this chapter. We posit the need for developing an intelligent middleware-layer solution to support context-aware crowdsensing. The collection of sensing data on the device must act beyond merely interfacing with the embedded/connected sensors to transfer the data to the cloud. As far as possible, the solution must enhance the quality of the observations locally, from calibration to contextualization. While calibration may be achieved through regression analysis [166], contextualization requires inference. The intelligent solution must implement soft/virtual sensors (as opposed to hardware/physical sensors) that run on the end device to analyze and mine the data provided by the ever-growing set of cheap embedded sensors.

The accurate monitoring of the physical environment through crowdsensing requires knowing the location of the contributed observations, but such contextualization is not sufficient. The user's activity impacts the quality of the quantitative observations that mobile crowdsensing gathers [128], which may be inferred from machine learning over motion sensor data [53]. The location must not be limited to the geographical coordinates in the Euclidean space. Indeed, the "user behavior at the location" has a significant impact on the quality of the quantitative observations contributed through mobile crowdsensing [128]. Knowing whether the smartphone (sensing device) is in-/out-pocket, in-/out-door, and under-/upper-ground is particularly important because the regular sensor needs to be in a position that enables –yet does not interfere with– sensing the physical characteristics of the surrounding [185]. The sensing context must distinguish between in-pocket and out-pocket observations. The former leads to a quite significant deviation from the ground truth and is thus not readily usable [161]. The same applies to in-door versus out-door measurements since aggregating them together to analyze environmental phenomena leads to unreliable results [138]. Similarly, under-ground and upper-ground scenarios contribute observations that must be distinguished.

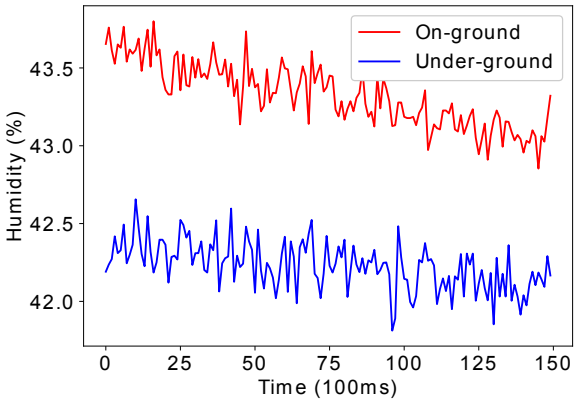
Figure 3.1.1 illustrates the impact (i.e., significant bias) of the above elements of the "sensing context" on physical measurements that are collected at the "same" geographical location. The context information allows keeping more observations –and even correcting them– for aggregating environmental knowledge, rather than



(a) In-pocket *vs* out-pocket sound level



(b) In-door *vs* out-door temperature



(c) Upper-ground *vs* under-ground humidity

Figure 3.1.1: Impact of crowdsensing contexts on environmental observations

filtering out drastically the crowdsensed data [82].

The current work on the inference of the smartphone context focuses on a single context element. At the same time, it is essential to characterize the sensing contexts accurately, that is, to identify whether a contributed observation is sensed with the device: in-/out-pocket, in-/out-door or under-/upper-ground. Machine learning is a candidate to systematize such a characterization. However, there is no single solution for the inference. The set of embedded sensors differs from one smartphone to another, and the characteristic of the user behavior impacts the inference on the observations. The challenge is then to ensure that the classifier accounts for the diversity of the crowd contributors. Indeed, there is a significant variation in the characteristics of contributing devices, the behavior of contributing users, and even the usage scenarios. A robust inference system must be able to evolve automatically depending on the sensor availability and the environment dynamics [200]. As a result, the classifier for context inference must be adaptive to each participating user, considering the user's behavior, device model, and contribution scenarios.

This chapter introduces our self-adaptive intelligent solution *ContextSense* for crowdsensing, which implements soft sensors that contextualize locally the collected observations. *ContextSense* leverages supervised and online machine learning so that inferring the context of the observations adapts to the available sensors and the sensing environment. *ContextSense* allows customizing the classifier on each user's device while minimizing the user's involvement, which is essential to motivate the engagement of a sufficiently large crowd. Our contribution is as follows:

- Starting from the sensors available on today's smartphones, we assess the particular relevance of the candidate features to characterize the three elements of the sensing context, to derive the best features that serve to classify each of them (Section 3.3.1).

- We then analyze the performance of candidate updatable learning algorithms to initialize the three resulting classifiers, taking into account their accuracy as well as their runtime and memory efficiency (Section 3.3.2).

- The personalization approach follows, which includes the hierarchical inference of all three context elements (in-/out-pocket, in-/out-door, and under-/upper-ground) that are relevant to environmental monitoring using crowdsensing, and the opportunistic update requiring very few feedback from the user (Section 3.3.3).

- We validate the effectiveness of the solution using a *ContextSense* implementation (Section 3.4) and simulation to assess the accuracy of the context inference with respect to the negative user feedback. We confront our hierarchical approach with a multi-class classifier, and we assess the energy and runtime efficiency (Section 3.5).

To the best of our knowledge and as surveyed in the next section, no other work on the inference of the sensing context addresses the classification of the three context elements simultaneously. They neither deal with the effective personalization of the classifier on the crowdsensing device.

3.2 Related Work

Inferring the sensing context related to the physical environment under scrutiny has deserved little attention, while related solutions focus on a single context element and consider different eligible features.

3.2.1 Features for Context Inference

Detecting whether a smartphone is in-pocket/bag or out-pocket/bag may be inferred using various embedded sensors. The work in [211] leverages features from embedded proximity and light sensor signals, while another work in [161] uses additional data from a 3-axis accelerometer. Inferring whether an observation is made in-door or out-door may also rely on various sensors. For instance, previous work in [161, 32] simply utilizes the GPS signal. Alternatively, the sensing service in [119] leverages three onboard sensing resources, including light sensors, magnetic sensors, and cell tower signals. A similar solution is adopted in [131], although using the proximity instead of the magnetic sensor. Another detection system in [120] uses two features, which are the received signals of Wi-Fi and the light density. The solution in [6] leverages these two features as well as the user's activity. At the same time, the approach in [197] exclusively uses radio signals and requires the signal strength of four neighboring GSM base stations. Comparatively, under-/upper-ground classification has deserved much less attention. Monitoring a quick change of the barometric pressure is suitable for detecting indoor floor transition [134] or an underground scenario because the air pressure allows estimating the altitude [80].

As outlined above, diverse combinations of embedded sensors may serve to characterize the context of crowdsensing, while the resulting inferences differ in their accuracy and energy-efficiency. The state of the art analysis further shows that the most relevant observations are related to the physical and communication environment. Still, it is critical to account for the variation in the availability of features across the contributing devices. Our approach addresses this requirement by being self-adaptive to the diversity of users (and devices). The user-centric classification on the user's device may be performed on any subset of the above features to account for the diversity of devices and related embedded sensors. The feature set may further be customized and freely recomposed to tradeoff energy efficiency and accuracy, thus also offers adaptiveness.

3.2.2 General Methods of Context Inference

Various algorithms have been considered for the inference of the mobile sensing context, among which rule-based solutions. A classifier based on the non-linearity of the sensor readings is presented in [211] for the in-pocket/out-pocket classification. Rule-based classifiers are also introduced for the in-door/out-door case but differ in the number of sub-detectors and their approach to aggregation. The algorithm in [119] uses three sub-detectors while the one in [120] uses two sub-detectors. The approach in [6] is also rule-based but hierarchical and switches among three rule-based modules to distinguish the contributing scenarios at fine-grain. Independent of the rule-based approach's specifics, adequate thresholds for the classifier must be set, which is done empirically [134, 131, 32]. An alternative to the rule-based approach for the context inference is to leverage machine learning algorithms. For instance, the work in [161] uses the KNN (K-Nearest Neighbor) algorithm with the trained classifier being evaluated using cross-validation. A range of machine learning algorithms for the classification of in-door *vs* out-door is investigated in [197], including the Decision Tree, Random Forest, Support Vector Machine, KNN, Logistic Regression, Naive Bayesian, and Neural Network; they conclude that KNN performs the best in terms of average accuracy.

The literature investigates the inference of each of these context elements separately, but they are all equally important. Moreover, the proposed solutions do not account for the diversity of the contributing devices. Further, they train and test the learning model using a single dataset. None of them addresses the classifiers' personalization to cope with the diversity of the devices, users, and scenarios. Differently, our solution solves three classification problems together and is implemented as an on-device service providing the sensing context to crowdsensing applications. We emphasize the customization of the context inference per user, which requires an online learning algorithm to keep the classifier evolving on each crowdsensor.

3.2.3 Towards Personalized Methods of Inference

Supervised learning for one analyzed environment does not always translate to another environment. Transfer learning [101] has been proposed to solve the problem, but it requires complete datasets, and deep learning is still too heavy for end devices. Otherwise, the baseline learning model needs to evolve according to the environment in which it is used. For instance, the user-specific touch input model in [199] is updated using calibration input requested to the users. The method for in-door/out-door detection in [160] employs semi-supervised machine learning without user involvement to learn new information. It uses co-training classifiers requiring two Naive Bayes models, resulting in a double computational cost on the device. Although the estimation of the class of a new instance does not involve the user, it is not the ground truth. Rather than excluding user involvement, we believe that the opportunistic participation of the user to gather the ground truth allows a more effective personalization of the classifiers.

We introduce three classifiers, *aka*, machine learning models, to comprehensively characterize the crowdsensor's context, that is, whether the sensing device is: in-pocket/out-pocket, in-door/out-door, and under-ground/upper-ground. We denote the corresponding classifiers M_{pocket}, M_{door}, and M_{ground}. We particularly leverage an online learning approach to deal with the diversity of the contributing devices, users, and usage scenarios, while offering a resource-efficient solution that minimizes the required user participation.

3.3 Online Personalization

Our *ContextSense* solution includes three phases: the feature selection, the classifier selection followed by the initial training, and the online-learning update algorithm.

3.3.1 Feature Selection

Today's smartphones embed an increasing number of sensors that may serve to contextualize the observations that the crowdsensors gather. Although the list of relevant sensors varies from one smartphone to another, high-end smartphones may provide the following *features*: *Light density*; *Abstract proximity* (the distance from an object to the screen of the device); *Magnetic strength*; *Temperature* of the ambient air; *Pressure*; *Humidity*; *GPS accuracy*; *GSM RSSI* value; *Wi-Fi raw RSSI*; and *Abstract RSSI level* (i.e., the overall signal quality). We specifically focus on eliciting the features that best contribute to classify the observation context with respect to: in-/out-pocket (M_{pocket}), in-/out-door (M_{door}), and under-/upper-ground (M_{ground}).

Following the state-of-the-art analysis and taking into account the sensors embedded in today's smartphones, we find the most relevant feature set for each of our three classifications using a dataset. We leverage the *DATASET 1*, which provides us with the supporting ground truth for the feature selection.

> *DATASET 1* assembles labeled sensor data from a Crosscall Trekker-X3 smartphone, covering all the scenarios to be classified. All the environmental sensors and network modules were active during the data collection. *DATASET 1* contains $20k$ labeled entries contributed by a single user; it includes all the candidate features and covers uniformly the three contexts to be classified. Each instance has three user-encoded labels, which represent the ground truth result for the three classifications.

We assess the significance of each candidate feature using the following scoring metrics [43]: *Information gain* is the expected amount of gained information, *aka*, reduction of entropy. *Gain ratio* is a ratio of the information gain and the attribute's essential information, which reduces the bias towards multi-valued features that occur in information gain. *Gini* is the inequality among values of a frequency distribution. *Chi2 (χ^2)* is the dependency between the feature and the class as mea-

sured by the chi-square statistic. Finally, *ReliefF* is the ability of an attribute to distinguish between classes on similar data instances.

Table 3.1 provides the significance of the features for M_{pocket}, M_{door}, and M_{ground} within *DATASET 1*. As illustrated in the table, not all the features are relevant to each of the three classifications M_{pocket}, M_{door} and M_{ground}.

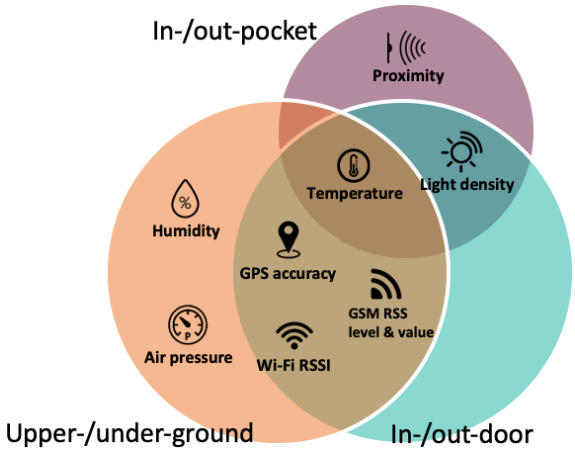


Figure 3.3.1: The features selected for classification

We note that the features that show a higher *information gain* and *gain ratio*, also show higher *Gini* and *ReliefF*. We then select the features that induce both high *information gain* and high *gain ratio*, while we set the required threshold value to 0.1 for both. The metrics lead to the selection depicted in Figure 3.3.1 for each of the three classifiers.

Feature set for M_{pocket}: The top three features for the M_{pocket} classification are proximity, temperature, and light density. Proximity is widely used for in-pocket detection. Light density remains obvious, although less significant, candidate. While temperature is an additional relevant feature, not all devices can provide it.

Feature set for M_{door}: The M_{door} classification uses GPS accuracy, abstract RSSI, GSM RSSI, Wi-Fi RSSI, light density, and temperature. Although GPS is the most important feature to distinguish in- and out-door scenarios, it is also the most power-consuming feature. Rather than totally disabling it, it can be occasionally used depending on the preference of the user. Wireless RSSI shows a lower but still significant contribution to classification because, in an indoor environment, the wireless signal is non-line-of-sight as it reflects on walls and is obstructed by obstacles.

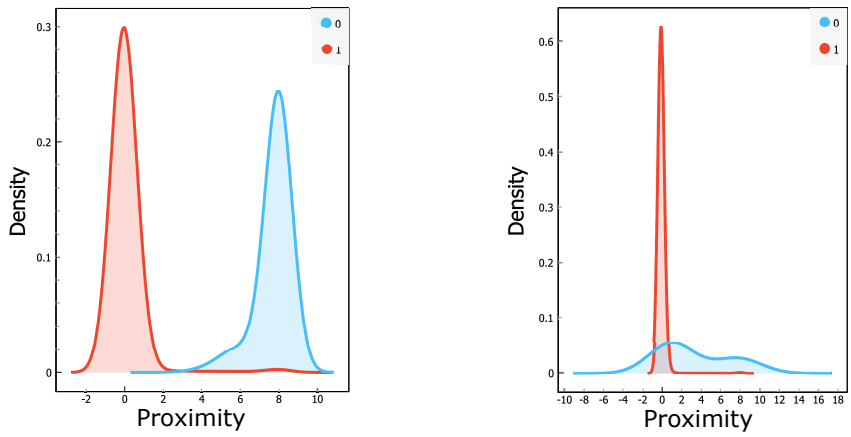
Feature set for M_{ground}: Even though the under-ground scenario is considered a sub-case of the in-door scenario, it requires abstract RSSI, GPS accuracy, temperature, GSM RSSI, pressure, Wi-Fi RSSI, and humidity. Pressure and humidity

Feature/Metric	Info. Gain	Gain Ratio	Gini	χ^2	ReliefF
In-/Out-Pocket classification M_{pocket}					
Light density	0.310	0.155	0.169	3758.434	0.034
Proximity	0.931	0.720	0.478	17776.819	0.329
Magnetic strength	0.024	0.012	0.016	405.081	0.011
Temperature	0.344	0.172	0.213	273.650	0.097
Pressure	0.063	0.032	0.042	298.334	0.036
Humidity	0.096	0.048	0.064	1276.633	0.076
GPS accuracy	0.057	0.042	0.037	165.818	0.001
GSM RSSI value	0.017	0.008	0.011	256.149	0.034
Wi-Fi raw RSSI	0.073	0.069	0.040	682.765	0.030
Abstract RSSI level	0.027	0.017	0.017	382.020	0.058
In-/Out-Door classification M_{door}					
Light density	0.255	0.127	0.157	5041.293	0.050
Proximity	0.098	0.076	0.063	1427.619	0.038
Magnetic strength	0.167	0.084	0.093	2633.329	0.008
Temperature	0.228	0.114	0.125	50.212	0.070
Pressure	0.127	0.064	0.077	341.875	0.127
Humidity	0.045	0.022	0.029	0.343	0.094
GPS accuracy	0.974	0.715	0.482	9085.097	0.996
GSM RSSI value	0.738	0.370	0.384	11211.066	0.157
Wi-Fi raw RSSI	0.320	0.180	0.148	437.694	0.133
Abstract RSSI level	0.794	0.493	0.416	15416.226	0.293
Under-/On-Ground classification M_{ground}					
Light density	0.102	0.050	0.061	1662.044	0.009
Proximity	0.078	0.060	0.051	1599.679	0.007
Magnetic strength	0.017	0.008	0.011	297.901	0.006
Temperature	0.485	0.243	0.255	1905.520	0.255
Pressure	0.376	0.188	0.202	6136.352	0.224
Humidity	0.276	0.138	0.142	2475.257	0.161
GPS accuracy	0.434	0.318	0.222	4182.467	0.528
GSM RSSI value	0.463	0.232	0.262	8031.071	0.143
Wi-Fi raw RSSI	0.181	0.170	0.086	5594.195	0.103
Abstract RSSI level	0.547	0.340	0.296	11586.949	0.250

Table 3.1: The significance of features for the classifications

are decisive as long as the device can provide them. The wireless RSSI contributes because the network connectivity is usually weak in an under-ground compared to an on-ground environment.

3.3.2 Classifier Initialization



(a) 1 device contribution (b) 3 devices contribution

Figure 3.3.2: A feature distribution *wrt* in-pocket(1)/out-pocket(0) classification

We may now train the three classifiers –M_{pocket}, M_{door}, and M_{ground}. Using mixed training sources causes the risk of weakening the ability of classification due to the diversity of the feature values across devices and user behavior. As an illustration, Figure 3.3.2 shows the distribution of the proximity feature in the in/out-pocket scenarios in the case of a single training set (3.3.2a: single user device) and a mixed training set (3.3.2b: three user devices), respectively. We observe that involving more devices and users in the training set creates interference between the distribution of the individual's features and results in blurring the features. The same phenomenon for light density has been observed in [119, 120]. Crucially, the single user-specific model outperforms the model trained on data pooled from several users [199]. Similarly, a machine learning model for each smartphone/sensor brand would provide a better classification performance. Nevertheless, in practice, this is hardly feasible given the diversity of smartphones/sensors as it would require performing training for each smartphone/sensor brand. Thus, we initialize our three classifiers, i.e., machine learning models, with *DATASET 1* that we used for the feature selection.

We need to select the best machine learning algorithm for the M_{pocket}, M_{door}, M_{ground} classifiers using the most significant features associated with each of them. The classifiers must be efficient in terms of both classification accuracy and time/space cost, especially with respect to their local inference/update running on the device. There are various algorithms eligible to address the classification problem [151],

although fewer are updatable. We have specifically compared six candidate updatable algorithms [203]: *Hoeffding Tree* (Very Fast Decision Tree), *IBk* (Instance Based K-nearest neighbors classifier), *KStar* (Instance-based Learner), *LWL* (Locally Weighted Learning), *updatable Naive Bayes* (*NaiveBayes* for short), and *SGD* (Stochastic Gradient Descent).

Using *DATASET 1* again, we have assessed the candidate algorithms for M_{pocket}, M_{door} and M_{ground} according to the following metrics: *Size* is the serialized model size of the initial classifier, which is an important metric due to the (relative) resource constraint of the mobile device and the fact that the size may increase as the model gets updated locally. *CVCA* (10-fold Cross-Validation Classification Accuracy) characterizes the cross-validation split of the data into 10 folds, where the machine learning model is tested by holding out examples from 1 fold at a time. The model is then induced from the other 9 folds, and examples from 1 fold are classified. The procedure is repeated for all 10 folds. The classification accuracy is the proportion of correctly classified examples. *OLR* (Online Learning Runtime) indicates the time taken for updating a machine learning model with a fresh instance (i.e., user feedback). Finally, *IR* (Inference Runtime) indicates the time taken for carrying out an inference using an incoming feature vector.

Table 3.2 compares the candidate algorithms according to the same four metrics for our three classifications: M_{pocket}, M_{door} and M_{ground}. The result for M_{pocket} in Table 3.2 shows that all the classifiers can provide a similar high *CVCA* of about 99%. However, a significant difference appears among the sterilized sizes: *IBk*, *KStar*, and *LWL* are storing training instances inside the model, which makes the size of the classifier proportional to the size of the training dataset. Instead, *Hoeffding Tree*, *NaiveBayes* and *SGD* require a much lower *Size*. *IBk* and *LWL* have an *OLR* greater than $3ms$, while it is less than $1ms$ for the other four algorithms. *IBk*, *KStar* and *LWL* all have much longer *IR* than *Hoeffding Tree*, *NaiveBayes* and *SGD*. A better *CVAC* result is discovered for M_{door}: All the algorithms provide the maximum classification accuracy in cross-validation of 100%. However, although the dataset is unchanged compared to M_{pocket}, the serialized size of *IBk*, *KStar* and *LWL* increases due to the number of selected features. Besides, the *OLRs* do not change significantly and are less than $1ms$ except for *IBk* and *LWL*. *IBk*, *KStar* and *LWL* still have a longer *IR* than the other three algorithms. Result for M_{ground} shows that *LWL* and *NaiveBayes* give lower classification accuracy in cross validation than other algorithms but still over 95%. The high storage cost remains for *IBk*, *KStar* and *LWL* as they require storing historical data. They further cost a much longer *IR* than the other three algorithms. As for *Hoeffding Tree*, *NaiveBayes* and *SGD*, both their *OLR* and *IR* remain below $1ms$, with the negligible exception of the *IR* of *NaiveBayes* at 1.223. Overall, *IBk*, *KStar* and *LWL* show the highest space and time costs, and we discard them.

In particular, all the algorithms provide a high CVCA. Depending on the algorithm, the sterilized size significantly differs because IBk, KStar, and LWL are storing the training instances in the model. Hence, the size of the classifier gets

Metric / Model	Hoeffding Tree	IBk	KStar	LWL	Naive Bayes	SGD
M_{pocket}						
Size (kB)	16	1158	1157	1158	3	5
CVCA (%)	99.1538	99.469	99.350	99.149	98.999	99.180
OLR (ms)	0.020	3.809	0.081	4.344	0.012	0.123
IR (ms)	0.057	10.545	165.844	91.325	1.635	0.018
M_{door}						
Size (kB)	9	1612	1791	1764	4	6
CVCA (%)	100	100	100	100	100	100
OLR (ms)	0.036	5.150	0.062	5.813	0.011	0.172
IR (ms)	0.071	11.823	364.790	109.644	1.610	0.047
M_{ground}						
Size (kB)	13	1763	1763	1763	4	6
CVCA (%)	100	100	100	98.060	97.105	100
OLR (ms)	0.024	4.628	0.062	6.720	0.009	0.111
IR (ms)	0.061	15.160	238.916	128.149	1.223	0.018

Table 3.2: Evaluation of the learning algorithms

proportional to the size of the training dataset. IBk and LWL have a greater OLR compared to others. IBk, KStar, and LWL all have much longer IR than Hoeffding Tree, Naive Bayes, and SGD. The OLR and IR remain low for Hoeffding Tree, Naive Bayes, and SGD. We finally selected the Hoeffding Tree as a training algorithm because it is characterized by the highest accuracy and lowest space/time costs.

3.3.3 Inference and Update

The classifiers for the sensing context need to be personalized. They should be deployed on smartphones to evolve according to the specifics of the device, user, and even scenario contributing to the crowdsensed observations. In particular, online learning for classifiers update copes with the following aspects:

1. *Biases in the feature value across diverse device models:* The values collected for the same feature on different devices can show significant biases due to the diversity of hardware models.

2. *Availability of features depending on the device and user preference:* The feature availability depends on both the device's capability and the configuration set by the owner, who decides which embedded components are switched on/off.

3. *Classification on new scenarios not covered during the initial training:* The classification may not work in, e.g., a new physical region, another season, a different city. The initial classifier may not cover scenarios that will be encountered across time and space.

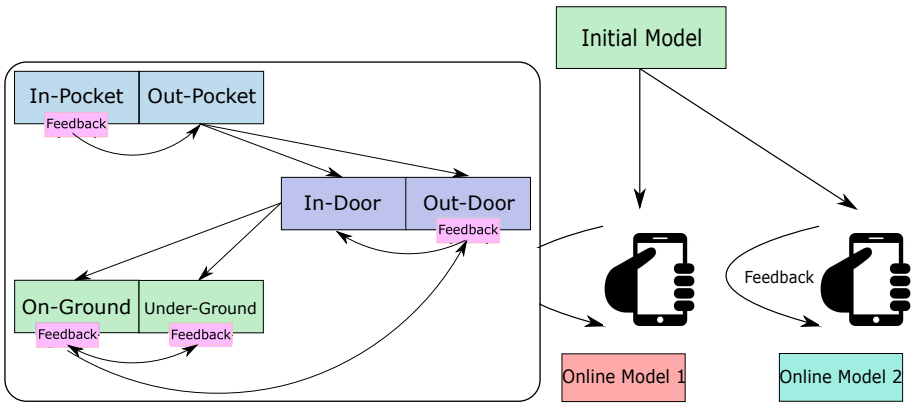


Figure 3.3.3: Online learning for personalization - the initial model is deployed on the participating devices, and becomes a local user-centric model on each device, which is evolving through feedback; the feedback is collected only following some inference results and the inferences are hierarchical

Figure 3.3.3 summarizes the design rationale of our online learning solution embedded in *ContextSense*: the initially trained classifier is deployed on the participating devices at the time of installation of the integrated crowdsensing middleware/application, e.g., Ambiciti [82]. While the inference of the sensing context is running on devices, feedback is requested to users for assessing the correctness of the inference result. The opportunistically collected feedback is then converted to a labeled training instance that updates the current machine learning models. We boost online learning by creating a new instance from the feedback and using it multiple times for the model update.

Algorithm 1 Hierarchical inference and update

Input: A feature vector $\vec{f}$ with a feedback $u \in U$ (options)
Output: The integrated inference result c shown to the user
$u \in \emptyset \subset U$ *possible if the user does not give feedback*

1: $c \leftarrow null$
2: **if** $M_{pocket}(\vec{f}) = inpocket$ **then**
3: $c \leftarrow inpocket$ **and** $U \leftarrow \{outpocket\}$
4: **if** $u = outpocket \in U$ **then**
5: update M_{pocket} using instance $(\vec{f}, u)$
6: **end if**
7: **else if** $M_{door}(\vec{f}) = outdoor$ **then**
8: $c \leftarrow outdoor + outpocket$ **and** $U \leftarrow \{indoor\}$
9: **if** $u = indoor \in U$ **then**
10: update M_{door} using instance $(\vec{f}, u)$
11: **end if**
12: **else if** $M_{ground}(\vec{f}) = underground$ **then**
13: $c \leftarrow underground + indoor + outpocket$ **and** ...
$U \leftarrow \{onground\}$
14: **if** $u = onground \in U$ **then**
15: update M_{ground} using instance $(\vec{f}, u)$
16: **end if**
17: **else**
18: $c \leftarrow onground + indoor + outpocket$ **and** ...
$U \leftarrow \{underground, outdoor\}$
19: **if** $u = underground \in U$ **then**
20: update M_{ground} using instance $(\vec{f}, u)$
21: **else if** $u = outdoor \in U$ **then**
22: update M_{door} using instance $(\vec{f}, u)$
23: **end if**
24: **end if**
25: **return** c **and** run from line 1 for next loop

The feedback requirement should be limited as much as possible to minimize the

burden on users, while still enhancing the accuracy of our three classifiers M_{pocket}, M_{door} and M_{ground} over time. We achieve this by applying a hierarchical inference and by updating the three classifiers conditionally (See Algorithm 1). The hierarchy follows the in-pocket classifier's predominant role over the two others and the in-door classifier over the under-ground one when sensing the physical environment. In more detail, a crowdsensed measurement is relevant for the analysis of most environmental phenomena if it is out-pocket; in-pocket devices have less opportunity to contribute to the crowdsensing, as shown in [185]. The in-door/out-door detection is meaningful only when the device is out-pocket and ready for sensing. Furthermore, the under-ground/upper-ground case is a sub-scenario of the in-door situation. Also, while requesting the user's feedback about a single inference may be acceptable, requesting the feedback about three inferences is too much to ask.

3.4 Prototype Implementation

We have implemented the proposed personalization solution *ContextSense* as an Android application prototype. Our *ContextSense* solution for crowdsensing is also available at GitHub[1]. We outline below the integration of our online learning approach to the contextualization of the contributed observations (see Figure 3.4.1).

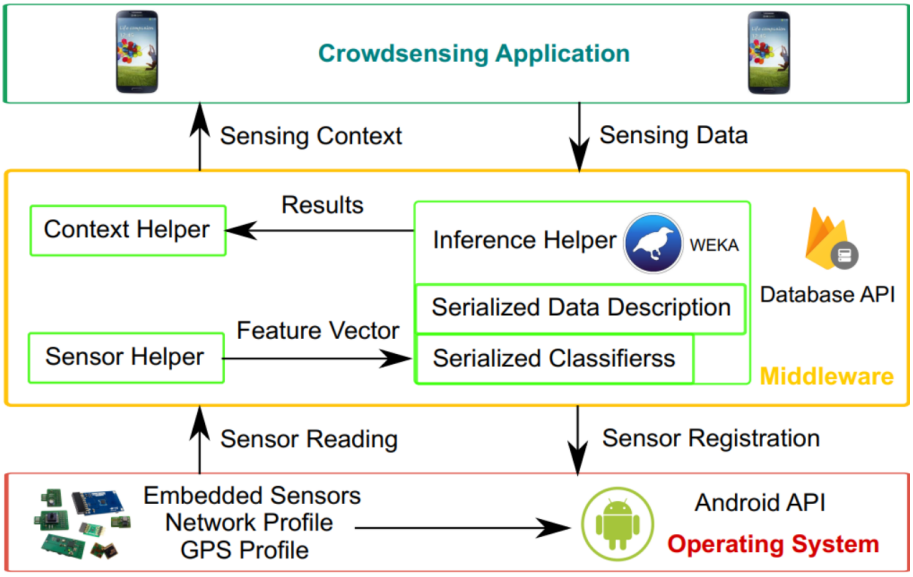


Figure 3.4.1: The *ContextSense* middleware solution

The initial classifiers are trained only once on a computer and are deployed at the time of installation of *ContextSense*. They are serialized and imported into the Android application to be loaded at runtime and updated locally. The application collects features from sensor readings, infers the current sensing context, collects

[1] https://https://github.com/sensetogether/ContextSense

opportunistic feedback, and updates the classifiers accordingly. The application also implements a feature extractor for using the feedback and a user interface for requesting feedback. The *ContextSense* solution relies on the *Hoeffding Tree* and uses only negative feedback hierarchically to keep the amount of feedback to a minimum, thus limits the burden put on the end user. We rely on the WEKA library [203] that implements the *Hoeffding Tree* algorithm.

While the inference of the sensing context is running on devices, *ContextSense* needs to collect the user feedback to assess the correctness of the inference result. The feedback is then converted to a labeled training instance that updates the current machine learning models. Practically, the *opportunistic* feedback from the user is collected using a permanent notification. The notification provides the user with information about the inferred context. Then, the user decides if and when to provide feedback upon incorrect inferences.

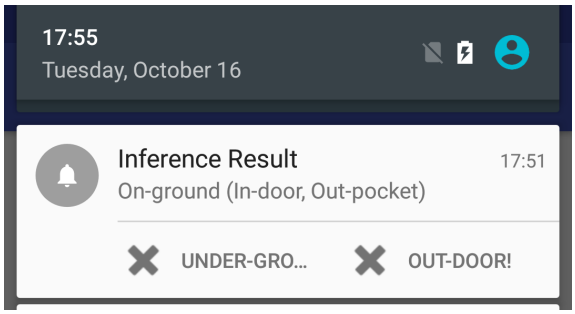


Figure 3.4.2: Example of *ContextSense* notification UI - the notification includes at the top the inferred context and at the bottom some button(s) that can be pressed to provide a (Boolean) feedback

An example of such a notification is illustrated in Figure 3.4.2. The information about the context is brief, and the user can acknowledge any incorrectness with respect to the inferred context. This feedback then serves to improve the performance of the classifier. The hierarchical inference has four genres of results: "in-pocket", "out-door, out-pocket", "under-ground, in-door, out-pocket", "on-ground, in-door, out-pocket". For the first three types of results, the user has only one feedback option to select each time; for the last result, the user has two options to make a selection.

3.5 Performance Evaluation

We evaluate the *ContextSense* solution considering the inference accuracy and number of feedback as metrics. We also compare our solution with the alternative non-hierarchical approach. Finally, we show the resource cost of *ContextSense* for the end device.

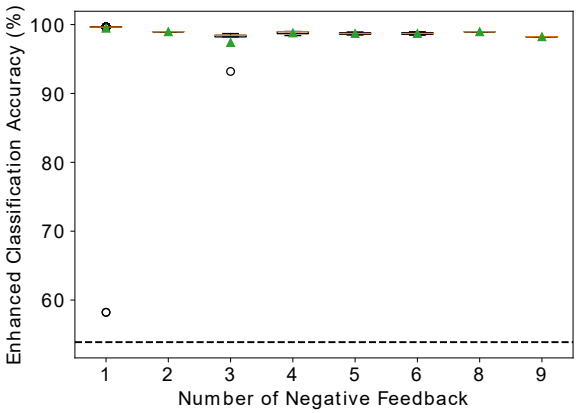
3.5.1 Accuracy and Number of Feedback

Recall that the initial models are trained using *DATASET 1*. We evaluate our personalization approach using a totally new *DATASET 2* as the test dataset.

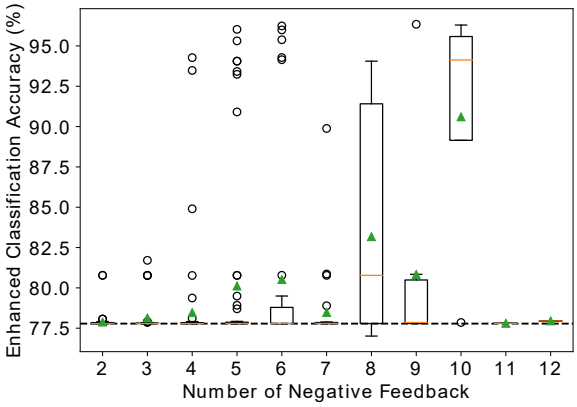
> *DATASET 2* contains $20k$ instances, with each embedding three labels representing the ground truth, and covers all the relevant scenarios (i.e., in/out-pocket, in/out-door and under/upper-ground) uniformly. Different to *DATASET 1*, the environmental sensors, including temperature, humidity, pressure, are not available on the contributing device Xiaomi Redmi Note 4, and the available sensors come from a distinct manufacturer. Besides, the user switches off the Wi-Fi and the GPS modules from time to time. Furthermore, the data gathered for *DATASET 1* and *DATASET 2* correspond to two different physical environments. They were collected in two different city areas and different months.

We assess our hierarchical algorithm's accuracy according to the number of negative user feedback (i.e., when the inference is wrong). We performed 500 runs of the experiment where, at each run, we randomly select 30 entries from *DATASET 2* to perform inference and request for feedback. While the entire *DATASET 2* ground truth serves to assess the accuracy of classifications. Over the 500 runs, at most 9 inferences were wrong for M_{pocket}, 12 for M_{door}, and 13 for M_{ground}, respectively. Also, as depicted in Figure 3.5.1, the classification accuracy gets better 99%, 86% and 71% of the time over the 500 runs for M_{pocket}, M_{door} and M_{ground}, respectively. The enhanced accuracy for M_{pocket} and M_{ground} is high compared to the initial accuracy, and is less significant for M_{door}. While the accuracy for M_{pocket} remains stable, the classification accuracy for M_{door} and M_{ground} slightly varies before reaching 91% and 90%, respectively. In summary, our hierarchical algorithm enhances the classification accuracy most of the time while limiting the amount of feedback (overall best 12) requested to the end user.

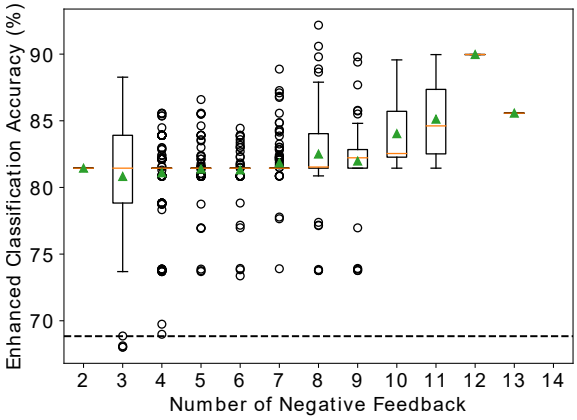
The F_1 score is another measure of test accuracy, considering both the precision and the recall of the test. The F_1 score ranges from 1 (best) to 0 (worst). We assess the F_1 score of the hierarchical algorithm according to the number of negative user feedback occurrences. We also performed 500 experiments where the initial classifiers are trained with *DATASET 1* and are evaluated by simulating negative feedback that leveraging *DATASET 2*. Figure 3.5.2 provides the F_1 score according to the number of (negative and hierarchical) feedback occurrences. Among the 15 occurrences, at most 4 are related to M_{pocket}, 12 to M_{door} and 15 to M_{ground}. The F_1 score gets an enhancement 100%, 90% and 71% of the time for M_{pocket}, M_{door} and M_{ground}, respectively. The enhancement of the F_1 score is the most significant for M_{pocket} and the least significant for M_{door}. Overall, 8 hierarchical feedback occurrences provide a high F_1 score for all three classifiers.



(a) 99% enhancement of M_{pocket}

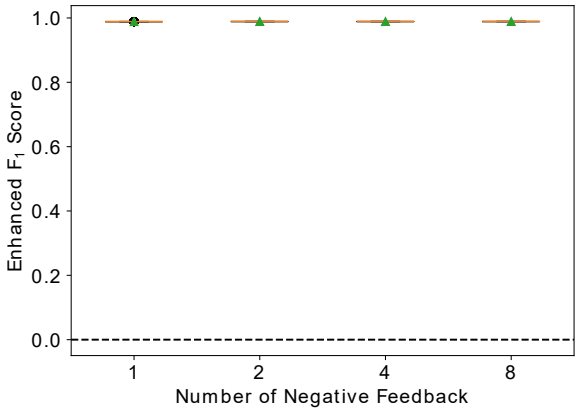


(b) 86% enhancement of M_{door}

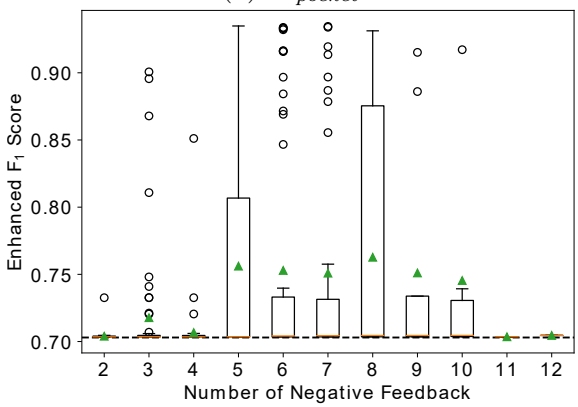


(c) 71% enhancement of M_{ground}

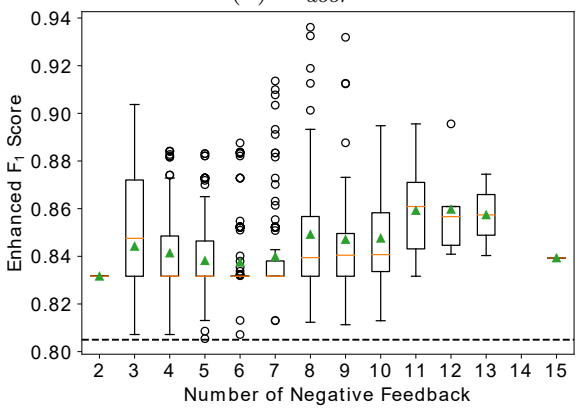
Figure 3.5.1: Enhancement of classification accuracy - box plots showing the improvement of the hierarchical classification accuracy *wrt* the number of feedback compared to the original accuracy obtained without feedback (dashed line)



(a) M_{pocket}



(b) M_{door}



(c) M_{ground}

Figure 3.5.2: Enhancement of classification F_1 score - box plots showing the improvement of the hierarchical classification F_1 score *wrt* the number of feedback compared to the original F_1 score obtained without feedback (dashed line)

3.5.2 Hierarchical vs Multi-class Classifier

An alternative to our approach lies in performing a single and multi-class classification, which distinguishes 8 combinations of in/out-pocket, in/out-door, and under/upper-ground. To compare the multi-class classifier with our hierarchical classifiers, we performed 100 runs of experiments using the same settings of Section 3.5.1.

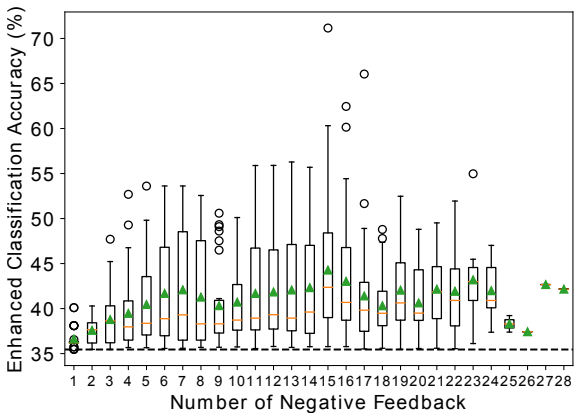


Figure 3.5.3: Multi-class classification accuracy box plot according to the number of feedback

As illustrated in Figure 3.5.3, the mean of enhanced classification accuracy (triangles in the plot) of the multi-class classification always remains below 45%. Also, the initial and enhanced accuracy of a multi-class classifier is 1/2 time lower compared to the hierarchical classifiers (See Figure 3.5.1). Besides, the multi-class classifier requires more feedback compared to the hierarchical classifiers: up to 28 feedback is required. Also, with the multi-class classifier, the user must make a selection among 7 options instead of one or two options.

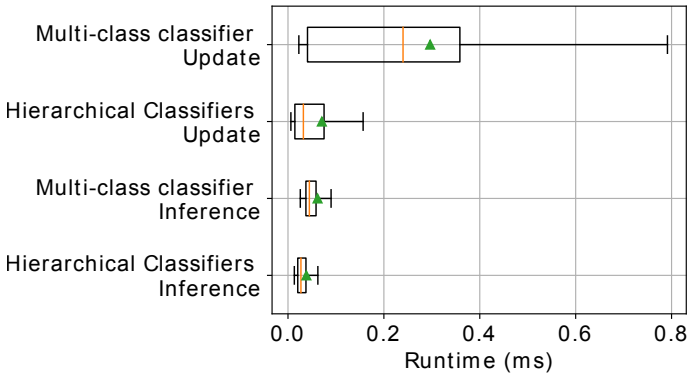


Figure 3.5.4: Comparison of classification execution time - single multi-class Hoeffding Tree *vs* hierarchical binary-class Hoeffding Trees

As illustrated in Figure 3.5.4, our hierarchical classifier involves a much shorter update execution time and a slightly shorter inference execution time compared to

the multi-class classifier, because the hierarchical classifiers do not always perform all the classifications.

Overall, using hierarchical classifiers offers many advantages: (1) using a specific classifier for each context element implies that the classification accuracy remains high for each of them; (2) each classifier only relies on the most relevant features, which reduces the inference and update execution time; (3) a classifier is easily added/removed/replaced when a new context element is handled for the benefit of the crowdsensing application; (4) hierarchical classifiers limit the number of inferences that are triggered; (5) the user feedback required for the personalization of hierarchical classifiers is little and straightforward.

3.5.3 Energy and Resource Efficiency

The components on the smartphone have different power consumption. The power consumed by the GPS module, proximity sensor, and Wi-Fi component is relatively high (e.g., 50 mA, 3 mA, and 2 mA, respectively) compared to the one consumed by the light/magnetic/pressure/humidity sensor, which remains below 1.2 mA.

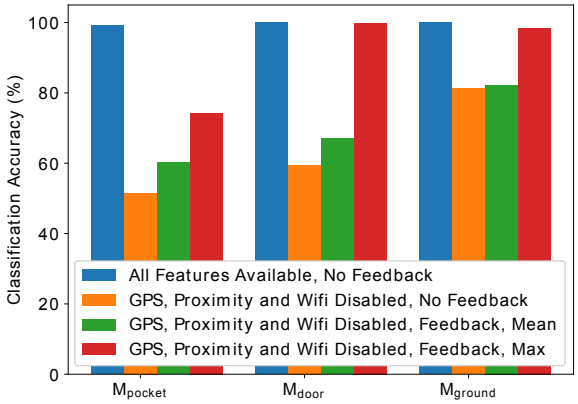


Figure 3.5.5: Classification accuracy under different power modes

> *DATASET 3* includes $20k$ entries obtained when the GPS module, proximity sensor and Wi-Fi components are disabled. The other properties of instances are as same as *DATASET 1*.

In Figure 3.5.5, we evaluate the impact of disabling these three power-consuming components for which we rely on another *DATASET 3*. When the GPS, proximity and Wi-Fi are disabled, the classification accuracy drops down with a decrease of 50%, 40% and 20% for M_{pocket}, M_{door} and M_{ground} respectively. While providing feedback, the mean classification accuracy increases by 10%, 8% and 1% respectively, and the maximum classification accuracy after feedback reaches 74%, 99% and 98% for M_{pocket}, M_{door} and M_{ground} respectively. Indeed, our approach personalizes the classifiers and deals with (power-consuming) features that may be disabled by the user.

Our *ContextSense* prototype requires around 100MB of memory on the smartphone, and it leads to an increase of 5% of the CPU (Qualcomm Snapdragon 636). The inference and update of contexts necessitate around 3MB of memory. The inference execution time is on average $0.2ms$, $0.1ms$ and $0.1ms$ for M_{pocket}, M_{door} and M_{ground} respectively, while the execution time necessary to update the model is $7.3ms$, $7.5ms$ and $10.0ms$ on average, respectively.

Figure 3.5.6: *ContextSense* performance on Android smartphone - red dots represent user feedback actions

Figure 3.5.6 shows the CPU, memory, and energy consumption of running *ContextSense* performing hierarchical inference every second and opportunistic update according to the user feedback. When receiving feedback from the user, the CPU usage slightly increases due to the update of the models. The memory contains the amount consumed by essential Android application components, and the computing does not affect the memory. There is no impact on the network consumption, and the level of energy consumption remains stable and light.

In summary, our *ContextSense* solution allows adapting the tradeoffs between power consumption and accuracy while inducing limited resource consumption on the smartphone.

3.6 Discussion

Although opportunistic crowdsensing is a practical way to achieve ubiquitous sensing for urban environmental monitoring, heterogeneous crowdsensing devices and users introduce inaccurate measurements due to the lack of contextualization. Indeed, context-awareness is essential to the development of mobile crowdsensing applications that can be informed when a suitable sensing context is detected to maximize the effectiveness of the crowdsensing application without requesting the end user to provide contextual details explicitly.

This chapter introduced the *ContextSense* solution to classify the context under which crowdsensors operate and infer whether the smartphone is in-/out-pocket, in-/out-door, and under-/upper-ground. To do so, we leverage online machine learning to contextualize the gathered observation in a resource-efficient way, while account-

ing for the specifics of the crowdsensors that span the device characteristics, the end users' behavior, and the scenarios. We compared six online learning algorithms to select the one achieving the best efficiency, expressed in terms of classification accuracy, execution time, and memory consumption. We further introduced a hierarchical algorithm that requests very few feedback and hence reduces the burden put on the user. Implementation and experiment results show that despite very few feedback, there is a high probability of getting a good/enhanced classification accuracy. Compared to a single multi-class classifier, our hierarchical approach of context inference and update is more efficient in terms of execution time and feedback collection. Our approach is also flexible because the sensing context can be inferred even when some sensor(s) or communication module(s) are unavailable or switched off.

In the next chapter, we leverage the context information of the device, including the inferred physical environment from *ContextSense*, to establish efficient collaborative groups among crowdsensors that are close to each other.

Chapter 4

BeTogether: Let Opportunistic Crowdsensors Collaborate

Contents

4.1 Opportunistic Collaboration

Traditional crowdsensing relies on the cloud/infrastructure server to achieve efficient management. The cloud server may recruit the most useful set of crowdsensors or plan the path of multiple participants to enhance the efficiency of *participatory*

crowdsensing [88, 99], but this results in a limited amount of data and sensing coverage. Alternatively, *opportunistic* crowdsensing creates the potential to provide fine-grained observations that cover urban areas 27/7, provided the engagement of a sufficiently large and diverse crowd [113, 157], but at the cost of high redundancy and resource waste. The cloud server then needs to filter out abnormal contributions and perform post-processing on observations, which leverages the context information to, e.g., eliminate in-pocket measurements and aggregate out-door measurements for an area of interest. The burden to ensure the gathering of relevant data/knowledge is thus shifted from the user to the end device and the server.

For illustration, we refer to the experience with the Ambiciti (formerly SoundCity) solution[1], which comes from a partner start-up company of our research team. Ambiciti features an opportunistic crowdsensing application and cloud-based platform for monitoring the individual and cumulative exposure to environmental pollution, and particularly noise [69]. Opportunistic crowdsensors gather the sound level using the smartphone's microphone along the user's journeys. The development of Ambiciti started in 2014, resulting in the first launch of the application in summer 2015[2]. It has then shown that the assimilation of crowdsensed observations allows generating street-level noise pollution maps that enhance traditional simulated maps [184], provided the calibration of the application [185]. However, the analysis of noise data collected in Paris over the years 2015 - 2017 have highlighted that only less than 10% of the crowdsensed observations contribute to the assimilation of relevant knowledge [82, 115]. Of course, one may consider that 10% of massive data is still a valuable source. Still, low-quality and redundant sensing incurs a significant waste of resources from end devices to cloud server, which is not sustainable. Moreover, in the –not so exceptional– cases where the crowdsensing application attracts a few committed users, then the knowledge from the collected observations is not worth the spending.

An efficient crowdsensing mainly requires the application to not deplete crowdsensors' resources. We thus aim at increasing the data quality while reducing the resource cost on the end device to support opportunistic crowdsensing at scale. Group-based collaboration is a way to achieve efficiency, by sharing the workload among several workers. However, collaboration in opportunistic crowdsensing differs from the one in participatory crowdsensing, and, thus requires a specific solution. Indeed, *participatory* crowdsensing campaigns may request a group of people to sense, exchange information, and share their findings, to improve the quality of the collected data [171, 8, 17, 224, 88]. With *opportunistic* crowdsensing, the collaboration must be opportunistic: the potential groups of crowdsensors that may collaborate must be found/detected in a proactive and autonomous way.

Still, the fact remains that crowdsensors tend to group together naturally. Indeed, crowdsensors are "*social sensors*" that outfit people, as opposed to mere physical equipment that senses the environment and transmits data. When the crowd-

[1] http://ambiciti.io/

[2] https://project.inria.fr/siliconvalley/2015/08/19/soundcity/

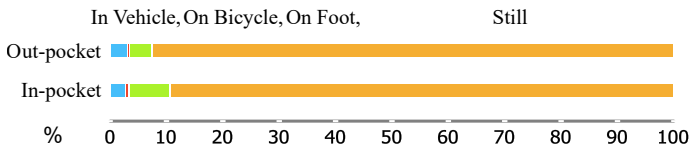


Figure 4.1.1: Distribution of crowdsensor activities - observation on one-year crowdsensing dataset, 92.61% out-pocket and 89.40% in-pocket smartphones perform crowdsensing when people are still

sensing is opportunistic, users follow their daily routine without being directly involved in the sensing task [38]. Thus, at the micro-level, behavioral signatures (i.e., routines) as well as recurrent meeting patterns reflect the underlying relational dynamics of organizations or communities to which the user is affiliated [129]. In particular, a fact that further supports a group-based collaboration strategy for opportunistic crowdsensing, is that people tend to cluster/stay with others at small scale, as part of their social activities [41, 167], following a daily/weekly routines [56, 45]. For example, many employees work together in the same building area; friends and families usually meet and, e.g., sit together on the grass of a park for leisure time. Meanwhile, as illustrated in Figure 4.1.1 and also pointed out in [82], time-related to commuting is low comparing to the time during which people are mostly still.

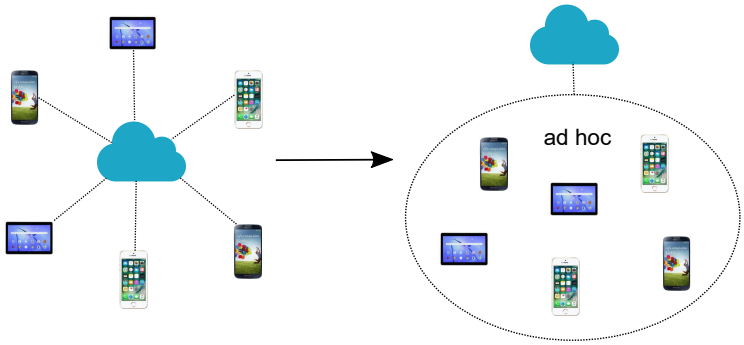


Figure 4.1.2: Collaborative crowdsensing groups

Leveraging the social nature of crowdsensors, we introduce the *BeTogether* middleware that enacts collaborative crowdsensing groups (Figure 4.1.2). The design rationale of *BeTogether* takes advantage of the location- and time-dependence of the crowdsensing: co-located crowdsensors contribute related observations [122], and may thus collaborate to improve the provided measurements and avoid unnecessary duplication of work. More precisely, the crowdsensing tasks that need to be performed on the end devices encompass: environment sensing, location provisioning, data processing, and uploading to the cloud. Then, while the replication of physical sensing within a group potentially allows the gathering of more accurate knowledge, any of the other tasks only needs to be performed by the most cost-effective group member(s).

Nevertheless, a primary challenge for the *BeTogether* solution is to cope with the dynamics and heterogeneity of the crowdsensors [82]: the user activity, the resources available on the device, and the position of the sensors (smartphones) are all criteria that characterize the crowdsensor's contribution to the upper layer application. The combination of these criteria defines the *context* of the grouping, which enriches the sensing context defined in the previous Chapter 3.

BeTogether thus creates context-aware collaborative groups of opportunistic crowdsensors in an *ad hoc* way, which follows three principles: the members within any collaborative group (i) stay together for a long enough time to prevent constant changes and unnecessary grouping reconfiguration; (ii) operate within the same physical environment (e.g., in-door *vs.* out-door) and hence sense related physical phenomena; and (iii) perform the same activity so that they behave alike –e.g., it is preferable that all the crowdsensors that collaborate either move together or stay still. Then, upon creating a group, *BeTogether* distributes the crowdsensing tasks to the adequate group members according to the crowdsensor's context, e.g., a smartphone located in the pocket cannot adequately sense the surrounding sound level. Overall, the *BeTogether* solution promotes opportunistic crowdsensing that imposes a minimal burden on end-users, while increasing the accuracy and resource efficiency of the crowdsensing system.

After positioning the *BeTogether* solution with respect to the related work in the next section, this chapter details the following contributions to the field of opportunistic crowdsensing:

- Introducing opportunistic, context-aware crowdsensing groups, so that any group delivers more accurate knowledge and enhances –both local and global– resource-efficiency compared to the gathering of the individual contributions of its member on the cloud (Section 4.3). Such a grouping mechanism relies on:
  1. User-specific context inference including the solution in Chapter 3 that accounts for the specifics of end devices and user behaviors (Section 4.3.1);
  2. A set of utility functions that drive the distribution of the crowdsensing tasks among the group members so that any crowdsensing group does meet the objectives of improved data quality and resource-efficiency (Section 4.3.2);
  3. An algorithm for the context-aware discovery of co-located crowdsensors, grouping configuration, and further task assignment (Section 4.3.3).
- A prototype implementation of the *BeTogether* solution for the Android platform, which paves the way for experimental evaluation but also the implementation of crowdsensing applications by third parties (Section 4.4).
- Evaluation of the *BeTogether* solution. We leverage a one-year dataset of nearly one million entries, which comes from the Ambiciti crowdsensing application. Results show that the collaborative crowdsensing group improves data

quality, while its behavior adapts according to the context of the crowdsensors for the sake of resource-efficiency (Section 4.5).

4.2 Related Work

In Chapter 2, we reviewed and positioned our contributions with respect to collaborative crowdsensing strategies that aim at guaranteeing the quality of observations at low sensing cost. In this chapter, we focus specifically on enacting groups of co-located crowdsensors in an ad hoc way, that is, without relying on any infrastructure. While not being exhaustive, we review below the most representative related work.

4.2.1 Group-based Collaboration

So far, most collaboration strategies among co-located crowdsensors applies centralized management: data flows directly from each crowdsensor to a cloud/edge server that optimizes the collection and analyzes the data [70, 110, 83]. The drawback of these solutions is that they require every crowdsensor to be connected to the same access point/server in advance, which limits the applicability of the mobile crowdsensing. Instead, a fully distributed collaboration that promotes in-network management without the involvement of an edge/cloud server for controlling and planning the task allocation would be more scalable and less failure-prone by design.

There exist few collaborative crowdsensing solutions that neither rely on an edge server nor require end devices to be connected to the same WLAN. We refer to this kind of strategy as "*ad hoc* collaboration". In practice, nearby crowdsensors may elect a single node that acts as proxy, gathers the local observations provided by other crowdsensors and forwards them to the cloud [78]. This election is repeated to deal with the *ad hoc* nature of the network. Several criteria have been used to determine which crowdsensor should operate as proxy, among which the physical proximity, the remaining energy level [50], the number of neighbors, the connection quality, the stability of the connection [146, 147].

The work mentioned above mostly concentrates on selecting a network proxy providing an access to the cloud, using various selection criteria. Very limited work promotes an opportunistic and *ad hoc* collaboration among crowdsensors, beyond the data forwarding/uploading. A fully distributed collaboration is proposed in [166] to support an opportunistic macro calibration, which operates in the background and in an automated manner. Precisely, crowdsensors that are within the relevant sensing and D2D communication range coordinate so that the observations of the previously calibrated crowdsensors are used to calibrate the other participants. As a result, such collaboration is particularly well suited to encounters between people as, for example, in public spaces.

Through this specific work on distributed calibration, we have noticed that distributed collaboration goes beyond the simple relaying of sensing data [195, 187, 181].

There is a need for leveraging the increasing functionalities of smart devices to support additional functionalities (e.g., selective sensing, location sharing, and data processing). Furthermore, additional context factors (e.g., human mobility, device heterogeneity, and people relationship) should be considered to support effective collaboration among opportunistic crowdsensors.

In our work, we consider collaboration with respect to the various crowdsensing tasks/roles, which are not restricted to the proxy role. We then deal with the distribution of these tasks according to the context of the co-located nodes so as to enhance both resource utilization and data quality, locally and globally. We further group co-located crowdsensors that behave alike and observe compatible physical phenomena because they are likely to collaborate for a long time, which increases the group stability.

4.2.2 Collaboration via D2D Networking

The opportunistic networking of crowdsensors within a group requires special care. In particular, the opportunistic grouping of nodes has been extensively studied in the early 2000s as part of the emergence of MANET (e.g., see [125]). Nowadays, D2D communication, which is growing in popularity, offers a convenient base ground to opportunistically form a transient network, discover the nearby devices along with the services they can offer, and exchange data through the established D2D connections [214]. More recently, the variety of low-level D2D technologies (including Bluetooth Low Power, Wi-Fi Direct, and the very recent Wi-Fi Aware) have been integrated and used jointly to support multi-technology networking [93, 92]. Building on this trend, we leverage innovation in D2D networking for the creation of the opportunistic crowdsensing groups. In particular, the *BeTogether* design builds explicitly upon Wi-Fi Direct [7, 127], although alternative D2D communication technologies could be considered. Then, we customize the creation of *ad hoc* groups/networks according to the specifics of opportunistic crowdsensing; that is, the groups are created according to the context, spanning the user behavior, physical environment, device attributes, and the expected lifespan of the given context.

4.3 Context-aware Collaborative Groups

Our strategy to collaborative crowdsensing revolves around the opportunistic creation of groups of co-located crowdsensors, such that any given group fills the mandatory crowdsensing tasks: *Location tracking*, *Internet connection*, *Sensors measuring* for the target observations, and *Data processing* (over observations). The objective is further to ensure that the collaboration results in reducing resource consumption due to crowdsensing, both locally and globally, compared to a cloud-centric solution. In particular, we leverage the relatively cheap cost – in terms of energy consumption – of D2D communication compared to the costs associated with location management and cellular connection to the cloud.

4.3.1 Context Awareness

The *context* of a crowdsensor serves to assess the relevance of its peers/neighbors that are within D2D communication range for the sake of enhanced efficiency, that is, the ability to improve the gathered knowledge before transferring it to the cloud in a way that reduces resource-consumption.

Context Metadata

The contexts characterizing the crowdsensor's collaboration profile are subdivided into:

- **UA** (*User Activity*) refers to the mobility (or non-mobility) of the end user, which comes from the activity recognition module of the operating system. The module leverages machine learning [53] and relies on the sensors embedded in the smart device to determine if the user is, e.g., still, walking, cycling, or in a vehicle.
- **PE** (*Physical Environment*) reflects the position/placement of the sensing device, which influences the given observations of physical phenomena. It detects whether the crowdsensor is: in-/out-pocket, in-/out-door, and under-/upper-ground. We leverage *ContextSense* in Chapter 3 that applies online learning for such a user-centric inference, which learns the user behavior incrementally on the device [47, 48].
- **DA** (*Device Attribute*) characterizes the ability of the device to contribute to the various crowdsensing tasks. The attributes include the available networking (e.g., type of Internet access, uploading bandwidth) and computing capabilities (e.g., remaining battery, CPU frequency, memory size), the type of embedded sensors (e.g., {"Temperature", "Light", "Pressure", "Humidity", "Sound level"}) together with the related power consumption and accuracy.

In Table 4.1, we illustrate explicitly the context list of *UA*, *PE* and *DA* that we consider in the solution.

Context Duration

The effectiveness of the collaboration strategy depends not solely on the ability of grouping together crowdsensors that have compatible context [3, 4], but also on creating groups that last long enough. Indeed, the latter is critical to ensure that the benefit of the collaborative crowdsensing at the edge outperforms the overhead due to the group management/configuration.

Building upon the fact that human activity is predictable and follows a periodic behavior [55, 56, 45], we leverage online machine learning in a way similar to the inference of *PE* [47, 48], to predict how long the current *UA* and *PE* are going to last. That is, the predictor is continuously updated on the end device so that it

Key	Value (Type, value, metric)
"User Activity"	String, {"Vehicle", "Bicycle", "Foot", "Still", "Unknown"}
"User Activity Duration"	Float, value in minutes
"In-Pocket"	Boolean, {True, False, Null}
"In-Door"	Boolean, {True, False, Null}
"Under-Ground"	Boolean, {True, False, Null}
"<Physical Environment> Duration"	Float, value in minutes
"Location Type"	String, {"GPS", "Network", Null}
"Location Accuracy"	Float, value in m unit
"Location Power"	Float, value in mA unit
"Internet Connection"	String, {"Wi-Fi", "Cellular", Null}
"Internet Up-link Bandwidth"	Float, value in Kbps
"Internet Power"	Float, value in mA unit
"Remaining Battery"	Float, value in mAh unit
"CPU Frequency"	Float, value in MHz unit
"Memory Size"	Float, value in MB unit
"<Sensor> Accuracy"	Float %, set by calibration
"<Sensor> Power"	Float, value in mA unit

Table 4.1: Context attributes for collaboration corresponds to $< Key, Value >$ pairs

keeps evolving according to the specifics of the user. Each time the user changes UA, the following new data instance is provided as input to the machine learning model:

$$\{t_{prev\_UA},\ UA_{prev},\ t_{cur\_UA} - t_{prev\_UA}\}$$

where: t_{prev_UA} is the starting time of the previous UA, UA_{prev} is the previous value for UA, and t_{cur_UA} is the time at which the current UA started. The current time is defined as a tuple of *{day of week, hour of day, minute of hour}* because people tend to have repeated behavior at the week level, and the minute unit is fine-grained enough. The same applies to the update of PE duration, with PE substituting to UA in the above data instance definition.

Thanks to the online learning algorithm, we predict the duration of the current UA (respectively PE) according to the current time and UA (respectively PE). The prediction of the UA/PE duration is a regression problem, rather than a classification problem. Many online learning algorithms may address the classification problem, but very few deals with numerical regression. We have investigated three eligible algorithms: IBk (Instance Based k-nearest neighbor algorithm), KStar (an instance-based learner), and LWL (Locally Weighted Learning) [203]. Ultimately, we selected *LWL* as the online learning algorithm because it provides the lowest RMSE (Root Mean Square Error) and execution latency.

Neighbors with Matching Context

Contexts	Matching Rules (i, j)
PE (except pocket)	**Consistent** for crowdsensors i and j
PE duration	**Greater than** D_{min} for both i and j
UA	**Consistent** for crowdsensors i and j
UA duration	**Greater than** D_{min} for both i and j
$RSSI$ value	**Greater than** $RSSI_{min}$ for both i and j
Bearing (of moving)	**Consistent** for crowdsensors i and j

Table 4.2: Neighbor matching rules

We set several constraints to find the peers that may be collaborative neighbors. Table 4.2 lists the resulting rules that determine the grouping of crowdsensors where the minimum duration is set according to the specific application. PE and UA are the two primary contexts that need to be matched. In practice, the RSSI is used to estimate the distance between two crowdsensors (i.e., transmitter and receiver) [20, 122] and determine whether the two crowdsensors are close enough, i.e., $RSSI > RSSI_{min}$, with respect to the problem at hand [77]. We also consider the RSSI-based distance for assessing the relevance of grouping crowdsensors, which we enrich with the closeness of their respective context that should last for long enough (as predicted using machine learning).

Given the crowdsensors set in the D2D communication range of smartphone i, we define *neighbors* N_i of i as the nearby crowdsensors that match all the rules. The grouping and the assignment of crowdsensing tasks to a set of neighbors then depend on their utilities to carry out the tasks.

4.3.2 Assessing the Crowdsensor Utilities

We introduce a set of utility functions to evaluate to which extent each crowdsensor is eligible to carry out some crowdsensing tasks. We rely on a classical squashing function to normalize the values used in the computation and render comparable the utilities. The squashing function f corresponds to a shifted and scaled logistic function (see Figure 4.3.1):

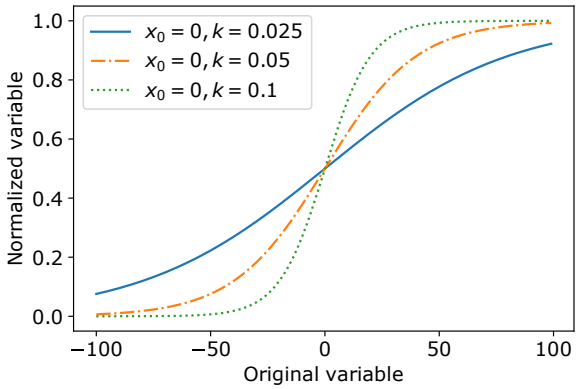


Figure 4.3.1: Squashing function for normalization

$$f(x, k, x_0) = \frac{1}{1 + e^{k(x_0 - x)}}$$

where: the range of f lays into $(0, 1)$, k is the logistic growth rate or steepness of the curve, and x_0 is the x-value of the midpoint of sigmoid, while x is the variable to be normalized. The values of k and x_0 are set according to the domain of the specific x. In the following, we denote k_f and x_f the values of k and x_0 for a given function f, while we define the actual values used in our prototype for the various parameters in Section 4.4.

The following tasks must be implemented within any collaborative group: *Coordinator* (implementing the D2D network access point for the group and assigning the crowdsensing tasks to the connected group members, i.e., *neighbors*); *Location provider* (supplying geographical coordinates); *Internet proxy* (providing Internet access and thus transmitting data to the cloud); *Data aggregator* (analyzing together the collected data locally before sending to the cloud to, e.g., calibrate the sensors [166], aggregate the data [184]); and, of course, regular *Sensors*.

Coordinator

The selection of the coordinator is based on the following criteria:

- *The number of neighbors* has a significant impact on the overall performance because increasing the number of collaborators enables analyzing more data locally and reducing the transmission of data to the cloud. In contrast, there is no benefit in creating groups with too few crowdsensors for which the minimum value δ depends on the application. Given a crowdsensor i and the set N_i of its neighbors, we define:

$$\Delta_i = f(\max\{0, |N_i| - \delta\}, k_\Delta, x_\Delta)$$

- *The occurrences of collaboration* between a crowdsensor i and its neighbors is another important parameter:

$$h_i = f(\frac{\sum_{j \in N_i} t_{collab}(i,j)}{|N_i|}, k_h, x_h)$$

  where: $t_{collab}(i,j)$ is a numeric parameter indicating the number of times i and j have collaborated before, as recorded in the history cache of i.

- *The UA and PE (except in-/out-pocket)* of the crowdsensor should last sufficiently long. Given the predicted $duration(UA)$ (respectively $duration(PE)$) of the current UA (respectively PE) for i, we define:

$$d_i = f(\min\{duration(UA), duration(PE)\}, k_d, x_d)$$

- The *remaining battery capacity* of the crowdsensor should be taken into account. Its normalized value from its original value bat_i is denoted as:

$$b_i = f(bat_i, k_b, x_b)$$

Next, provided the weights w_Δ, w_h, w_d, w_b (all $\in [0,1]$) set for the above criteria, we define the utility $u_c(i) \in (0, w_\Delta + w_h + w_d + w_b)$ of the crowdsensor i associated with acting as a coordinator, as the weighted sum of the above functions:

$$u_c(i) = w_\Delta.\Delta_i + w_h.h_i + w_d.d_i + w_b.b_i$$

Location Provider

While the GPS location brings higher accuracy out-door, it also comes with higher energy consumption and latency than network-based positioning. This leads us to introduce the following normalized value l_i to the support of the positioning by i:

$$l_i = \begin{cases} f(acc_l, k_l, x_l) - f(pow_{gps}, k_{lp}, x_{lp}) & , \text{ GPS} \\ f(acc_l, k_l, x_l) - f(pow_{net}, k_{lp}, x_{lp}) & , \text{ Network} \\ -\infty & , \text{ Null} \end{cases}$$

where: we assume that any crowdsensor i advertises a single location service among *(GPS, Network, Null)*, acc_l is the accuracy of the advertised location service, and pow_{gps} (respectively pow_{net}) is the power consumed by the GPS (respectively network) location service.

The utility function for crowdsensor i of being a location provider is thus defined as $u_l(i) \in (-\infty, 1+w_d+w_b)$, which accounts for the location service source, location accuracy and remaining battery capacity:

$$u_l(i) = l_i + w_d.d_i + w_b.b_i$$

Internet Proxy

The Internet proxy transmits the (aggregated) data provided by the group of crowdsensors to the cloud server. This service may be provided using either long-range cellular or short-range Wi-Fi transmission, while we assume that a node supporting both networks will offer the Wi-Fi-based transmission by default. Then we define:

$$n_i = \begin{cases} f(bw_{up}, k_n, x_n) - f(pow_{wifi}, k_{np}, x_{np}) & , \text{ Wi-Fi} \\ f(bw_{up}, k_n, x_n) - f(pow_{cell}, k_{np}, x_{np}) & , \text{ Cellular} \\ -\infty & , \text{ Null} \end{cases}$$

where: bw_{up} is the upstream bandwidth for the Internet interface; pow_{wifi} (respectively pow_{cell}) is the power consumption of Wi-Fi (respectively cellular Internet) transmission.

The utility function for crowdsensor i acting as Internet proxy is then defined as $u_p(i) \in (-\infty, 1 + w_d + w_b)$, and it accounts for the Internet connection interface, up-link network bandwidth, and remaining battery capacity:

$$u_p(i) = n_i + w_d.d_i + w_b.b_i$$

Data Aggregator

The data aggregator takes in charge the analysis of the sensing data collected locally. While lightweight data processing can be performed by the coordinator or by the proxy, complex data analysis may be delegated to a dedicated device with

the necessary processing capabilities (spanning available memory, CPU frequency, and remaining battery capacity). This results in the following definition for the supporting utility function $u_a(i) \in (-w_c, w_c + w_r + w_d + w_b)$:

$$u_a(i) = w_c.[f(cpu_i, k_c, x_c) - f(pow_c, k_{cp}, x_{cp})] + w_r.f(mem_i, k_r, x_r) + w_d.d_i + w_b.b_i$$

where: cpu_i is the CPU frequency, pow_c is the power consumption of the CPU, mem_i is the available memory, and coefficients w_c, w_r are the weights set for the metrics.

Sensors

The utility of a crowdsensor to carry out the sensing task depends on the accuracy of the contributed observations and their power consumption. In particular, a crowdsensor within a pocket/bag is ignored, which we filter out using the duration of the crowdsensor being out of the pocket (as defined by the PE_{out} value of PE):

$$d'_i = f(duration(PE_{out}), k_d, x_d)$$

which leads to the following utility function, $u_s(i) \in (-1, 1 + w_d)$:

$$u_s(i) = d'_i.[f(acc_i^s, k_a, x_a) - f(pow_i^s, k_{sp}, x_{sp}) + w_d.d_i]$$

where: acc_i^s (respectively pow_i^s) is the accuracy (respectively power consumption) of the sensor of type s on crowdsensor i.

Group Utility

Given the above set of utility functions, the utility of a group with coordinator i is then defined as:

$$U_G(i) = u_c(i) + u_l(l) + u_p(p) + u_a(a) + \sum_{s \in S} \sum_{n \in NS_i} u_s(n)$$

where:

- $l \in N_i$ is the node with the highest utility to act as the location provider among i's neighbors.
- $p \in N_i$ is the node with the highest utility to act as the Internet proxy among i's neighbors.
- $a \in N_i$ is the node with the highest utility to act as the data aggregator among i's neighbors.
- S is the set of sensor types requested by the upper layer crowdsensing application and the set $n \in NS_i$ that defines the nodes with the highest utility to act as sensors s within N_i.

Overall, the coordinator tries to find the best set of crowdsensor neighbors to allocate the tasks to maximize the utility function. Note that there might be neighbors in IDLE state, which only stay in the group but do nothing. During the selection, the following constraints are met:

- Each crowdsensor should be in a group; otherwise, it works alone for all tasks.
- There is no better coordinator than the one elected inside each collaborative group.
- Each collaborative group should have at least one role for each task.

Note that global optimal is not possible because of the lack of global view. What the group achieves is the best effort from the local scope view.

4.3.3 Grouping Algorithm

Provided the utility functions, the grouping algorithm leverages the communication and discovery protocols implemented at the link layer by the state of the art D2D communication technologies. Without loss of generality, our grouping algorithm builds upon the Wi-Fi Direct technology [7]. In short, Wi-Fi Direct establishes a P2P opportunistic network through the discovery of nearby nodes followed by the election of the node acting as the network's access point according to the criteria provided by the upper layer [102, 103] (i.e., in our case, the coordinator utility value provided by the *BeTogether*). The election of such an access point for Wi-Fi Direct is equivalent to a cluster-head election problem [21].

Then, the grouping algorithm runs on every crowdsensor i, either on-demand or periodically, according to the network's dynamics. The algorithm decomposes into the following steps with the duration of each step being set by the upper layer application (see Algorithms 2 and 3 for detail):

1. Advertise presence with context metadata (UA_i, PE_i, DA_i) and their respective duration, so as to discover nearby peers and know their contexts. Create a list of neighbors that matches the context rules.
2. Compute and advertise the utilities including $u_c(i)$, $u_l(i)$, $u_p(i)$, $u_a(i)$, $u_{s_1}(i)$, ..., $u_{s_{|S|}}(i)$, simultaneously receiving the utilities metadata from neighbors following the advertisement of nearby peers.
3. Self-elect as coordinator if $\forall j \in N_i : u_c(i) \geq u_c(j)$ and $\forall k \in N_i : (u_c(i) = u_c(k)) \wedge (ID(i) > ID(k))$ with the crowdsensor's ID being unique and assigned by the middleware to break the tie.
4. Create the Wi-Fi Direct network if self-elected as coordinator and advertise the network to the neighbors that will ultimately join if they have not joined another Wi-Fi Direct network since the beginning of the period.

Algorithm 2 Group Configuration

Require: i: crowdsensor
Input: T: set of tasks that should be allocated
Input: c_i: context of crowdsensors i
Input: $u_t(i)$: utility that crowdsensor i implements task t
Local variable: N_i: neighbors of i sharing the same context
Local variable: C_i: set of contexts provided by neighbors
Local variable: U_i: set of utilities provided by neighbors

1: $N_i \leftarrow \varnothing$, $C_i \leftarrow \varnothing$, $U_i \leftarrow \varnothing$
2: Register service (*BeTogether*, c_i)
3: *Nearby Services Discovery* to get C_i and N_i
4: **for** each task $t \in T$ **do**
5: Compute $u_t(i)$
6: Register service (t, $u_t(i)$)
7: **end for**
8: *Nearby Services Discovery* to get U_i
9: **if** $u_c(i)$ is the highest value in U_i **then**
10: Create a Wi-Fi Direct group as group owner
11: **for** each task $t \in T$ **do**
12: Find crowdsensor k with highest utility $u_t(k) \in U_i$
13: Invite crowdsensor k to implement t
14: **end for**
15: **else**
16: Find crowdsensor k with highest utility $u_c(k) \in U_i$
17: Accept invitation and join group of k
18: **end if**

Algorithm 3 Nearby Service Discovery

Require: i: crowdsensor
Input: T: set of tasks that should be implemented
Output: N_i: neighbors of i sharing same context
Output: C_i: set of contexts provided by the neighbors
Output: U_i: set of contexts provided by the neighbors
1: Discover Wi-Fi Direct Service

2: **Upon** reception of advertisement (*BeTogether*, c_j) from j
3: **if** Rules matched (c_j, c_i) **then**
4: $N_i \leftarrow N_i \bigcup \{j\}$, $C_i \leftarrow C_i \bigcup \{c_j\}$
5: **end if**
6: **return** C_i and N_i

7: **Upon** reception of advertisement (t, $u_t(j)$) from j
8: **if** $j \in N_i$ **then**
9: $U_i \leftarrow U_i \bigcup \{u_t(j)\}, \forall t \in T$
10: **end if**
11: **return** U_i

5. Assign the crowdsensing tasks to the best suited group members, as defined by the highest respective utilities.

Once a group is created, the arrival/departure of members is detected at the link layer by the underlying Wi-Fi Direct protocol. The dynamic enables two approaches for the reconfiguration of the group: (i) periodically, or (ii) on-demand upon the detection of the departure/arrival of a node. The latter performs efficiently and induces almost no cost when very few topology changes occur and when the context evolves slowly. On the other hand, it may lead to unnecessarily high control traffic and constant re-assignment in a highly dynamic environment. Keeping in mind that, in practice, users are most of the time still (and hence evolving in the same context), we adopt an on-demand approach (based on the context change) by default. Nevertheless, the middleware switches to periodic re-assignment when the context gets highly dynamic.

4.4 Prototype Implementation

We implement the *BeTogether* prototype on the end device. Figure 4.4.1 depicts the architecture of the *BeTogether* prototype: the (mobile) crowdsensor components run on Android 6.0+ smartphones or tablets; the server part on the cloud archives the data that is collected using a no-SQL database provided by Cloud Firestore[3].

[3] https://firebase.google.com

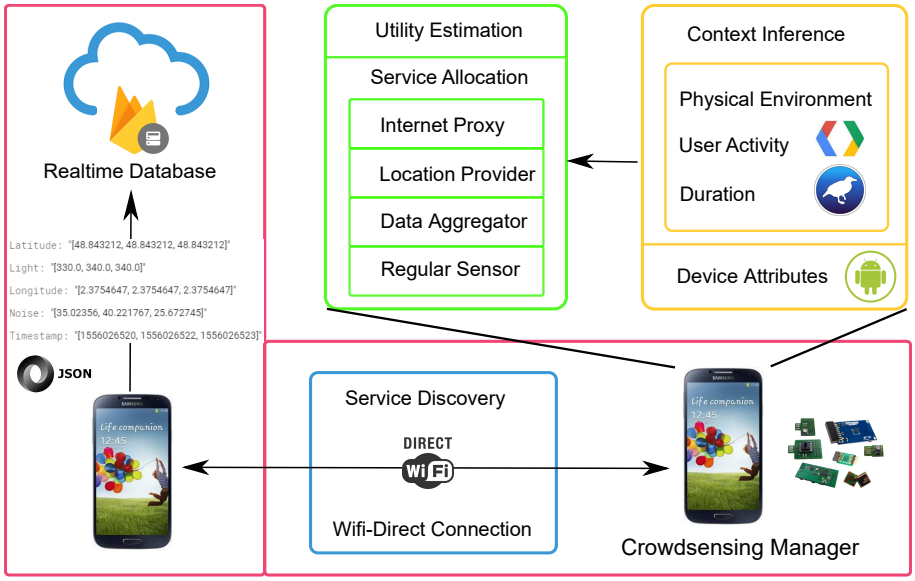


Figure 4.4.1: The *BeTogether* prototype architecture

The *BeTogether* prototype implementation is available at GitHub[4]. In short, the application running on the crowdsensor features the following components:

- *Crowdsensing manager:* It controls the workflow of the collaborative crowdsensing according to Algorithm 2, which includes the orchestration of the service discovery, context inference, computation of utility functions, communication management, and the assignment of tasks. When a task(s) is assigned to a device, the manager further instantiates the corresponding functionality/ and starts a separate thread to execute the related task.

- *Context inference module:* It extracts the crowdsensor context (*UA*, *PE* and *DA*). It infers the *UA* value using the Google Activity Recognition API[5]. It also infers the *PE* values using our user-centric inference module available at GitHub[6]. The inference is performed using Hoeffding Tree, which provides incremental decision tree induction and is included in the WEKA library[7]. It further predicts the time during which the *PE* and *UA* will remain unchanged, using the LWL algorithm also supported by the WEKA library.

- *Utility estimator and task assignment module:* The inferred/predicted contexts are further used to estimate/compute the utilities that assess to which extent the crowdsensor should carry out among the parallel tasks, i.e., coordinator, location provider, Internet proxy, data aggregator and environmental sensors.

[4] https://github.com/sensetogether/BeTogether
[5] https://developers.google.com/location-context/activity-recognition
[6] https://github.com/sensetogether/ContextSense
[7] https://www.cs.waikato.ac.nz/ml/weka

- *Service discovery:* It leverages *Wi-Fi Direct* [7], which provides both D2D/P2P communication and service discovery at the data-link layer. Note that Wi-Fi Direct supports several service discovery protocols, such as Bonjour or UPnP, while our prototype uses the default Bonjour. A Wi-Fi Direct client plays the (logical) role of AP (Access Point) while other Wi-Fi Direct clients join the network that is governed by the AP. Such a network is equivalent to a classical Wi-Fi network.

Normalization Parameters	**Values**
k_Δ, x_Δ, k_h, x_h, k_d, x_d	1, 1, 1, 1, 0.1, $10mins$
k_l, x_l, k_{lp}, x_{lp}	0.1, $10m$, 0.1, $30mA$
k_n, x_n, k_{np}, x_{np}	10^{-5}, $10^5 kbps$, 0.01, $100mA$
k_b, x_b, k_{cp}, x_{cp}	10^{-3}, $10^3 mAh$, 0.01, $100mA$
k_c, x_c, k_r, x_r	10^{-3}, $10^3 MHz$, 10^{-3}, $10^3 MB$
k_a, x_a, k_{sp}, x_{sp}	0.1, 10%, 1, $0.1mA$
Collaboration Parameters	**Values**
w_Δ, w_h, w_d, w_b, w_c, w_r	1, 1, 1, 1, 1, 1

Table 4.3: Parameters configuration and scenario variables

The *BeTogether* prototype implementation and test-bed experiment detailed next, use the parameters defined in Table 4.3. These parameters are set to smooth the normalization curve (Figure 4.3.1) considering nowadays' smartphone hardware as a baseline. As for the weights w_x of the utilities, we consider equality with no preference. The key parameters δ and D_{min} that determine the willingness to collaborate are application-dependent: (i) the minimum size of the group, as defined by δ, relates to the density of the contributing users, and (ii) the required stability of the group depends on the frequency of the measurements. We set both parameters according to the Ambiciti application's behavior that supports the crowdsensing of urban noise measurements. Precisely, as the dataset is sparse, we set δ to 4, i.e., handling groups starting at 5 members, and noise measurements are by default taken every 5 minutes leading to set D_{min} to 5 minutes.

4.5 Performance Evaluation

We evaluate the performance of our solution by assessing: (i) the impact on the power consumption of the device running *BeTogether* in the background, (ii) the performance of the context-aware grouping, and (iii) the potential benefit of the collaborative crowdsensing at the edge from the standpoint of data quality and communication cost. Notably, our empirical experiment is based on a one-year dataset obtained from the Ambiciti application for urban pollution monitoring.

4.5.1 Power Consumption

Wi-Fi TX	Wi-Fi Scan	Cellular TX	Light Sensor	GPS RX
173 mA	2 mA	186 mA	0.2 mA	60 mA

Table 4.4: Active power of components for Nexus 5X

We estimate the power consumption on a single crowdsensor theoretically. Table 4.4 shows the power of the main components used in our crowdsensing middleware solution. The reference values come from the Google Nexus 5X smartphone, as accessible by the Android OS[8] according to the power profile provided by the manufacturer. Herein, we select the light sensor to represent a regular sensor, which involves a power consumption comparable to that of most sensors [47, 48]. Note that the power consumption associated with the communication depends on the transmission duration, which increases linearly with the packet size. We assume that cellular and Wi-Fi Direct communications have the same transmission speed.

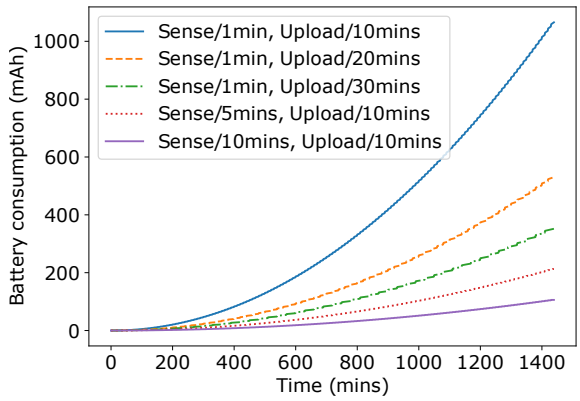


Figure 4.5.1: Power consumption of individual crowdsensing - varying frequencies

Figure 4.5.1 shows the power consumption of a crowdsensor working individually at various sensing and uploading frequencies. The uploading is the most energy-demanding, and energy consumption can be reduced by lowering the uploading frequency. As a comparison, Figure 4.5.2 provides the power consumption of the various crowdsensors contributing to collaborative crowdsensing with regards to a high sensing frequency (every 1 minute) and uploading frequency (every 10 minutes) for the individual case. Results show that the proxy always consumes the most energy due to the cellular transmission that occurs with the cloud, followed by the coordinator that communicates with the nearby devices to distribute the tasks. Other group participants consume much less energy than the individual case, even though this consumption includes the cost related to neighbor discovery and D2D transmission to the coordinator. Our prototype implementation running on several smartphones shows a similar performance.

[8] https://source.android.com/devices/tech/power/values

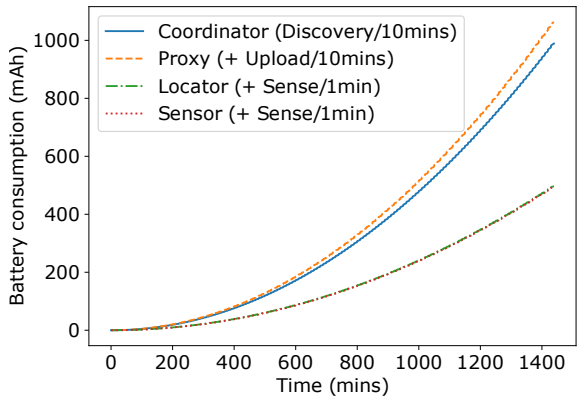


Figure 4.5.2: Power consumption of collaborative crowdsensing

4.5.2 Grouping Evaluation

We apply both theoretical analysis and simulation based on the real world crowdsensing dataset to assess the grouping strategy.

Scalability Analysis

First, we assess the scalability of our grouping algorithm, considering the number of discovery messages exchanged. Such control messages allow to discover neighbors, exchange context information as well as utility information, and assign tasks. This "discovery" leads to the generation of D2D messages (by Wi-Fi Direct and by *BeTogether* together).

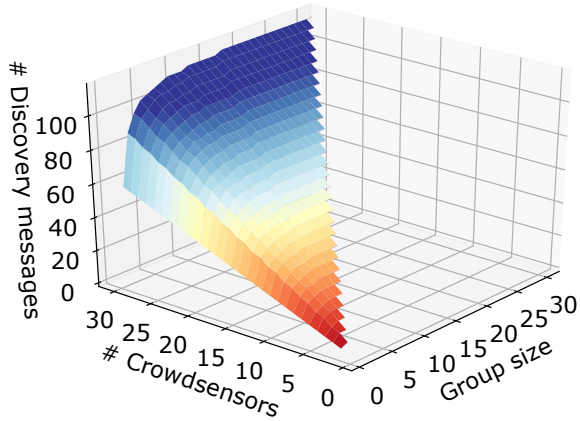


Figure 4.5.3: Amount of D2D traffic generated by *BeTogether*

In Figure 4.5.3, we show the overall control traffic (i.e., number of discovery messages). When the number of crowdsensors is given, the message amount remains relatively flat when the group size increases. More small groups do not introduce more control traffics because the message mainly depends on the number of nodes. When the group size is fixed, the message amount increases linearly along with

the number of crowdsensors. Overall, the control traffic used to discover services increases with the number of crowdsensors while it remains relatively flat when the group size increases.

Analyzing the Crowdsensor Behavior

We leverage a dataset extracted from the database of Ambiciti, which is a cloud-based crowdsensing application available on Google Play[9] and on Apple AppStore[10]. Ambiciti monitors the noise pollution using the smartphone's microphone [185].

> *Ambiciti DATASET* contains 946,573 observations gathered both in-door and out-door in Paris over the year 2017 from 550 crowdsensors. The average data uploading duty cycle is around 5 minutes. Each entry provides the upload time-stamp, the location, the (anonymized) ID of crowdsensor, the noise level, and its measurement bias, a description of the user activity (still, on foot, on the bicycle, in-vehicle, unknown), and whether the device is in-/out-pocket (based on proximity).

Note that due to privacy and commercial concerns, the Ambiciti company shares the data with us in the framework of a collaboration agreement while data cannot be released openly. With 550 crowdsensors for the whole Paris city, the dataset seems sparse. Hence, it does not provide the most suited case for opportunistic collaboration at the edge. Still, this allows us to assess the effectiveness of *BeTogether* even with a sparse deployment.

Based on the dataset, we analyze the stability of the crowdsensors' context, which is used for the grouping configuration, and we consider only the User Activity (UA). Indeed, the Physical Environment (PE) value is limited to in-/out-pocket cases in the dataset, which influences only the sensor utility u_s. The context-awareness is assessed on each day of the year. Starting with the initial location l of any crowdsensor i within the dataset, we consider that i changes the group when it reaches another location l' that is more than the D2D communication range away from l, and repeatedly so with l' as the new reference location.

Figure 4.5.4 then shows the distribution of the duration of the crowdsensor staying within the estimated group above for all the crowdsensors in our dataset according to the device's location: the duration varies from 5 minutes to 55 minutes where we recall that we set $D_{min} = 5$ minutes as the minimum duration of the group. Hence, many crowdsensors remain at the same location long enough to group.

Figure 4.5.5 further compares the three following grouping strategies in terms of the average number of messages sent per crowdsensor every day for discovering nearby crowdsensors. *Periodic* (after every uploading), which is the approach found in related work, *on-demand* (detected by Wi-Fi Direct protocol), which is the de-

[9] https://play.google.com/store/apps/details?id=fr.inria.mimove.quantifiedself&hl=fr

[10] https://itunes.apple.com/us/app/ambiciti/id1080606926?mt=8

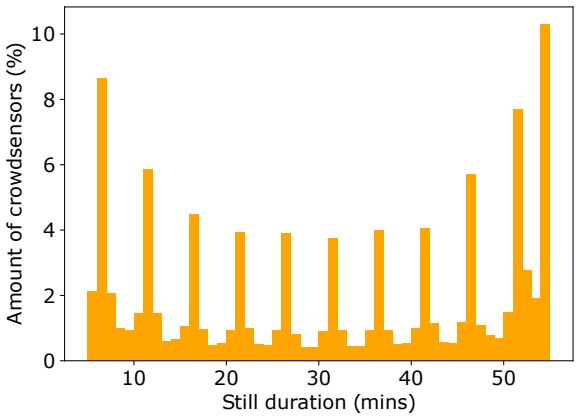


Figure 4.5.4: Distribution of the still duration

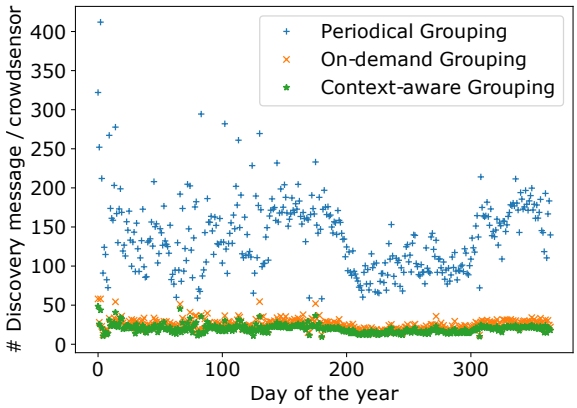


Figure 4.5.5: Comparison of grouping strategies

fault, and *context-aware* (triggered by context change), which accounts for the users' activities. On average, the amount of traffic generated by the on-demand strategy is 80.810% lower than the periodic approach. The traffic amount of our context-aware grouping is 21.844% lower than the on-demand approach.

4.5.3 Efficiency Enhancement

The enhanced efficiency of opportunistic crowdsensing is also evaluated empirically based on the Ambiciti *DATASET*.

Analyzing the Efficiency Gain of Grouping

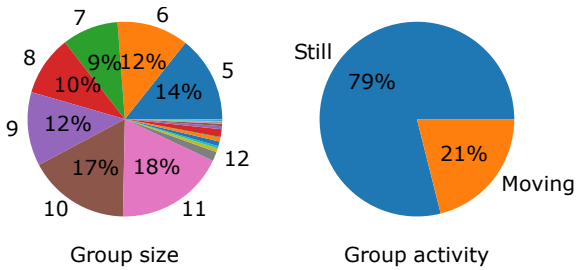


Figure 4.5.6: Distribution of group sizes

To detect the clusters of crowdsensors that are within D2D communication range (at 10m away, re-scaled) from each other, we rely on the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm [54], which handles clusters that are arbitrarily shaped and that are of varying density. According to our parameter $\delta = 4$ that configures groups of size 5 and more, Figure 4.5.6 shows the distribution of the resulting group sizes in Ambiciti dataset, with 79% of the groups being still.

Figure 4.5.7 further compares the average duration of the identified groups depending on whether the context is accounted for or not. Interestingly, results show that even in real-life scenarios (including both still and mobile groups, not only considering still grouping as in Figure 4.5.4), our context-aware grouping strategy finds groups of longer duration, which is up to 3.256 times of the non-context-aware grouping. As the group size grows, the difference between a context-aware and non-context-aware approaches becomes little. The likelihood of grouping co-located crowdsensors having the same context gets high. A decrease of the lifetime is observed for groups of 12 members and more members, partly due to the sparsity of our dataset and the higher probability of members moving away from the group.

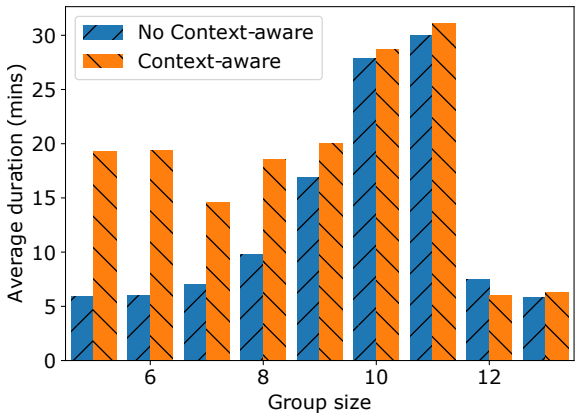


Figure 4.5.7: Lifetimes of collaborative groups

Although the crowdsensing data of the Ambiciti dataset is very sparse in time and space, the context-aware collaborative crowdsensing at the edge still brings benefits in terms of data quality and global resource consumption. It is particularly efficient as the size of the group is large. We compare our *context-aware* collaboration approach with the collaborative crowdsensing *without context-awareness* and with the baseline individual crowdsensing.

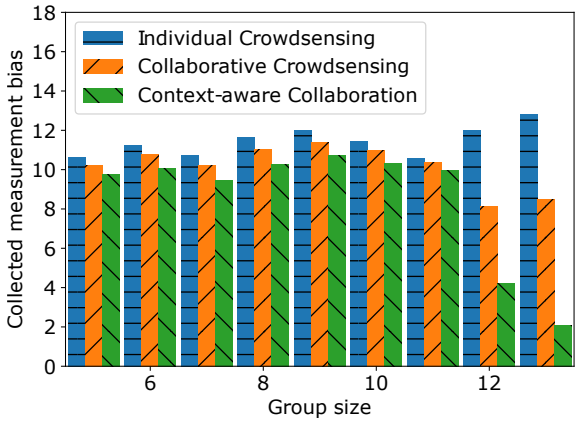


Figure 4.5.8: Impact of *BeTogether* on collected data quality

Figure 4.5.8 shows that our *context-aware* collaboration approach delivers the best data quality: the collected measurement bias is reduced by up to 615% (respectively 407%) compared to the individual (respectively non-context-aware collaborative) crowdsensing approach. The better quality is because of the selective sensing performed within each *BeTogether* group, leading to the collection of the most accurate observations rather than a simple average of the observations (non-context-aware collaborative crowdsensing) or all the raw sensing data (individual crowdsensing).

As shown in Figure 4.5.9, the amount of sensing data that our context-aware collaboration approach uploads to the cloud are reduced by up to 197% compared

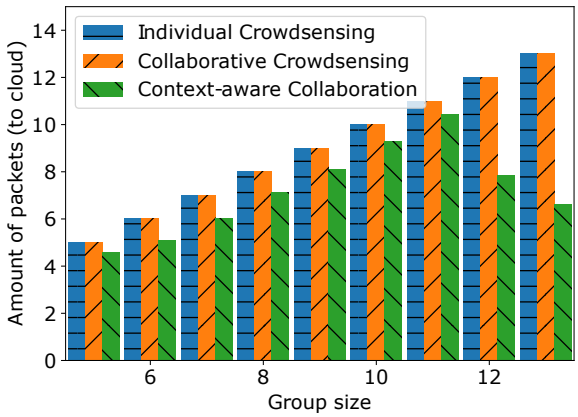


Figure 4.5.9: Impact of *BeTogether* on uploaded data traffic

to both the individual and non-context-aware collaborative crowdsensing approach. There are two reasons for this: (i) collaborative crowdsensing uploads only the aggregated data (e.g., concatenated hash tables) via the group proxy rather than uploading the raw data supplied by each crowdsensor as in the individual crowdsensing; (ii) our context-aware collaboration also filters out the sensing data of low quality and collected in-pocket, which further reduces the amount of data aggregated and uploaded to the cloud.

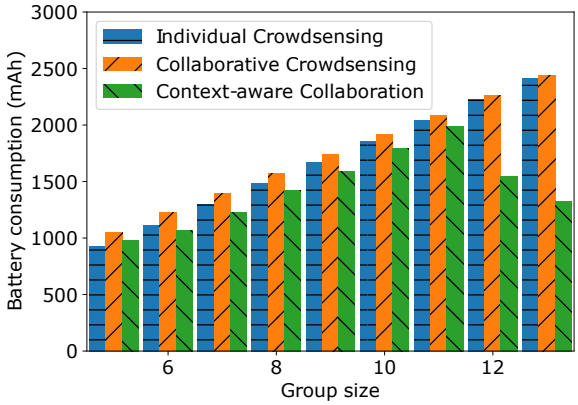


Figure 4.5.10: Impact of *BeTogether* on overall battery consumption

Finally, Figure 4.5.10 focuses on the power consumption for one hour (in mhA), which is associated with both Wi-Fi Direct transmission (assuming that all the tasks are distributed, which is the most expensive) and cellular transmission. Our *BeTogether* solution reduces the energy consumed by up to 181% (respectively 183%) compared to the individual (respectively non-context-aware collaborative) crowdsensing approach.

Overall, evaluation results show that the context-aware collaboration among crowdsensors achieves a better data quality at lower sensing and transmission cost. When the group size is large, the tasks can be assigned to more devices, which

results in less task duplication and more filtering and aggregation.

4.6 Discussion

Opportunistic crowdsensing shows excellent potential for monitoring the physical environment easily and flexibly by empowering people to contribute as part of their daily life. While the growing participation of citizens helps to cover urban-scale areas, it also necessitates to ensure the quality of the knowledge gathered from crowdsensors and to keep to a minimum the resource consumed by the end devices. Research has shown that the efficiency can be enhanced at the very edge by leveraging *ad hoc* collaboration among co-located end devices.

In this chapter, we introduced the *BeTogether* middleware-layer solution that implements a context-aware collaborative group for the crowdsensors to enhance the quality of the data transmitted to the cloud while reducing the on-device communication cost and power consumption. For this purpose, we integrate the context inference including *ContextSense* presented in Chapter 3 and apply a set of utility functions that assess to which extent a crowdsensor should carry out a given crowdsensing task to achieve a balance between the benefit for all (the group) and the related cost for the crowdsensor. We also presented the implementation of the *BeTogether* prototype, which builds upon the Wi-Fi Direct D2D communication technology for the creation and management of the *ad hoc* collaborative group. The evaluation of the *BeTogether* solution using a large crowdsensing dataset from Ambiciti shows that our context-aware collaboration strategy improves the quality of the sensing data transmitted to the cloud as well as reduces the on-device resource consumption, both locally and globally. One reason for the success of *BeTogether* is that it avoids redundant work across co-located crowdsensors, which includes, e.g., uploading and positioning as well as inaccurate sensing.

The group-wise strategy of *BeTogether* is particularly well-suited for crowdsensing within crowded/dense areas where people are clustered on a community scale, for instance, within public spaces. However, human behavior is highly varied and many do not stay near each other for an extended period of time. During the investigation on the Ambiciti dataset, we have found that there exist many crowdsensor pairs that encounter each other very briefly. Such brief encounters typically occur when people are traveling and moving quickly across urban spaces. Clearly, *BeTogether* cannot be applied to crowdsensors pairs of this kind, since their interactions are too brief to be detected as a collaborative group. Therefore, these crowdsensors require a different collaboration strategy. Presumably, crowdsensors of this kind have been sensing from different urban areas until the moment of their encounter. This new case led us to introducing, in the following chapter, a collaborative data analysis suitable for sensing at a sparse and urban scale.

Chapter 5

IAM: Interpolation and Aggregation on the Move

Contents

5.1 Distributed Data Analysis

D2D collaboration enhances mobile computing by enabling the sharing of heterogeneous computing and communication resources among powerful end devices [205, 34]. This is in particular the case of the *BeTogether* solution presented in Chapter 4, which enhances the crowdsensing efficiency in the phase of data collection. *BeTogether* has shown that collaboration among clustered crowdsensors improves the overall system efficiency (including both infrastructures and end devices) in terms of data quality and resource consumption [49]. *BeTogether* also allows end users to share knowledge effortlessly and with little or even no cost. In practice, *BeTogether* leverages the co-located crowdsensors that stay together while observing/sensing the same phenomenon.

Nevertheless, a robust collaborative crowdsensing procedure must take into consideration the alternative cases where crowdsensors make only momentary contact, i.e., when users only encounter each other very infrequently. In such cases, collaboration can only occur within a short window of time, hence why we introduce the pair-wise collaboration system. At the very least, the risk of losing anonymity is lowered [90], meaning that end users are more likely to be willing to share their information/data during these brief encounters. We assume that these crowdsensors have been sensing different urban areas and that they maintain complementary observations when they meet, i.e., each crowdsensor that is moving across the city monitors the phenomenon through its own trajectory. Collaboration may then take place during the data processing phase rather than the data collection phase, such a collaboration also requires the temporary caching of data for analysis. In fact, one of the challenges of sensing data analysis comes from the uneven distribution of observations [193]. Also, the sensing coverage over a city is limited by spatio-temporal behavior of the user [159, 97, 210, 37].

State-of-the-art crowdsensing systems address the shortcoming of sparse data through the centralized analysis –covering aggregation and interpolation– of observations provided to cloud infrastructure servers [106, 142, 62, 61, 184]. The centralized solution implemented then severely limits the adoption of crowdsensing for environmental monitoring due to the resulting resource and financial costs, and also introduces user privacy leak (e.g., mobility extraction) [173]. We argue that enabling fully decentralized collaboration, including the underlying large-scale data analysis, is key to the democratization of environmental monitoring using crowdsensors. In this chapter, we focus on the collaborative data analysis and introduces the *IAM* (*Interpolation and Aggregation on the Move*) solution. Concretely, *IAM* allows a pair-wise collaboration for crowdsensing: crowdsensors interpolate data, and aggregate their respective contributions to the observations of the phenomenon in a collaborative way. The intent is to overcome the spatio-temporal sparsity and to limit –or even avoid– the use of a centralized infrastructure server.

There are many interpolation methods for inferring spatio-temporal phenomena, and the smartphone is becoming increasingly powerful to perform such advanced

tasks. In particular, we have introduced *BeTogether* that enables the discovery of nearby crowdsensors and the exchange of data between peers, making the collaboration feasible [49]. Some crowdsensing systems already exploit the pair-wise collaboration of crowdsensors as they meet [196, 195, 44]. However, the collaboration primarily deals with handling the relay of data, while deployed static edge servers are in charge of the distributed data aggregation. Our approach leverages the advantage of the former and overcomes the disadvantage of the latter: it implements an opportunistic data relay and analysis on the move, across the crowdsensors on the move.

In summary, this chapter makes the following contributions to enable collaborative data processing, which covers more scenarios in the collaborative crowdsensing at the edge:

1. A *fully distributed approach to the aggregation and interpolation of crowdsensing data on the move* (Section 5.3), which exploits the smartphone's capability to perform 3D tensor completion. In particular, we thoroughly analyze and compare available state-of-the-art interpolation methods, which leads us to leverage the Gaussian Process Regression that produces the most cost-effective inference for spatio-temporal phenomena along with an estimation of the inference uncertainty (Section 5.3.1). Another advantage compared to alternative interpolation approaches (i.e., ordinary kriging & tensor decomposition) is that all the parameters that are specified can be learned, and the most relevant kernel may be selected. Finally, only a small portion of the interpolation is running on each crowdsensor.

2. A *distributed lightweight and quality-aware aggregation strategy based only on linear operations* (Section 5.3.2) that combines the tensors that each crowdsensor establishes autonomously. Such a linear aggregation takes place opportunistically across the end devices, and possibly at the server if the D2D communication does not allow covering all the end devices over the given time window. That is, when crowdsensor peers get in a shared D2D communication range, our algorithm selects one of them to aggregate their interpolated tensors and further relay the new tensor. At the end of a predefined time window, the crowdsensor uploads its tensor to the server unless it has previously met a peer that takes care of the relay. A key aspect of the proposed approach is that the aggregation is much less resource consuming (in time and space) than the interpolation, and the server performs only the aggregation needed to fill the gap between the network islets that the contributing crowd forms through D2D communication.

3. A *prototype implementation* (Section 5.4) of the *IAM* solution and its *performance evaluation using a one-year crowdsensing dataset* (Section 5.5). The evaluation shows that *IAM* aggregates a global knowledge that is both robust and accurate compared to the centralized approach. Based on our empirical evaluation, we select the best kernel, which is characterized by a fairly

good balance between accuracy and resource consumption. Most importantly, compared to the centralized approach and baseline relay-based aggregation mechanisms, *IAM* significantly lowers the communication, computation and financial costs of crowdsensing-based environmental monitoring.

Following, we first surveyed the background work regarding the baseline interpolation and aggregation methods. Then we motivated enabling distributed and pair-wise collaborative analysis, as opposed to the state-of-the-art solutions relying on cloud/edge servers.

5.2 Related Work

Sensing data fusion deals with the combination of observations so as to enhance the quality of the knowledge we gather from the sensors. In the specific case of fusing data from mobile crowdsensors and assuming that all the crowdsensors are trusted to provide equally accurate measurements, the supporting algorithms serve the two following functions: (i) *aggregating* together the measurements associated with related observations, and (ii) *interpolating* the missing measurements to overcome the sparse contribution coverage.

5.2.1 Data Aggregation

Related crowdsensing observations are usually aggregated using an average [173, 107] or a weighted average [192, 88] function, although more complex functions may be found in the literature depending on the observed phenomenon [85]. The aggregation may be executed either on the cloud, or in a distributed way –at least partially– for which most crowdsensors provide the necessary computing resources. We undertake the latter decentralized approach in our work so as to benefit from ubiquitous computing and in particular limit the dependence on –and related resource and financial costs due to the usage of– a cloud/edge infrastructure. The challenge is therefore to perform a distributed aggregation in a way that both (i) delivers an overall accurate knowledge, and (ii) incurs bearable resource consumption for the end devices. To achieve so, we use the *Average* aggregate function that is duplicate-sensitive and decomposable; in particular, a batch aggregation is equivalent to several pairwise aggregations.

The distributed aggregation protocol may rely on either a structured (e.g., hierarchical, ring-based.) or unstructured (e.g., flooding, random walk, gossip.) routing; it may even implement a combination of the two. Our solution is based on the unstructured random-walk routing that matches the mobility behavior of opportunistic crowdsensors. That is, a crowdsensor exchanges its observations with another crowdsensor as they get in the D2D communication range of each other, so that one of them aggregates the two sets of observations, and in turn repeats the process as it

meets a new crowdsensor. The protocol stops when reaching a predefined criterion, which is a given time-window in our case (see Section 5.4).

The distributed aggregation of sensing data based on random walk routing usually relies on probabilistic methods to select the aggregation node, due to the stochastic nature of the process [85]. Instead, at each meeting, we select the aggregation/relay node according to the data quality (see Section 5.3.2), which is a relevant selection criterion to achieve an accurate aggregation and to promote as far as possible a more significant number of P2P aggregations. Still, the distributed aggregation protocol that deals with the truth discovery [121] must complement the interpolation of missing values [191].

5.2.2 Data Interpolation

There exist various eligible approaches to the interpolation of crowdsensed observations, wherein the challenge is to overcome both the related data sparsity and computing complexity. In statistical geography, *multivariate interpolation* and *spatial interpolation* play an important role as they enable modeling a large-scale phenomenon (and producing a digital elevation model) provided a set of observations/points. Many interpolation techniques may be applied, depending on the characteristics of the observed data points [30].

Ordinary kriging [186, 142, 61] is one of the methods that is widely applied as it supports a fine-grained interpolation at each location over a 2D space. Ordinary kriging aims at estimating the value $\hat{y}(x)$ at an unobserved and arbitrary location x based on known observation values $y(x_1), \cdots, y(x_n)$. The objective lies in finding the weights $w_1(x), \cdots, w_k(x)$ at any location x_k so that the prediction variance $Var(y(x) - \hat{y}(x))$ is minimal. Ordinary kriging introduces an estimation technique where the phenomenon under study is assumed to be a realization of a random function characterized by a specific spatial covariance. It performs well as long as the observation is uniformly distributed. In particular, the performance dramatically degrades for large amounts of missing data [5], and the computational cost gets prohibitively high [52]: the time complexity is $\mathcal{O}(n^3)$ with n denotes the number of observations.

Compressive sensing is a recent alternative approach to the interpolation of data for the production of phenomena maps, while dealing with sparse observations [52]. Compressive sensing requires very few sample values by leveraging spatial and temporal correlations among the data sensed in several sub-areas to infer the data for the uncovered area [193]. It has been used to infer urban-scale physical phenomena from crowdsensed observations stored in a 2D matrix [207, 192, 61]. Modeling the observations into a 2D matrix then involves representing the geographical location in one dimension and the time in the other to deal with spatio-temporal interpolation. However, physical phenomena often have more than two modes of variation and are therefore best represented as multi-dimensions observations, which leads to address compressive sensing using 3D tensors [5, 222, 159, 96]. The tensor completion

problem is usually solved by tensor decomposition to perform the approximation [222, 159, 96, 63]. The computational cost of most existing algorithms for tensor decomposition increases exponentially with the tensor order. For instance, the computationally cheapest algorithm [156] associated with a 3-order $I \times J \times K$ tensor that based on ALS (Alternating Least Square) has a computational cost of O($RIJK$), with R being the estimated rank. In addition to the high cost associated with interpolating 3D tensors, one major drawback is that it trades uncertainty for efficiency, as there is no confidence interval associated with the inference.

Multiple regression [73, 202, 149] is another approach to infer missing values, which transforms the interpolation problem into multiple regressions. For instance, the geographically weighted regression has coefficients that are not fixed but depend on the observations' geographical coordinates. The hypothesis is that the closer geographically two observations are, the more similar the influence of the explanatory variables on the dependent variable, i.e., the closer the coefficients of the critical parameters of the regression. OLS (Ordinary Least Squares) can be used for estimating the unknown parameters in the regression model. The idea of regression inspired us to use the GPR (Gaussian Process Regression) to solve the tensor completion problem. GPR is a well-known, and general approach applied in data science [124], which supports a non-parametric and interpretable Bayesian model. GPR naturally supports multi-dimension, and it is efficient. Compared to ordinary kriging and tensor decomposition, GPR gives more insight into the data by offering many choices of non-stationary covariance functions characterized by different smoothness, which can be tested and evaluated. GPR further enables estimating the level of uncertainty associated with the produced model. We compare the three interpolation approaches in the evaluation (Section 5.5), where we show that GPR is the best suited to support distributed interpolation across mobile crowdsensors.

5.2.3 Centralized vs Distributed Data Fusion

The great majority of crowdsensing platforms applies centralized analysis to the sensing data: the cloud/edge server first aggregates the raw crowdsensing measurements and then interpolates the missing observations. Existing platforms adopt various interpolation and aggregation methods, while considering different sensor types (static vs. mobile). For instance, one of the solutions leverages ordinary kriging on the cloud to generate a map from the measurements contributed by a combination of static sensors and mobile crowdsensors [62, 61]. Some platforms [96, 97, 159, 210] exploit additional urban data sources (e.g., road networks, check-in data) to infer the phenomenon, which is represented as a 3D tensor completion using centralized tensor decomposition. Rather than integrating more data sources, the crowdsensing application may leverage historical data to improve the accuracy of the generated map. In the case of deployment downscale, supervised learning is used to select the best historical map [36]. Then, a multi-Output Gaussian Process serves as a unified map generation model, which takes multiple instances (a current sparse instance

and an appropriate historical dense instance) to generate an improved air quality map.

The distributed interpolation and aggregation of sensing data have deserved less attention than the centralized counterparts, with most solutions targeting WSN. A localized, distributed interpolation & aggregation scheme based on kriging is introduced in [183] for a tree-structured WSN; it allows inferring a phenomenon over the holes that the static WSN does not cover. In [209], a hierarchical WSN is organized such that sensors send the measurements to their respective cluster heads, which in turn encode the sparse measurements and use compressive sensing (matrix decomposition) to interpolate the overall phenomenon.

The work that is the most related to ours is the edge-mediated spatial-temporal crowdsensing proposed in [212]. The solution relies on a trusted edge server (e.g., a deployed cloudlet) that coordinates a few crowdsensors. In practice, crowdsensors independently perform a part of matrix decomposition using stochastic gradient descent, and they exchange factor vectors with the crowdsensors that are in the same WLAN (Wireless Local Area Network). Each edge server is responsible for a batch of crowdsensors that are fully connected to perform iterative optimization and ultimately recovers an interpolated map for the sub-area. The set of crowdsensors and the edge server must remain connected and communicate over multiple rounds to establish a single interpolation. Furthermore, as stated previously, the tensor decomposition that applied is less accurate than GPR, which we leverage within *IAM*.

Different from previous work, the *IAM* solution: (1) leverages 3D tensors that embed more information than the 2D matrix data model, and efficient Gaussian Process Regression for interpolation, (2) supports both interpolation and aggregation at the end device in a resource-efficient way, so as to limit the dependence on the infrastructure (edge/cloud server) at a bearable additional resource (including energy) cost for the users' crowdsensing devices, and (3) implements opportunistic P2P aggregations, which benefit from the encounter with other crowdsensors and is hence not constrained by any hierarchical/tree network structure.

5.3 Lightweight Decentralized Crowdsensing Data Fusion

The *IAM* solution for decentralized crowdsensing takes benefit of today's smartphones capability, while limiting the resulting additional resource consumption on devices. That is, *IAM* implements lightweight collaborative sensing data fusion across the participating crowdsensors as they are in the D2D communication range.

Problem Statement

Let *IAM* be deployed over m mobile crowdsensors (e.g., smartphones) embedding (built-in or connected) sensors providing measurements related to the physical phenomenon $\mathcal{P}$. We further assume that the m crowdsensors are all trustworthy and provide equally accurate measurements (i.e., they underwent appropriate calibration [166] prior to contribute measurements). *IAM* supports the periodic monitoring of $\mathcal{P}$ over (a possibly large) area $\mathcal{A}$ and a given period $\mathcal{D}$ using the contributions of the m crowdsensors.

We represent the data that each crowdsensor collects as a concise 3D tensor. The first two dimensions refer to the spatial space (i.e., latitude and longitude), and the third one refers to the temporal domain (time). In particular, we discretize the target region $\mathcal{A}$ into a $I \times J$ areas, which are cells of equal spatial sizes. We also discretize $\mathcal{D}$ into K time slots of equal durations. Then we denote $\mathcal{Y}_s \in \mathbb{R}^{I \times J \times K}$ the tensor that crowdsensor s ($1 \leq s \leq m$) maintains. The entry $y_s(x) \in \mathbb{R}$ at position $x := (i, j, k) \in \mathbb{R}^3$ is the average of the measurements collected by s over the area cell indexed by (i, j) during the time interval indexed by k. The value $y_s(x)$ is null if s does not sense/observe at position x. In other words, any crowdsensor s contributes a tensor $\mathcal{Y}_s$ that provides a sparse/incomplete observation of the physical phenomenon according to its behavior.

IAM promotes the opportunistic combination of various tensors $\mathcal{Y}_s$ from multiple contributing crowdsensors s. It combines interpolation and aggregation, so as to compute an overall $\mathcal{Y}$ for a spatio-temporal characterization of phenomenon $\mathcal{P}$ over area $\mathcal{A}$ and duration $\mathcal{D}$. It further achieves so in a way that limits the resource consumption on end devices and is, in particular, energy-efficient. As a base design principle, crowdsensors first apply interpolation over their local tensor $\mathcal{Y}_s$ before engaging in the collaborative aggregation, which is key to the –both local and global– resource-efficiency of the *IAM* solution, while supporting the computation of a globally accurate knowledge.

5.3.1 Spatio-temporal Interpolation

Given a sparse tensor $\mathcal{Y}_s$ resulting from the averaging of the local measurements collected at crowdsensor s over area $\mathcal{A}$ and duration $\mathcal{D}$, the interpolation allows completing the tensor by estimating missing cells. The resulting (denser) tensor is denoted as $\hat{\mathcal{Y}}_s$. The quality of the estimation can be established based on the approximation error (i.e., residuals). The overall residual is then given by $\mathcal{E} = \|\mathcal{Y} - \hat{\mathcal{Y}}\|$.

Let Ω be the set of *observed cells* on a given crowdsensor, that is, the cells to which the crowdsensor contributed observations. The Boolean mask tensor $\mathcal{M} \in \mathbb{B}^{I \times J \times K}$ is defined such that $m(x) = 1$ if there is a corresponding value at point $x \in \Omega$, and $m(x) = 0$ otherwise. Thus, $\mathcal{M} * \mathcal{Y}$ provides a tensor resulting from actual observations (i.e., ground truth as sensed). When estimating $\hat{\mathcal{Y}}$ based on a sparse tensor $\mathcal{Y}$ with mask $\mathcal{M}$, we seek to minimize the following loss function,

which is associated with the approximation $\hat{\mathcal{Y}}$:

$$J(\mathcal{Y}, \hat{\mathcal{Y}}) := \frac{1}{2} \sum_{x \in \Omega} e(x)^2 = \frac{1}{2} \|\mathcal{M} * (\hat{\mathcal{Y}} - \mathcal{Y})\|^2$$

where: $e(x)$ is the residual at point x, $\|\cdot\|$ denotes the Euclidean norm of a tensor, and $*$ represents the element-wise multiplication.

Recall that the function $y : \mathbb{R}^3 \mapsto \mathbb{R}$ maps an arbitrary point $x := (i, j, k)$ to its cell value $y(x)$. Following the assessment of the eligible interpolation methods summarized in Section 5.2.2, we leverage the *Gaussian Process Regression* [124] to compute $\hat{\mathcal{Y}}$ out of $\mathcal{Y}$. That is, we assume that y follows a Gaussian Process (Gaussian distribution over functions), namely:

$$y(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$$

where: $\mu(x) = \mathbb{E}[y(x)]$ refers to the mean function, and $k(x, x')$ is the covariance matrix, i.e., the kernel of the GPR, which verifies $k(x, x') = \mathbb{E}[(y(x) - \mu(x))(y(x') - \mu(x'))]$.

The kernel is a crucial ingredient of GPR as it encodes the notion of similarity between two nearby data points x and x', on the basis that observations that are closer to each other (on Euclidean distance) are likely to have a higher correlation. Thereby, the actual measurements that are close to an approximated observation are assumed to be highly informative for the inference at that point. Various families of kernels exist (see [163] for an overview). In our case, the Matérn [144] induced the lowest error and execution time compared to alternative kernels (see Section 5.5). We detail below the computation of GPR with Matérn kernel.

The Matérn class of kernel determines to which extent two points x and x' are correlated:

$$k(x, x') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{\|x - x'\|}{l} \right)^{\nu} K_\nu \left(\sqrt{2\nu} \frac{\|x - x'\|}{l} \right)$$

where: Γ is the *Gamma function*, K_ν is the modified *Bessel function* of the second kind, and l and ν are non-negative parameters of the covariance. These hyper-parameters are optimized during the regression process.

Considering a regression that aims at establishing $y = y(x) + \epsilon$ where the function $y(x)$ follows a Gaussian Process ($y(x) \sim \mathcal{GP}(\mu, k)$), and the noise ϵ is additive, independent and corresponds to an identical Gaussian distribution: $\epsilon \sim \mathcal{N}(0, \sigma_e^2)$. Following, given n observed points at locations $X = [x_1, \cdots, x_n]^\top$ and the corresponding observed values $Y = [y_1, \cdots, y_n]^\top$, the joint distribution of $y(X) = [y(x_1), \cdots, y(x_n)]^\top$ follows:

$$[y(x_1), \cdots, y(x_n)]^\top \sim \mathcal{N}(\boldsymbol{\mu}(X), \mathcal{K}(X, X))$$

where: the mean vector $\boldsymbol{\mu}(X) = [\mu(x_1), \cdots, \mu(x_n)]^\top$, and $\mathcal{K}(X, X)$ is a $n \times n$ covariance matrix with $\mathcal{K}_{ij} = k(x_i, x_j)$.

Given m unobserved points at locations $X^* = [x_1^*, \cdots, x_m^*]^\top$, our interpolation consists in inferring the missing values $Y^* = y(X^*)$. The observed values Y and the unobserved/inferred values Y^* together follow:

$$\begin{bmatrix} Y \\ Y^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}(X) \\ \boldsymbol{\mu}(X^*) \end{bmatrix}, \begin{bmatrix} \mathcal{K}(X,X) + \sigma_e^2 \mathcal{I} & \mathcal{K}(X,X^*) \\ \mathcal{K}(X,X^*)^\top & \mathcal{K}(X^*,X^*) \end{bmatrix} \right)$$

where: $\boldsymbol{\mu}(X^*) = [\mu(x_1^*), \cdots, \mu(x_m^*)]^\top$. $\mathcal{K}(X^*,X^*)$ is a $m \times m$ matrix and $\mathcal{K}(X,X^*)$ is a $n \times m$ matrix wherein $\mathcal{K}(X,X^*)_{ij} = k(x_i, x_j^*)$. The $n \times n$ identity matrix corresponds to $\mathcal{I}$.

According to the Bayes theorem and based on the known prior distribution $p(Y)$ and the above joint probability distribution $p(Y,Y^*)$, we can deduce the posterior probability:

$$p(Y^*|Y) = \frac{p(Y|Y^*)p(Y^*)}{p(Y)} = \frac{p(Y,Y^*)}{p(Y)}$$

Then, knowing X and Y from observations, the inference Y^* on X^* also follows a Gaussian distribution:

$$p(Y^*|X^*,X,Y) = \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}^2 + \sigma_e^2 \mathcal{I})$$

According to the marginalization and conditional distribution theorem [79], the mean and variance of Y^* are respectively estimated as:

$$\begin{aligned} \hat{\boldsymbol{\mu}} &= \mathcal{K}(X,X^*)^\top [\mathcal{K}(X,X) + \sigma_e^2 \mathcal{I}]^{-1} Y \\ \hat{\boldsymbol{\sigma}}^2 &= \mathcal{K}(X^*,X^*) - \mathcal{K}(X,X^*)^\top [\mathcal{K}(X,X) + \sigma_e^2 \mathcal{I}]^{-1} \mathcal{K}(X,X^*) \end{aligned}$$

where: the mean vector $\hat{\boldsymbol{\mu}}$ and variance vector $\hat{\boldsymbol{\sigma}}^2$ are computed using the Cholesky decomposition [163].

$\mathcal{Y}_s$	Sensing data tensor, equaling to $\hat{\mathcal{Y}}_s$ after the interpolation
$\mathcal{M}_s$	Mask tensor, indicating observed/inferred elements in $\mathcal{Y}_s$
$\boldsymbol{\Sigma}_s^2$	Variance tensor, recording the $\hat{\sigma}(x)$ of each element in $\mathcal{Y}_s$
n_s	Merge count, recording how many times $\mathcal{Y}_s$ has been updated

Table 5.1: Variables maintained at a crowdsensor s

In summary, once a GPR model is trained, the inferred mean value $\hat{\mu}(x)$ and its variance $\hat{\sigma}(x)$ of any input point x are generated (as regression process). The complete approximation tensor $\hat{\mathcal{Y}}$ is produced using $\hat{\mu}(x)$ as $\hat{y}(x)$ cell value, while a variance tensor $\hat{\boldsymbol{\Sigma}}^{\mathbf{2}} \in \mathbb{R}^{I \times J \times K}$ is also maintained, in which each element $\hat{\sigma}^2(x)$ refers to the variance of corresponding $\hat{y}(x)$. Table 5.1 summarizes all the necessary variables maintained at each crowdsensor s.

GPR is computationally demanding as the training scales in $O(n^3)$ with n being the number of *observed cells*. Thus, applying such a regression over the overall dataset on the cloud incurs significant computation costs. As an alternative, *IAM* distributes the training and inference load over the crowdsensors, which further results in a limited computational cost on the device due to the relatively low number of contributed cells (see Section 5.5).

5.3.2 Opportunistic Collaborative Aggregation

Upon the P2P meeting (i.e., discovery through D2D communication) of two crowdsensors s and s', *IAM* selects one of them to aggregate their respective tensors $\hat{\mathcal{Y}}_s$ and $\hat{\mathcal{Y}}_{s'}$. Assuming the selected crowdsensor is s, then $\hat{\mathcal{Y}}_s$ is updated as the aggregation result of $\hat{\mathcal{Y}}_s$ and $\hat{\mathcal{Y}}_{s'}$, and $\hat{\mathcal{Y}}_{s'}$ is set to *null*. Then, s may relay $\hat{\mathcal{Y}}_s$ when it meets another crowdsensor, or till the current time window $T_{+\mathcal{D}}$ expires.

The P2P meetings that *IAM* fosters correspond to a stochastic process since crowdsensors meet each other in an opportunistic way based on their own mobility. Upon such a meeting, a straightforward aggregation approach would consist of randomly selecting one of the two crowdsensors to take in charge of the aggregation and relay the tensors. We assume that all the crowdsensors have equal resource budgets. However, the crowdsensor that is the best suited to act as the relay node is the one that will meet more crowdsensors in the future. Indeed, we hypothesize that such a crowdsensor will produce a denser aggregated data tensor and thus a better inference quality. Still, we do not rely on the costly monitoring of the mobility profiles of the contributing users to select the relay crowdsensor [66, 35]. Instead, we use the respective quality of inference of the crowdsensor tensors for the relay selection. The evaluation results confirm the relevance of the criterion, and suggest it is an indicator of future meeting occurrences (see § 5.5). Precisely, we leverage the inference quality as defined by the following asymmetric and positive loss function:

$$D(\hat{\mathcal{Y}}_s, \hat{\mathcal{Y}}_{s'}) := \frac{\|(\mathcal{M}_{s'} * \neg\mathcal{M}_s) * (\hat{\mathcal{Y}}_{s'} - \hat{\mathcal{Y}}_s)\|^2}{2\|\mathcal{M}_{s'} * \neg\mathcal{M}_s\|^2}$$

where: $\neg$ corresponds to the *NOT* Boolean operation; and $\mathcal{M}_s$ and $\mathcal{M}_{s'}$ correspond to the mask tensors of s and s' respectively.

Algorithm 4 introduces the aggregation procedure that crowdsensor s runs upon meeting with crowdsensor s' (with s' running the same algorithm). Only one of the two should perform the actual aggregation (merge the tensors) and act as the relay node, which we call the *mainstay*. The crowdsensor with the lowest loss function selects itself (Lines 1-2) as the *mainstay*. The other crowdsensor no longer maintains its local data (Lines 3-4) so that there is no duplicated uploading. We highlight that the selection of the mainstay aims at optimizing the quality of the data delivered by the distributed aggregation process for which we consider the inference quality as criterion. It is area for future work to increase the overall robustness of the opportunistic aggregation protocol by taking into account additional criteria (e.g.,

Algorithm 4 Asymmetric P2P aggregation at s

Require: local data tensor $\mathcal{Y}_s$, local mask tensor $\mathcal{M}_s$, local variance tensor $\mathbf{\Sigma}^2_s$, local merge count n_s
Input: remote data tensor $\mathcal{Y}_{s'}$, remote mask tensor $\mathcal{M}_{s'}$, remote variance tensor $\mathbf{\Sigma}^2_{s'}$, remote merge count $n_{s'}$
1: **if** $D(\mathcal{Y}_s, \mathcal{Y}_{s'}) < D(\mathcal{Y}_{s'}, \mathcal{Y}_s)$ **then**
2: $\quad \mathcal{Y}_s, \mathcal{M}_s, n_s, \mathbf{\Sigma}^2_s \leftarrow$ Crowdsensing data tensors merger (s, s') – see Algorithm 5
3: **else**
4: $\quad \mathcal{Y}_s, \mathcal{M}_s, n_s, \mathbf{\Sigma}^2_s \leftarrow null$
5: **end if**

mobility behavior, available resource, and fault tolerance). We may leverage state of the art algorithms [196, 195], while still keeping the process energy-efficient.

Algorithm 5 Crowdsensing data tensors merger (s, s') at the mainstay s

Input: data tensor $\mathcal{Y}_s$, mask tensor $\mathcal{M}_s$, variance tensor $\mathbf{\Sigma}^2_s$, merge count n_s
Input: data tensor $\mathcal{Y}_{s'}$, mask tensor $\mathcal{M}_{s'}$, variance tensor $\mathbf{\Sigma}^2_{s'}$, merge count $n_{s'}$
Output: aggregated data tensor $\mathcal{Y}_{ss'}$, mask tensor $\mathcal{M}_{ss'}$, variance tensor $\mathbf{\Sigma}^2_{ss'}$, merge count $n_{ss'}$
1: $\mathcal{Y}_{ss'} \leftarrow \mathbf{0}^{I \times J \times K}$
2: $\mathcal{Y}_{ss'} += \frac{n_s \mathcal{Y}_s + n_{s'} \mathcal{Y}_{s'}}{n_s + n_{s'}} * (\mathcal{M}_s * \mathcal{M}_{s'})$
3: $\mathcal{Y}_{ss'} += \mathcal{Y}_s * (\mathcal{M}_s * \neg \mathcal{M}_{s'})$
4: $\mathcal{Y}_{ss'} += \mathcal{Y}_{s'} * (\mathcal{M}_{s'} * \neg \mathcal{M}_s)$
5: $\mathcal{Y}_{ss'} += (\beta_s \mathcal{Y}_s / \mathbf{\Sigma}^2_s + \beta_{s'} \mathcal{Y}_{s'} / \mathbf{\Sigma}^2_{s'}) / (\beta_s \mathbf{\Sigma}^{-2}_s + \beta_{s'} \mathbf{\Sigma}^{-2}_{s'}) * (\neg \mathcal{M}_s * \neg \mathcal{M}_{s'})$
6: $\mathbf{\Sigma}^2_{ss'} \leftarrow (\beta_s \mathbf{\Sigma}^{-2}_s + \beta_{s'} \mathbf{\Sigma}^{-2}_{s'})^{-1}$
7: $n_{ss'} \leftarrow n_s + n_{s'}$
8: $\mathcal{M}_{ss'} \leftarrow \mathcal{M}_s \vee \mathcal{M}_{s'}$
9: **return** $\mathcal{Y}_{ss'}, \mathcal{M}_{ss'}, n_{ss'}, \mathbf{\Sigma}^2_{ss'}$

Algorithm 5 details the data merge function that the mainstay s runs, provided the tensor $\mathcal{Y}_{s'}$ from s', to compute the new tensor $\mathcal{Y}_{ss'}$. The algorithm distinguishes whether the values from $y_s(x)$ and $y'_s(x)$ at x result from actual sensor measurements or from interpolation, as known from the masks $\mathcal{M}_s$ and $\mathcal{M}_{s'}$:

- *Line 2 – The two values result from actual sensor measurements:* let $y(x)^i$ be the value for a given point x at the i-th averaging step; the incremental average up to the m-th averaging step is defined as: $\overline{y}^m(x) = \frac{1}{m} \sum_{i=0}^{m} y^i(x)$. Assuming that crowdsensor s (respectively s') has collected and averaged m (respectively n) measurements, the aggregated observation is: $\overline{y}^m_s(x) = \frac{1}{m} \sum_{i=1}^{m} y^i_s(x)$ (respectively $\overline{y}^n_{s'}(x) = \frac{1}{n} \sum_{i=1}^{n} y^i_{s'}(x)$). Starting with $\overline{y}^0_{ss'}(x) = 0$ and deduced from the incremental average definition, the merged average of two averages $\overline{y}^m_s(x)$

at step m and $\overline{y}^n_{s'}(x)$ at step n is:

$$\overline{y}_{ss'}(x) = \frac{m\overline{y}^m_s(x) + n\overline{y}^n_{s'}(x)}{m+n}$$

Note that the above merged average is built upon a simple algebraic expression of addition, which works well to fuse the measurements provided by scalar values, e.g., temperature sensors. Such a merged average should be tailored to deal with observations that do not follow an algebraic expression. For example, this is the case when merging the average sound levels provided by mobile crowdsensors, which is the focus of the experiment presented in Section 5.5.

- *Lines 3 & 4 – One value results from actual sensor measurements, and the other is inferred:* the actual sensor measurement is considered to be the ground truth. Thus it is selected over the inferred value.
- *Line 5 – The two values result from two inferences:* the aggregated value is then computed using the Generalized Product-of-Expert of GPR, as detailed in the following section.

5.3.3 Aggregating Multiple Inferences

The *Generalized Product-of-Expert* is a method that allows combining estimated results that have been inferred by several experts (e.g., inferences on several crowdsensors). In particular, it enables weighting the respective importance of the experts according to the reliability of their inference. Let $p_s(y^*|x^*, X_s, Y_s)$ denote the distribution of the measurements for point x^*, which is inferred by the crowdsensor s, knowing the *observed cell* values Y_s at points X_s. Assuming that m crowdsensors aggregate their inference results, the Product-of-Expert for a GPR estimates a value y^* at point x^* according to the following joint distribution [42]:

$$p(y^*|x^*, X, Y) = \prod_{s=1}^{m} p_s^{\beta_s(x^*)}(y^*|x^*, X_s, Y_s)$$

where β_s is a weighting parameter that allows tuning the relative importance of crowdsensor s according to the reliability of its inference.

The aggregation of multiple GPR inferences is a generalized Product-of-Expert, which accounts for multiple inference distributions p_s of an arbitrary point x^*. According to [24], it combines many Gaussian distributions with mean $\hat{\mu}_s(x^*)$ and variance $\hat{\sigma}^2_s(x^*)$ from any crowdsensor s, and the aggregation result is defined as:

$$\hat{\mu}(x^*) = \hat{\sigma}^2(x^*) \sum_{s=1}^{m} \beta_s(x^*)\hat{\sigma}_s^{-2}(x^*)\hat{\mu}_s(x^*)$$

$$\hat{\sigma}^2(x^*) = [\sum_{s=1}^{m} \beta_s(x^*)\hat{\sigma}_s^{-2}(x^*)]^{-1}$$

The generalized Product-of-Expert of a GPR allows merging several inferences (i.e., $m = 2$ inferences in our P2P case) in a cost-effective way, as it is characterized by only a $\mathcal{O}(n)$ time complexity with n being the number of merged points. Our opportunistic aggregation on the move is asymmetric, as captured by the loss function D. The evaluation (see Section 5.5) shows that assigning a greater β to the crowdsensor acting as the mainstay (i.e., resulting in the lesser loss function) leads to a higher aggregation accuracy.

5.4 System Design and Prototype Implementation

The *IAM* solution allows mapping quantitative spatio-temporal physical phenomena through opportunistic data interpolation and aggregation across the participating mobile crowdsensors. Any mobile crowdsensing application dealing with urban environmental monitoring (e.g., noise level, air quality, temperature.) may build upon *IAM* to produce such knowledge in a fully decentralized way. Still, the actual D2D meeting of the contributing crowdsensors within the area under monitoring depends on the size of the target area and the density of the contributing crowd. In practice, the monitoring of large areas requires running an ultimate aggregation process at a server to connect the islets that the crowd cover.

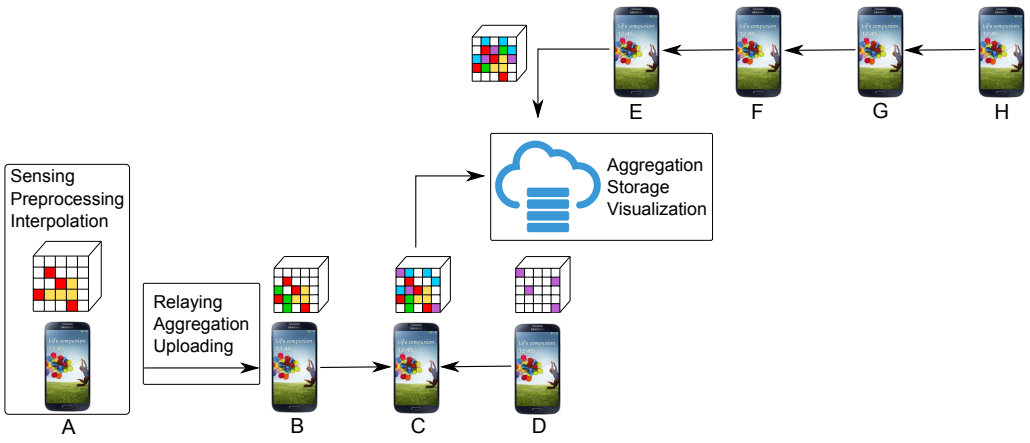


Figure 5.4.1: A *IAM*-based crowdsensing system

Figure 5.4.1 illustrates the resulting operation of a *IAM*-based crowdsensing system. Crowdsensors sense, pre-process and interpolate the data. They relay and aggregate the sensing data in a P2P way using wireless D2D communication (e.g., Wi-Fi Direct or Bluetooth technologies), so as to favor ubiquitous computing and thereby limit the dependence on the server infrastructure. At the end of the monitoring period $\mathcal{D}$, the remaining *mainstays* send their tensors to the server, which fairly composes the tensors using the generalized Product-of-Expert. Focusing on the illustrative figure: C aggregates data from B and D and then uploads to the

server the results of the three aggregations across A, B, D, and itself. Similarly, the distant mainstay E, which aggregates its contributions with the ones of H, G, and F, provides the resulting tensor to the server.

5.4.1 Aggregation Process at Crowdsensors

Algorithm 6 Crowdsensing process at crowdsensor s

Require: previous time window T, current time window $T_{+\mathcal{D}}$
Require: data tensor $\mathcal{Y}_s(T)$ related to previous time window,
current data tensor $\mathcal{Y}_s(T_{+\mathcal{D}})$
1: **while true do**
2: **while** Within $T_{+\mathcal{D}}$ **do**
3: Collect sensing data and fill $\mathcal{Y}_s(T_{+\mathcal{D}})$ //
4: Aggregate $\mathcal{Y}_s(T)$ upon P2P meeting till $\mathcal{Y}_s(T)$ sets to *null*
5: **end while**
6: Interpolate $\mathcal{Y}_s(T_{+\mathcal{D}})$ //
7: **if** $\mathcal{Y}_s(T)$ has not been relayed yet **then**
8: Upload $\mathcal{Y}_s(T)$ to the server
9: **end if**
10: **end while**

Algorithm 6 outlines the periodic process that *IAM* runs on every participating crowdsensor s to compute and relay/upload $\mathcal{Y}_s$. The process iterates over the time windows of duration $\mathcal{D}$ (Lines 1 and 2). Within a given time window $T_{+\mathcal{D}}$, two processes run in parallel: (i) the collection of the measurements provided by the embedded sensors to update the tensor $\mathcal{Y}_s(T_{+\mathcal{D}})$ of the current time window (Line 3), and (ii) the opportunistic aggregation of the local tensor of the previous time window T with one of the peers that s meets (Line 4 – See detail in Section 5.3.2).

At the end of the current time window, the spatio-temporal interpolation is applied to the associated local tensor to infer missing values (Line 6 – See detail in Section 5.3.1). We highlight that the interpolation runs on the end device, only once and before the aggregation process running over the next time window. The one-time interpolation allows: (i) minimizing the number of interpolation occurrences and thereby the resource cost on the device, and (ii) leveraging the locally completed tensor to assess the quality of the local measurements against the ones of the peers that the crowdsensor meets, which determines the *mainstay*.

Finally, still at the end of the current time window $T_{+\mathcal{D}}$, the crowdsensor sends its local tensor to the server, unless it has aggregated and relayed to another crowdsensor (Lines 7-9).

5.4.2 Prototype on the Smartphone

Our *IAM* solution assumes an *ad hoc* framework supporting the opportunistic meeting and collaboration among crowdsensors using D2D discovery and communication, which may be our *BeTogether*[1] solution presented in Chapter 4 [49], or other solutions in [78]. The *IAM* prototype is thus explicitly focused on the implementation of the distributed, collaborative interpolation and aggregation, further targeting Android smartphones/tablets as crowdsensors.

The *IAM* prototype is available at GitHub[2]. It requires a Python 3 environment as well as the following third-party packages for data analytics: NumPy[3] handling multi-dimensional arrays (tensors); Scikit-learn[4] implementing various machine learning algorithms that are used for GPR training and inference; and PyKrige[5] implementing the 3D ordinary kriging interpolation with various standard variogram models. The key components of our prototype implementation are:

- *Pre-processing* reads the crowdsensed data that is stored in a local file, and creates the corresponding tensors $\mathcal{Y}$ and $\mathcal{M}$. The tensor size is an application-specific parameter that is configured to map the target physical phenomenon $\mathcal{P}$ over the chosen $\mathcal{A}$ and $\mathcal{D}$. Precisely, the tensor size depends on the required sensing resolution, the geographical space that needs to be covered, and the time window. Overall, the parsing is characterized by a $\mathcal{O}(n)$ time complexity, where n denotes the number of *observed cells*.

- *Interpolation* creates a GPR model, trains the model based on the $\mathcal{M} * \mathcal{Y}$ *observed cells*, and uses the trained model to infer and produce the approximation tensor $\hat{\mathcal{Y}}$ along with the variance tensor $\mathbf{\Sigma}^2$. The interpolation has a $\mathcal{O}(n^3)$ time complexity with n being the number of *observed cells*.

- *Aggregation* computes the loss function of two tensors from a pair of crowdsensors, makes the aggregation decision, then merges the two tensors into one,and updates the current data, mask and variance tensors $\hat{\mathcal{Y}}$, $\mathcal{M}$ and $\mathbf{\Sigma}^2$ respectively, following Algorithm 4 and Algorithm 5. The aggregation process on each crowdsensor has a $\mathcal{O}(p)$ time complexity, where p represents the number of P2P meetings.

IAM supports the opportunistic aggregation of the sensing data along with the interpolation of a physical phenomenon provided relevant measurements from the crowdsensors. Observation values may relate to various physical phenomena (e.g., air temperature, sound level, illuminance, humidity, air pressure.).

[1] https://github.com/sensetogether/BeTogether
[2] https://github.com/sensetogether/IAM
[3] https://numpy.org
[4] https://scikit-learn.org
[5] https://pykrige.readthedocs.io

5.5 Performance Evaluation

We first introduce the experiment background and then show the evaluation results. We assess the effectiveness of the *IAM* distributed approach using a dataset.

Experiment Setup and Dataset

The experiment supporting our evaluation is focused on urban noise monitoring using crowdsensing. The dataset contains the measurements collected by the Ambiciti application in an opportunistic way [82]. Recall that a partner company Ambiciti provided us the dataset used for evaluation.

> *Ambiciti DATASET* specifically relates to the contributions gathered in Paris over year 2017 and includes about 950k entries from 550 crowdsensors. Each entry is a tuple of the form <*latitude index, longitude index, timestamp index, observation value*> with the *observation value* being the averaged sound level expressed in dB(A).

Note that a sound level in dB(A) is a logarithmic quantity, and hence sound levels cannot be simply averaged for aggregation. Instead, the sound levels in dB(A) are first converted into their energy equivalents. Then the energy equivalents are averaged algebraically, and finally, the resulted energy equivalent is converted back to its dB(A) value.

In the above context, *IAM* manages tensors that deal with the monitoring of the noise level over the whole area $\mathcal{A}$ of the city during $\mathcal{D} = 24$ hours. We decompose the city area into a 100×100 grid, and the sensing data that are collected during 1 hour are stored in the dedicated cell. Over 1 day, this results in a $100 \times 100 \times 24$ tensor, which contains at most $240k$ of data entries.

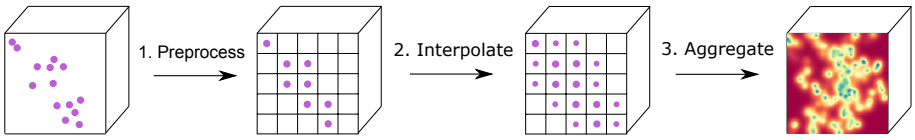


Figure 5.5.1: Producing a city noise map using *IAM* (at a zoomed scale)

Figure 5.5.1 illustrates the data analysis procedure for the computation of the urban noise map, using the *IAM* prototype. The figure zooms on the $5 \times 5 \times 1$ snapshot for which the crowdsensors provide sensor measurements. The figure highlights that the dataset is very sparse, which is the case with most crowdsensing applications and our reference application in particular.

In the following evaluation, the experiments are run either on a DELL Precision 7520 workstation, used both as the centralized server (Section 5.5.1) and for simulation (Section 5.5.2), or on an Android 6.0+ smartphone as end device test-bed (Section 5.5.2).

Accuracy Metrics

We consider the MAPE (Mean Absolute Percentage Error) and RMSE (Root Mean Square Error) to evaluate the accuracy of the interpolated and aggregated tensors. Given the observation data tensor $\mathcal{Y}$, the approximation data tensor $\hat{\mathcal{Y}}$ and the ground truth mask tensor $\mathcal{M}$ (indexing useful values), MAPE is defined as:

$$MAPE(\mathcal{Y}, \hat{\mathcal{Y}}, \mathcal{M}) := \frac{100\%}{\|\mathcal{M}\|^2} \sum \frac{\mathcal{M} * |\mathcal{Y} - \hat{\mathcal{Y}}|}{\mathcal{M} * \mathcal{Y}}$$

While RMSE is defined as:

$$RMSE(\mathcal{Y}, \hat{\mathcal{Y}}, \mathcal{M}) := \sqrt{\frac{\|\mathcal{M} * (\mathcal{Y} - \hat{\mathcal{Y}})\|^2}{\|\mathcal{M}\|^2}}$$

For cross-validation, we run 100 rounds of training followed by tests. At each round, both the training and evaluation sets are randomly shuffled, that is, for interpolation, 70% of the dataset is used for training (i.e., as actual observations to complete the tensor) and 30% to test (i.e., to assess the estimated values against the ground truth). Regarding aggregation, the entire approximation tensor is used for the evaluation.

5.5.1 Evaluation of the Interpolation Methods

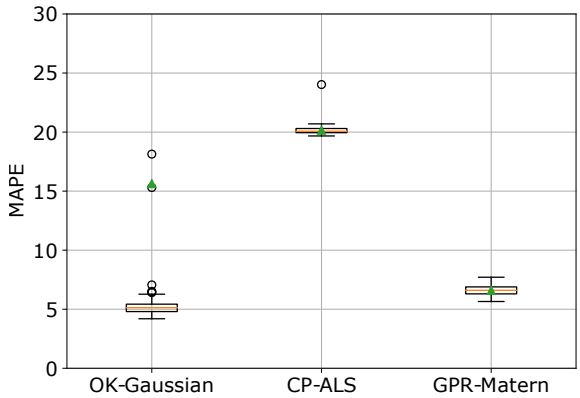


Figure 5.5.2: Interpolation accuracy comparison - MAPE

Figures 5.5.2 and 5.5.3 compare the robustness of the three inference strategies that are commonly used to interpolate physical phenomena (see Section 5.2.2): Ordinary kriging with Gaussian variogram model (*OK-Gaussian*), CP decomposition with Alternating Least Square (*CP-ALS*), and Gaussian Process Regression with Matern kernel (*GPR-Matern*). The interpolation is performed at the server (without involving any aggregation), using the dataset from which we selected the day during which the enormous amount of crowdsensing data was collected. The same

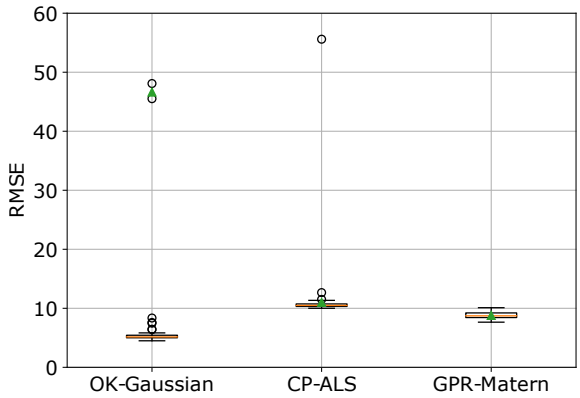


Figure 5.5.3: Interpolation accuracy comparison - RMSE

experiments were run using the whole dataset, and the same trends were observed. On the figures, the box corresponds to the interquartile range, the orange line is the median, and the green triangle is the mean. At first sight, OK-Gaussian seems to be accurate and hence promising, given the low MAPE and low RMSE interquartile range and median. However, some wrong inferences lead to abnormal values, as illustrated by high MAPE and RMSE mean values of 16% and 47, respectively. Similarly, but to a lower extent, CP-ALS shows some abnormal RMSE. Instead, GPR-Matern provides both an accurate and robust inference: stable MAPE and RMSE without outliers –hence characterized by the lowest variance– is observed.

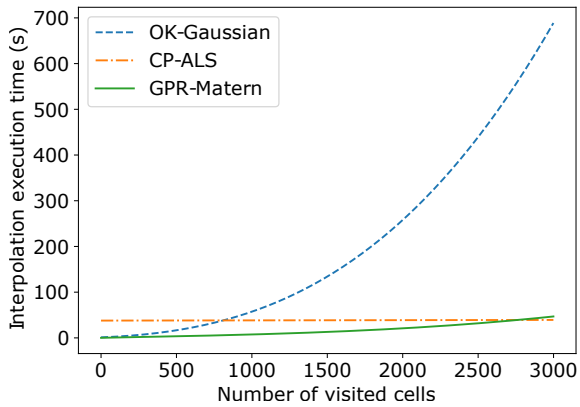


Figure 5.5.4: Comparison of interpolation execution time

Figure 5.5.4 shows the execution time of the three interpolation approaches. We run the experiments over the 365 days of our dataset, where the number of *observed cells* varies every day. The execution time of CP-ALS ($R = 1$) is constant in $\mathcal{O}(RIJK)$ regardless of the number of available *observed cells* since the computation applies on the entire fixed-size tensor. Both OK-Gaussian and GPR-Matern have a time complexity in $\mathcal{O}(n^3)$ with n being the number of *observed cells* used to fit the model. The execution time of GPR-Matern is lower than OK-Gaussian, and below CP-ALS when the number of *observed cells* is less than 2800. Note that in

our dataset, the number of *observed cells* collected by crowdsensors per day remains lower than 1500. Overall, GPR-Matern is the most efficient in terms of accuracy and robustness, while its execution time is also relatively lower.

We further evaluated the efficiency of the three interpolation methods in terms of memory consumption. GPR-Matern consumes the least memory: around $3.114MB$, with a variance of 1.718. Meanwhile, the memory consumption associated with CP-ALS (respectively OK-Gaussian) is of $3.258MB$ with a variance of 0.422 (respectively $4.644MB$ with a variance of 1.437). Note that the memory consumption is stable and does not depend on the number of *observed cells* since our approach always uses a fixed-size tensor that is filled with zeros in the absence of observations.

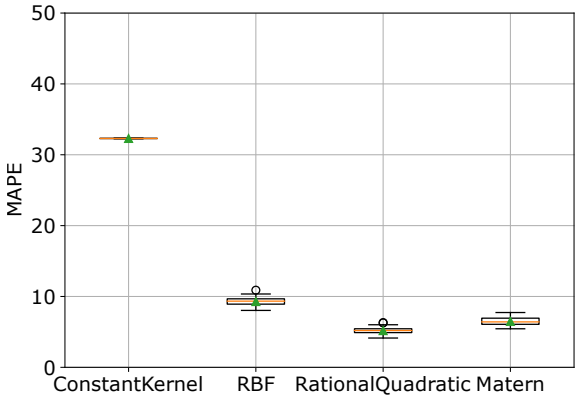


Figure 5.5.5: Kernel accuracy comparison - MAPE

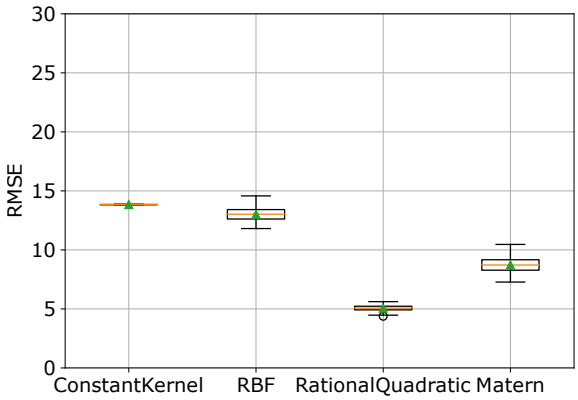


Figure 5.5.6: Kernel accuracy comparison - RMSE

Focusing on GPR, we assessed the accuracy (Figures 5.5.5 and 5.5.6) and associated execution time (Figure 5.5.7) of the following kernels: constant, RBF (Radial Basis Function), rational quadratic, and Matérn. The rational quadratic and Matérn kernels are the most accurate, while the former slightly outperforms the latter. However, the execution time of the quadratic kernel is twice as much as that of the Matérn kernel. We, therefore, leverage GPR with Matern kernel within *IAM*.

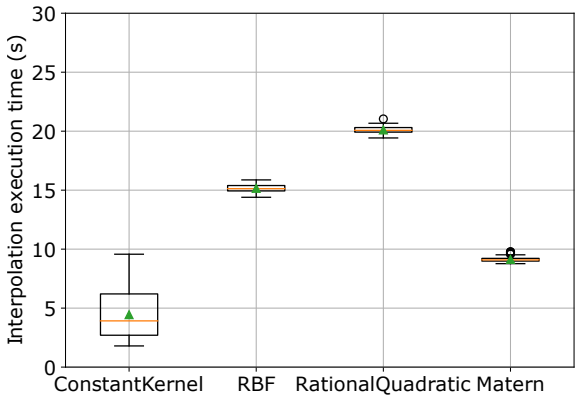


Figure 5.5.7: Comparison of Kernel execution time

5.5.2 Evaluation of the Distributed Aggregation

We perform both simulations based on the dataset and test-bed experiments.

Simulation-based Evaluation of the Aggregation

We compare the overall performance of the distributed vs centralized interpolation-mediated aggregation using our 1-year dataset. In the **centralized aggregation** case, the server collects all the sensing data and performs the aggregation and interpolation based on the whole dataset. As for the distributed aggregation (see Figure 5.5.1), we consider the following methods:

- **Ideal iterative aggregation** is a theoretical and sequential case in which the aggregation starts at the first crowdsensor that aggregates its tensor with the next crowdsensor and the aggregation process repeats with the following crowdsensors until the last crowdsensor is reached. This is the ideal case for which we ignore the actual locations of the crowdsensors.

- **Base stochastic aggregation** represents the real-life scenario: an aggregation occurs when at least two crowdsensors meet, as detected using the actual location and time proximity available from the dataset. The aggregation process thus depends on the mobility of the contributing users. Upon a meeting, the *mainstay* is selected randomly and $\beta = \beta' = 1$ for the generalized Product-of-Expert (see Algorithm 5). Ultimately, all the tensors are uploaded to and merged at the server, either directly or via a relay depending on the crowdsensors' P2P meetings.

- ***IAM* opportunistic aggregation** is similar to the above stochastic aggregation with the exception of the selection of the *mainstay* and the chosen β values. It follows our Algorithms 4 & 5, and we set $\beta_s = 1.5$ (resp. $\beta_{s'} = 0.5$) for crowdsensor s with lower D (resp. s' with higher D).

The above three methods are also compared with the **Centralized aggregation** where the server (DELL workstation in our experiment) collects the sensing data and performs the interpolation based on the whole dataset.

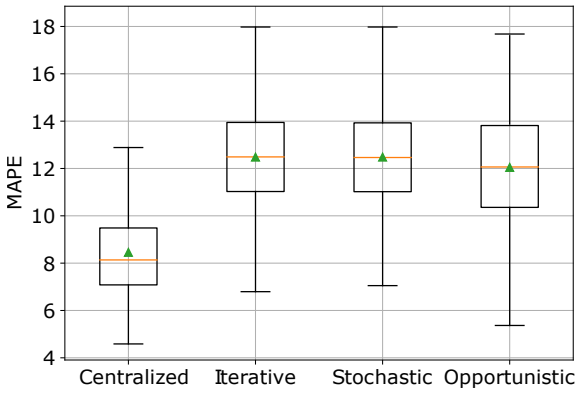


Figure 5.5.8: Aggregation accuracy comparison - MAPE

Figure 5.5.8 provides the MAPE of the above distributed and centralized aggregation approaches. As expected, the centralized approach is providing the most accurate inference as a baseline. The MAPE mean equals to 8.5%, and the MAPE median is 8.1%. The accuracy of distributed aggregations is quite similar: the MAPE is around 12.5%, with a median of 12.5%. In particular, our opportunistic aggregation is comparable to the centralized approach (e.g., the MAPE value is only 4 points higher than the centralized approach). It has MAPE mean and median of 12.0%. Overall, our approach is characterized by a slight decrease in accuracy compared to centralized aggregation.

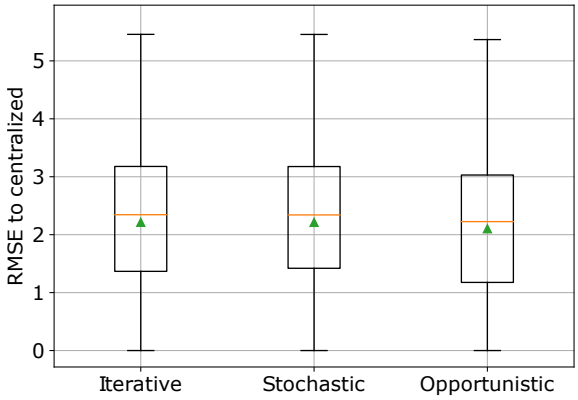


Figure 5.5.9: Aggregation accuracy comparison - RMSE

Figure 5.5.9 provides the RMSE of the three distributed aggregations compared to the centralized approach that serves as reference. Although the ideal iterative aggregation avoids the processing of interpolation and aggregation on the cloud, the RMSE mean equals 2.213 and the RMSE median equals 2.345. The accuracy of the

stochastic aggregation is quite similar, with a RMSE mean of 2.212, and a RMSE median of 2.341. Our opportunistic aggregation performs better than the other distributed approaches with a RMSE mean of 2.102 and a RMSE median of 2.225. Still, as expected, the decentralization impacts on the overall aggregation result, which is to be compared to the resulting resource gains. It is part of our future work to investigate further enhancement of the distributed interpolation-mediated aggregation by, e.g., accounting for the significance of the measurements gathered at a node when interpolating.

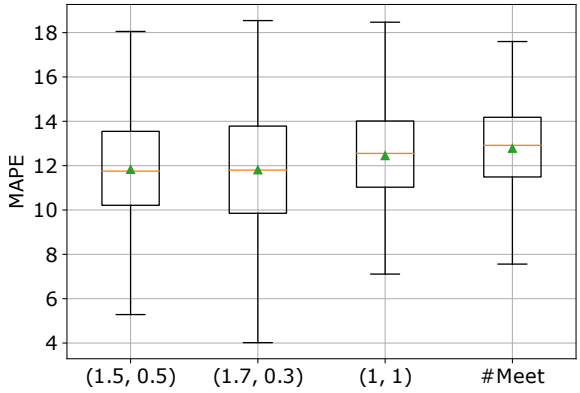


Figure 5.5.10: Impact of β setting on aggregation - MAPE

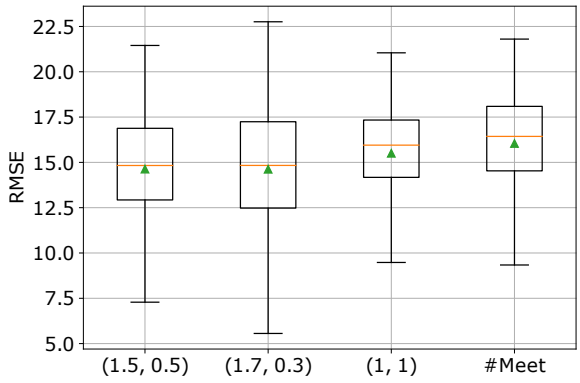


Figure 5.5.11: Impact of β setting on aggregation - RMSE

Figures 5.5.10 and 5.5.11 compare different weighting configurations $(\beta_s, \beta_{s'})$ for the two crowdsensors s and s' that aggregate. For three of the pairs, we manually set their values depending on the loss function D, i.e., the lower loss has a higher weight. For the fourth one, we set $\beta_s = 2\frac{n_s}{n_s+n_{s'}}$ using the ratio depending on the number of aggregations n that each crowdsensor previously has performed. The results show that assigning a higher weight to the crowdsensor with the lower loss function increases slightly the overall accuracy aggregated ultimately at the server.

Figure 5.5.12 shows the execution time associated with the entire procedure, including data pre-processing, interpolation, and aggregation, with the simulation

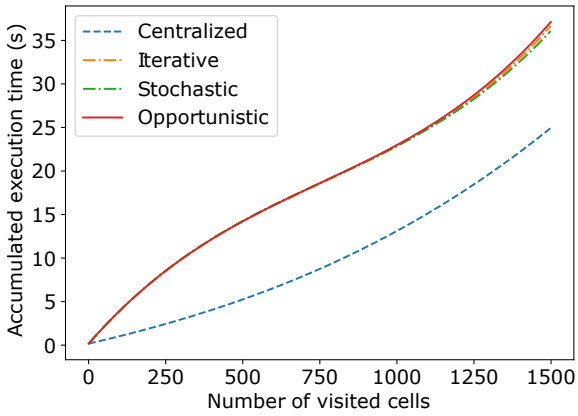


Figure 5.5.12: Comparison of Accumulated execution time

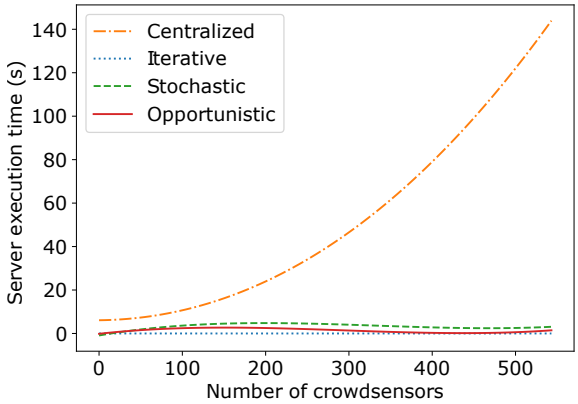


Figure 5.5.13: Comparison of server-only execution time

being performed at the server. By design, the centralized aggregation execution time is lower than the accumulated execution time of the distributed aggregation schemes. However, as illustrated in Figure 5.5.13, the execution time associated with the centralized aggregation (and interpolation) at the server significantly increases when the number of crowdsensors gets high. Instead, when the interpolation and aggregation are mainly performed by crowdsensors, the server execution time is almost negligible regardless of the number of crowdsensors. In addition, the storage requirement is minimized on the server because the data tensor size is always unchanged when aggregating new incoming data (via linear operations).

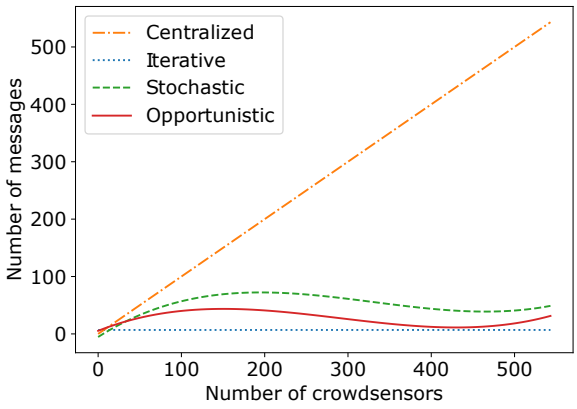


Figure 5.5.14: Comparison of directly uploaded messages

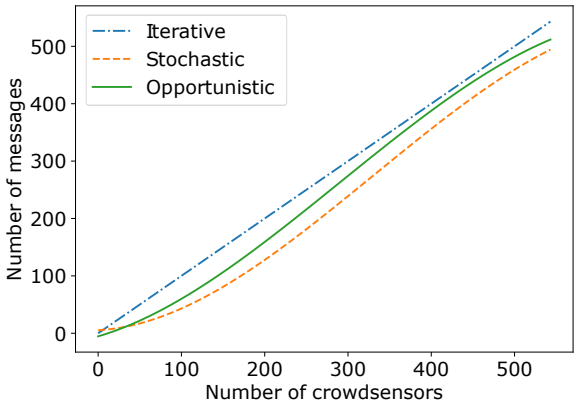


Figure 5.5.15: Comparison of P2P forwarded messages

Figures 5.5.14 and 5.5.15 compare the amount of traffic uploaded to the server and relayed among the crowdsensors respectively, in the centralized vs distributed cases. The traffic is evaluated based on the number of actual P2P aggregations (within relays). As expected, the distributed aggregation reduces the amount of traffic uploaded to the server, and hence the cellular network occupancy is kept to a minimum. Notably, the *IAM* opportunistic aggregation drastically reduces the uploading to the server by 54.2% compared to the stochastic aggregation. A portion

of the traffic sent to the server is replaced by the D2D forwarding among crowdsensors; there are more aggregations and thus more P2P traffics generated when the number of crowdsensors increases. This result supports our mainstay selection: crowdsensors with better inference quality tend to have more relays/aggregations.

Testbed-based Evaluation of the Aggregation

We now focus on the resource consumption of *IAM* on the end device, for which we analyze the execution time (depending on the number of *observed cells* and of aggregations) and power consumption (depending on the execution time and D2D protocol). We empirically assess the performance associated with the *IAM* prototype in terms of execution time and energy consumption, using Android smartphones. We conduct the experiment using our one-year dataset as data input.

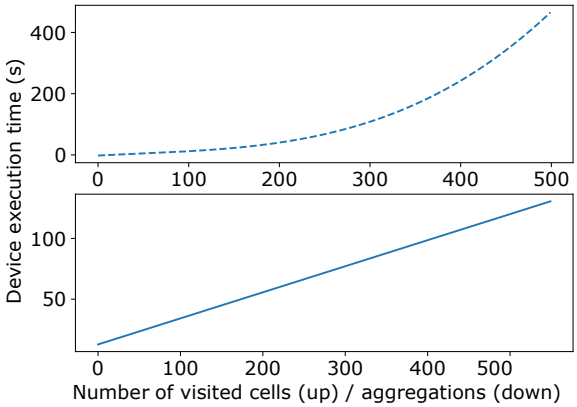


Figure 5.5.16: On-device execution time of computing

Then, we run experiments on a SAMSUNG GALAXY S7 smartphone embedding a 3000 mAh battery capacity. Figure 5.5.16 shows the interpolation execution time depending on the number of cells (i.e., area at the target scale) covered by the crowdsensor, and the aggregation execution time depending on the number of aggregations. Note that the figure shows no more than 500 entries, which is in practice a very high number of cells contributed/visited by one crowdsensor per day. As expected, the interpolation is computationally intensive compared to the aggregation, whose execution time is comparatively negligible: the interpolation takes a couple of minutes when the number of *observed cells* is greater than 500, while the aggregation takes less than 100 seconds for a number of aggregations below 500 and for a number of *observed cells* per crowdsensor varying from 1 to 500. The aggregation shows a linear time complexity.

Recall that each crowdsensor executes the interpolation only once, which is the most computation-intensive operation. Assuming the crowdsensor has 500 *observed cells*, an interpolation consumes the most energy with 88mAh, and an aggregation consumes only 2mAh. Figure 5.5.17 estimates the energy consumed by a smartphone that implements the interpolation and opportunistic aggregation when the

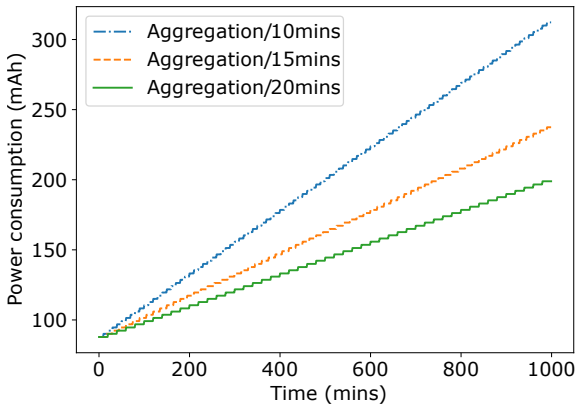


Figure 5.5.17: On-device energy consumed by computing

P2P meeting frequency varies: the more frequent is the aggregation, the more energy is consumed. Nevertheless, the related energy consumption remains under control because in practice the crowdsensor usually has already relayed/aggregated its data before encountering around 8 crowdsensors for a single day. In summary, the interpolation of 500 *observed cells* and 8 aggregations for a day cost only 2.88% of the battery capacity. With respect to communications, based on the power assessment in [94], uploading via cellular network consumes 8.9 (resp. 4) times of the D2D relay energy via Bluetooth (resp. Wi-Fi).

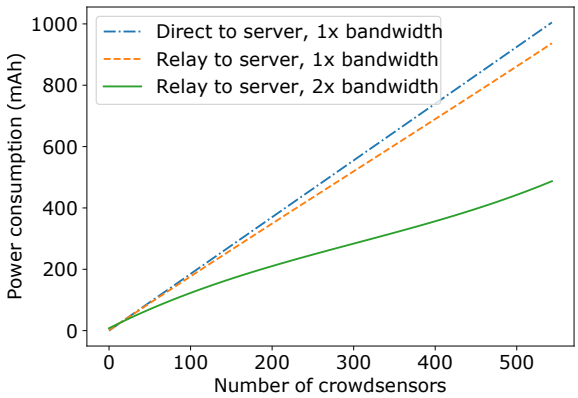


Figure 5.5.18: Energy consumed by network communication

To evaluate the power consumption due to communication, we rely on the power profile of an LG NEXUS 5X smartphone provided by the manufacturer. Assuming the bandwidth is constant, the Wi-Fi transmission consumes 1.72mAh while the cellular transmission consumes 1.85mAh power for a packet. We estimate the energy consumption associated with all the crowdsensors. Figure 5.5.18 shows the energy consumption associated with the local P2P traffic as well as cellular Internet traffic. The energy is reduced by replacing uploading with P2P relays. When the P2P bandwidth is higher than the cellular bandwidth (e.g., two times in figure), the energy-saving is more significant. Furthermore, the local P2P traffic is not only less

costly in terms of energy aspects but also in terms of budget.

5.5.3 Impact on the Financial Cost

We here assess the benefit of the decentralized *IAM* approach to crowdsensing from a financial perspective. We specifically compare the *IAM* solution with the more classical centralized one, which often relies on a cloud platform for data analysis and storage. This is in particular the configuration of the crowdsensing system of the Ambiciti company that provided us the dataset: the system initially used the Google Cloud Platform (GCP, https://cloud.google.com/products/), which we consider as an illustrative candidate for estimating the budget associated with a cloud-based configuration.

GCP provides the following vital services: *Cloud IoT Core* is responsible for connecting the cloud to the IoT devices and establishing two-way communication. Upon the reception of a packet, the *Cloud Pub/Sub* service creates and delivers an event notification to *Cloud Functions* that implements basic operations (e.g., average, maximum, minimum) needed to pre-process the data. *BigQuery* is used to store the pre-processed data temporarily during aggregation and interpolation. *Compute Engine* runs a virtual machine that performs the advanced computation (e.g., interpolation and aggregation). *Cloud Bigtable* is a NoSQL database that stores the resulting aggregated and interpolated sensing data. The usage of each of the above services is priced. For a detailed description, one may refer to the list[6] online. In short, the price depends on the amount of network traffic received/sent, the amount of storage needed, and the load associated with the computation (e.g., number of pre-processing functions invoked, and execution time associated with the interpolation and aggregation).

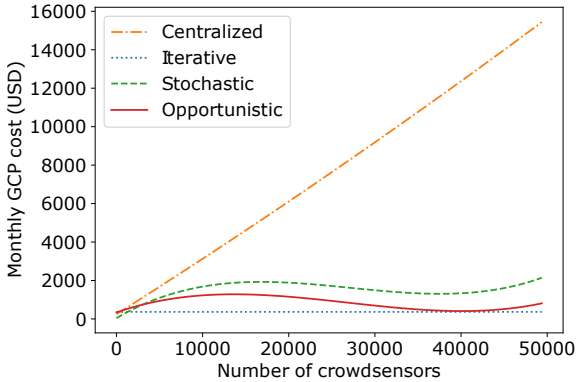


Figure 5.5.19: Monthly financial cost of a cloud-based deployment

Figure 5.5.19 estimates the monthly financial cost associated with running centralized vs distributed crowdsensing while using the GCP platform as the server and assuming that each crowdsensor sends a $2MB$ packet every day. The cost

[6] https://cloud.google.com/pricing/list

associated with the centralized approach increases linearly because the number of uploads and the computation involved to interpolate the phenomena are both high. Instead, the costs of the stochastic and opportunistic approaches remain low because communication toward the cloud is reduced and only lightweight aggregation is performed on the cloud. Our *IAM* opportunistic aggregation outperforms the stochastic approach because we further reduce the computing load on the cloud. In general, the *IAM* decentralized solution significantly alleviates the dependence on the server infrastructure, thereby enabling the wider adoption of crowdsensing systems by communities of users concerned with environmental monitoring.

5.6 Discussion

To the best of our knowledge, our work is the first to investigate and design a fully distributed interpolation and aggregation based on the opportunistic encounters of crowdsensors. Our key hypothesis is that the number of encounters –both past and future– is correlated to the number of observations and thus the inference quality. Other common criteria such as device resource/status may be further considered when selecting the mainstay. Still, our evaluation does not analyze the power consumption due to D2D communication, as we rely on the results of previous studies that show that D2D networking is cheaper than cellular networking [195, 44, 34]. It is part of our future work to further investigate the overall effectiveness of *IAM* in terms of resource efficiency vs data accuracy, compared to the centralized approach. In addition to the required energy efficiency and accuracy for any crowdsensing system, ensuring privacy is another key concerns for the end-users. Here, we claim that the fully decentralized approach of *IAM* outperforms the centralized approach in terms of privacy, further considering the opportunistic, impromptu encounters of the crowdsensors, which subsequently share aggregated & interpolated tensors about environmental phenomena. The opportunistic approach then raises the potential issue of un-trustworthy crowdsensors that may contribute malicious data. While the *IAM* solution presented in the paper assumes trustworthy crowdsensors providing equally accurate measurements, our aim in the near future is to investigate mechanisms that filter anomalous data (e.g., outliers) to deal with untrustworthy contributors.

We argue the collaborative crowdsensing at the edge must handle different interaction behaviors of users. While group-wise collaboration promoted by *BeTogether* improves the data collection, pair-wise collaboration in *IAM* enhances the data processing. In practice, the people's mobility makes the crowdsensing contribution unevenly distributed over space and time, which requires the analysis of the contributed observations that is in general performed at a central, often cloud-based, server. However, as the number of contributors grows, the increasing number of observations that the crowdsensing systems must process gets challenging: the high network and financial cost associated to a cloud-centric system hinders the widespread deployment of crowdsensing, and the high computational cost due

to the large amount of data makes intractable the modeling of the environmental phenomenon. Furthermore, the increasing –and relevant– concerns of the citizens about the privacy invasion of a centralized platform is challenging the participation of crowdsensing campaigns.

We tackle the above issues by exploiting the increasing computing capacity of today's smartphones, that is, we distribute the interpolation and aggregation associated with the sensing data at the powerful end devices. To do so, we introduce *IAM* that runs on the smartphone to capture complex relationships among the collected observations across both space and time by relying on Gaussian Process Regression and 3D tensors. Then, the resulting tensors are opportunistically combined together following a stochastic process based on the physical encounters of people. The benefit of our approach is threefold: (i) each crowdsensor (i.e., expert) independently establishes an interpolation of the region it covered; (ii) the aggregation resulting from the Product-of-Experts is sharper than any of the individual tensor and renders much more tractable the establishment of the overall tensor; and (iii) the computation achieved on the device is limited, and thus not energy-exhausting. Indeed, the evaluation using a real-world dataset shows that our approach significantly reduces the transmission to, and the computing resource consumed on, the infrastructure server, compared to the centralized approach.

Chapter 6

Conclusion

Contents

6.1 Summary of the Thesis

The data gluttony of AI is well known: data fuels artificial intelligence. Technologies that help to gather the necessary data are therefore essential, among which IoT is contributory. However, the deployment of IoT solutions raises significant challenges, especially with respect to resource and budget costs. It is our view that mobile crowdsensing, *aka* mobile phone sensing, has a significant role to play because it can potentially contribute massive amounts of data at a relatively low cost.

Notably, we believe opportunistic crowdsensing is the best way to support ubiquitous sensing and pervasive computing. It empowers citizens to sense objective phenomena, running autonomously, at a large and fine-grained scale. Nevertheless, crowdsensing would be unproductive, and even harmful, if the contributed data are not adequately processed. Especially opportunistic crowdsensing shifts the burden from users to the application and platform to extract valuable information. In short, the challenge lies in achieving efficiency, that is, simultaneously enhancing the quality of the data and reducing the cost of sensing.

People may provide valuable observations across time and space using their smart devices, e.g., smartphones, which are characterized by sensing capabilities along with powerful computing capabilities and are usually equipped with multiple short-range D2D network interfaces, e.g., Wi-Fi, Bluetooth. Furthermore, smart devices do not only embed physical sensors along with networking and computing components;

they also hold social properties associated with the people that e.g., encounter each others. Considering these characteristics, this thesis promotes collaborative crowdsensing at the end-device level, and, aims at making opportunistic crowdsensing a reliable means of urban environmental monitoring, for which we advocate enforcing the cost-effective collection of high-quality data. Our research introduces three contributions (presented in Chapters 3, 4, and 5) that implement the *collaborative crowdsensing at the edge* by supporting the three complementary functionalities:

1. **Context-awareness**: We start from the premise that the quality of the provided measurements depends on the adequacy of the sensing context with respect to the analyzed phenomenon. Chapter 3 concentrates more specifically on assessing the sensing context beyond its geographical position in the Euclidean space, when gathering observations about the physical environment, i.e., whether the smartphone is in-/out-pocket, in-/out-door and upper-/under-ground. We introduced *ContextSense* to leverage online learning for the local inference of the sensing context, in order to overcome the disparity of the classification performance due to the heterogeneity of the sensing devices as well as the diversity of user behavior and novel usage scenarios. *ContextSense* specifically features a hierarchical algorithm for the inference that requires few opportunistic feedback from the user. Evaluation results considering different users, show that *ContextSense* increases the accuracy of the context inference per user while generating a low resource consumption.

2. **Group-wise collaboration:** In crowded/dense areas wherein many crowdsensors remain close together and upload data, the uncontrolled collection of massive amounts of raw sensing data incurs significant resource consumption for both the end device and the server, and also negatively affects the quality of the aggregated observations. Chapter 4 tackles both challenges raised by opportunistic crowdsensing, enabling the resource-efficient gathering of high-quality observations. To achieve this, we introduce *BeTogether* to foster context-aware and collaborative groups constituted of co-located crowdsensors that operate in the same context and share the workload in a cost- and quality-effective way. We evaluate *BeTogether* using an implementation-driven evaluation that leverages the Ambiciti dataset, containing nearly one million entries contributed by 550 crowdsensors over one year. Compared to the cloud-centric approach, *BeTogether* increases the quality of the collected data while reducing the overall resource cost.

3. **Pair-wise collaboration:** As for uncrowded/sparse areas wherein crowdsensors encounter briefly and data is cached, crowdsensing at scale involves significant communication, computation, and financial costs due to the dependence on cloud/edge infrastructure for the processing of the spatio-temporal data. Although sorely needed to inform our knowledge of the environment, this steep cost limits the adoption of crowdsensing by activists. As an alternative to the centralized analysis of crowdsensed observations, Chapter 5

introduces *IAM*: a distributed collaboration for data analysis running on the smartphones. To do so efficiently, we model the interpolation as a tensor completion problem. We introduce an opportunistic lightweight aggregation method that anticipates future encounters according to the quality of the contributed data. Thus, depending on the pair-wise encounters of crowdsensors, *IAM* may shift from centralized processing to a distributed, on the move processing of crowdsensed data. The evaluation of *IAM* using part of quantitative environmental measurements contained in the Ambiciti dataset, shows that it significantly reduces –and may even remove– the dependence on the centralized infrastructure. At the same time, it incurs a limited resource cost on the crowdsensors, while the overall data accuracy remains comparable to that of the centralized approach.

In summary, *ContextSense*, *BeTogether*, and *IAM* jointly construct our solution of *collaborative crowdsensing at the edge*. They enhance the efficiency of opportunistic crowdsensing, on both the end device level and the cloud level. The inference accuracy and sensing data quality are improved on the crowdsensor and the cloud, respectively. At the same time, the network and computing resource consumption on the crowdsensor and the cloud are reduced.In other words, we propose a set of collaborative crowdsensing mechanisms among end devices that reduce the costs for both the end device and the cloud/server, while increasing the overall data quality.

6.2 Perspective and Future Work

Mobile crowdsensing is a feasible method for gathering massive amounts of data from ubiquitous smartphone users. Not only is this system capable of gathering data, but it can also carry out data analysis on the devices. For instance, crowdsensing has made significant contributions to the monitoring of the epidemic. People threatened by the COVID-19 virus can be tracked using their smartphone application; their movement is analyzed to alert them and prevent further spreading of the virus. Nonetheless, crowdsensing has its share of issues. The application crowdsensing analyzes the data available on the device, which have been generated by the physical sensors. We could possibly utilize more data related to e.g. application usage, user profile, and user involvement in social networks. However, the usage of such data may cause privacy concerns, therefore requiring full prior authorization by the users. This may in turn limit the system's wider application. Thus, the question returns to the issue of incentive: why should a user authorize opportunistic crowdsensing on their device? Regarding this issue, for now we have chosen to assume altruism as the primary motivation for participation. In this thesis we have decided to focus on addressing the issue of efficiency.

Following, we investigate the possible improvements of each contribution, and further explore future challenges.

Threat to validity: The main limitation of this thesis is related to the evaluation of *ContextSense*, *BeTogether* and *IAM*. Each of these evaluations is constrained by the datasets that were used during the assessment. Regarding the personalized context inference, a larger dataset containing information from a greater number of users would permit a better assessment of the robustness of our approach. Currently, we have selected three datasets containing only several days of information. In the Ambiciti dataset, users are distributed sparsely, and the contextual information is limited to user activity and in-/out-pocket status. For the context-aware collaborative grouping, a dataset containing more contextual information from a denser group of users, would enhance the evaluation of the proposed approach. A single dataset was used to validate the effectiveness of the interpolation and aggregation approaches. While interpolation alleviates the issue of sparsity, having more datasets holding other quantitative measurements may support a better validation process. Overall, our research aims at analyzing urban phenomena using information gathered from people. The construction of larger and richer datasets would require the involvement of many participants and the deployment of experiments at a massive scale. Unfortunately, unlike certain companies, we do not have access to massive amounts of personal data. Instead, we have tried our best to leverage the dataset in our collection, although it is not perfect. A public dataset following the FAIR (Findability, Accessibility, Interoperability, and Reusability) principles would be helpful.

Improvement: Our joint contributions can be described as a distributed *ad hoc* system combined with on-device machine learning. It is a cross-discipline solution built upon the *network*, *system* and *data science* fields. As such, it can be potentially extended in multiple aspects: Its capability to infer personalized contexts could be enhanced. We applied a online learning approach that requires user feedback. Active learning and transfer learning may be combined together to amplify the impact of the user's feedback. Consequently, the gradually more intelligent notifications will imply that less feedback are needed. Meanwhile, federated learning might be leveraged to deploy a pre-trained learning model on the same device models, which will speed up the on-device evolution of learning models. The pursued goal is to further reduce the burden on crowdsensing users and the financial cost associated with the gathering and processing of the data. A distributed crowdsensing system could also be improved. In particular, D2D communication among multiple mobile devices deserves an adapted protocol. Although the mobile ad hoc network has been studied during decades, the evolution of 5G introduces more ubiquitous networks: the D2D is always a trend with cellular network that should connect every thing [179, 9]. Our collaborative group is dynamic; as a result, its stability may be a concern. Without a properly centralized view, our solutions may not find the global optimal. Thus, the shift from a approximate and distributed tasks allocation to centralized optimization remains a problem to be solved leveraging 5G ubiquity.

Future work: We have partly investigated distributed machine learning. Mobile machine learning is the current trend, especially given the actual development of artificial intelligence of things. Our life will gradually be accompanied by, e.g., smartphones, smart homes, smart vehicles. Artificial intelligence, either in its classic form based on statistic/probability or in its emerging form based on neural networks, will be deployed on machines and other things. Distributed and ubiquitous learning would seem necessary in such a context. One variety of distributed machine learning is federated learning, which combines various models created by numerous experts. Although many end devices are now capable of running complex inferences (e.g., TensorFlow Lite), deep learning training remains too demanding for the mobile device. The question of how to effectively distribute computation load is still a point of interest. Edge devices designed specifically for neural networks have been developed recently. In the future, we are interested in context-aware smart sensing, which is in some ways an extension of crowdsensing and artificial intelligence of things. For example, the smart vehicle system requires such sensing to detect abnormalities and recommend services to the user. It is related to the user; it is a sensing service, and it requires intelligence.

Ultimately, distributed and collaborative crowdsensing demonstrates the potential and advantages of ubiquitous computing. We believe that the *ad hoc* collaboration among end devices via instant and short-range communication is an essential part of pervasive computing, especially from the machine learning standpoint.

Bibliography

[1] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer communications*, vol. 30, no. 14-15, 2007.

[2] Z. S. Abdallah, M. M. Gaber, B. Srinivasan *et al.*, "Streamar: incremental and active learning with evolving sensory data for activity recognition," in *IEEE International Conference on Tools with Artificial Intelligence*, vol. 1, 2012.

[3] A. B. Abkenar, S. W. Loke, W. Rahayu *et al.*, "Energy considerations for continuous group activity recognition using mobile devices: The case of groupsense," in *IEEE International Conference on Advanced Information Networking and Applications*, 2016.

[4] A. B. Abkenar, S. W. Loke, A. Zaslavsky *et al.*, "Garsaaas: Group activity recognition and situation analysis as a service," *Journal of Internet Services and Applications*, vol. 10, no. 1, 2019.

[5] E. Acar, D. M. Dunlavy, T. G. Kolda *et al.*, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, 2011.

[6] M. Ali, T. ElBatt, and M. Youssef, "Senseio: Realistic ubiquitous indoor outdoor detection system using smartphones," *Sensors Journal*, vol. 18, no. 9, 2018.

[7] W.-F. Alliance, "Wi-fi peer-to-peer services (p2ps) technical specification," Version 1.2, Tech. Rep., 2015.

[8] D. Amaxilatis, E. Lagoudianakis, G. Mylonas *et al.*, "Managing smartphone crowdsensing campaigns through the organicity smart city platform," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016.

[9] R. I. Ansari, C. Chrysostomou, S. A. Hassan *et al.*, "5g d2d networks: Techniques, challenges, and future prospects," *Systems Journal*, vol. 12, no. 4, 2017.

[10] A. Antonić, M. Marjanović, K. Pripužić *et al.*, "A mobile crowd sensing ecosystem enabled by cupus: Cloud-based publish/subscribe middleware for the internet of things," *Future Generation Computer Systems*, vol. 56, 2016.

[11] A. Antonic, K. Roankovic, M. Marjanovic *et al.*, "A mobile crowdsensing ecosystem enabled by a cloud-based publish/subscribe middleware," in *IEEE International Conference on Future Internet of Things and Cloud*. IEEE, 2014.

[12] A. Asadi and V. Mancuso, "Wifi direct and lte d2d in action," in *IEEE/IFIP Wireless Days Conference*, 2013.

[13] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, 2014.

[14] R. Azzam, R. Mizouni, H. Otrok, A. Ouali *et al.*, "Grs: A group-based recruitment system for mobile crowd sensing," *Journal of Network and Computer Applications*, vol. 72, 2016.

[15] J. Ballesteros, B. Carbunar, M. Rahman *et al.*, "Towards safe cities: A mobile and social networking approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, 2013.

[16] X. Bao and R. Roy Choudhury, "Movi: mobile phone based video highlights via collaborative sensing," in *ACM International Conference on Mobile Systems, Applications, and Services*, 2010.

[17] A. Baruch, A. May, and D. Yu, "The motivations, enablers and barriers for voluntary participation in an online crowdsourcing platform," *Computers in Human Behavior*, vol. 64, 2016.

[18] D. Belli, S. Chessa, L. Foschini *et al.*, "Enhancing mobile edge computing architecture with human-driven edge computing model," in *IEEE International Conference on Intelligent Environments*, 2018.

[19] D. Bonino, M. Alizo, C. Pastrone *et al.*, "Wasteapp: Smarter waste recycling for smart citizens," in *IEEE International Multidisciplinary Conference on Computer and Energy Science*, 2016.

[20] A. Bose and C. H. Foh, "A practical path loss model for indoor wifi positioning enhancement," in *IEEE International Conference on Information, Communications & Signal Processing*, 2007.

[21] U. Botrel Menegato, L. Souza Cimino, S. E. Delabrida Silva *et al.*, "Dynamic clustering in wifi direct technology," in *ACM International Symposium on Mobility Management and Wireless Access*, 2014.

[22] M. Budde, P. Barbera, R. El Masri *et al.*, "Retrofitting smartphones to be used as particulate matter dosimeters," in *ACM International Symposium on Wearable Computers*, 2013.

[23] N. Bulusu, C. T. Chou, S. Kanhere *et al.*, "Participatory sensing in commerce: Using mobile camera phones to track market price dispersion," in *International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems*, 2008.

[24] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of gaussian process predictions," in *Modern Nonparametrics 3: Automating the Learning Pipeline workshop at NIPS*, 2014.

[25] A. Capponi, C. Fiandrino, B. Kantarci *et al.*, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, 2019.

[26] A. Capponi, C. Fiandrino, D. Kliazovich *et al.*, "A cost-effective distributed framework for data collection in cloud-based mobile crowd sensing architectures," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 1, 2017.

[27] N. Capurso, B. Mei, T. Song *et al.*, "A survey on key fields of context awareness for mobile devices," *Journal of Network and Computer Applications*, vol. 118, 2018.

[28] I. Carreras, D. Miorandi, A. Tamilin *et al.*, "Matador: Mobile task detector for context-aware crowd-sensing campaigns," in *IEEE International Conference on Pervasive Computing and Communications Workshops*, 2013.

[29] A. Carroll, G. Heiser *et al.*, "An analysis of power consumption in a smartphone." in *USENIX Annual Technical Conference*, 2010.

[30] C. Caruso and F. Quarta, "Interpolation methods comparison," *Computers & Mathematics with Applications*, vol. 35, no. 12, 1998.

[31] D. Chatzopoulos, M. Ahmadi, S. Kosta *et al.*, "Openrp: A reputation middleware for opportunistic crowd computing," *IEEE Communications Magazine*, vol. 54, no. 7, 2016.

[32] K. Chen and G. Tan, "Satprobe: Low-energy and fast indoor/outdoor detection based on raw gps processing," in *IEEE International Conference on Computer Communications*, 2017.

[33] L. Chen, L. Wang, D. Zhang *et al.*, "Enup: Energy-efficient data uploading for mobile crowd sensing applications," in *IEEE International Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*. IEEE, 2016.

[34] X. Chen, L. Pu, L. Gao *et al.*, "Exploiting massive d2d collaboration for energy-efficient mobile edge computing," *Wireless Communications*, vol. 24, no. 4, 2017.

[35] Y.-C. Chen, E. Rosensweig, J. Kurose *et al.*, "Group detection in mobility traces," in *ACM International Wireless Communications and Mobile Computing Conference*, 2010.

[36] Y. Cheng, X. He, Z. Zhou *et al.*, "Maptransfer: Urban air quality map generation for downscaled sensor deployments," in *ACM/IEEE International Conference on Internet of Things Design and Implementation*, 2020.

[37] Y. Chon, N. D. Lane, Y. Kim *et al.*, "Understanding the coverage and scalability of place-centric crowdsensing," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2013.

[38] Y. Chon, N. D. Lane, F. Li *et al.*, "Automatically characterizing places with opportunistic crowdsensing using smartphones," in *ACM International Conference on Ubiquitous Computing*, 2012.

[39] S. Das, S. Chatterjee, S. Chakraborty *et al.*, "Groupsense: A lightweight framework for group identification," *IEEE Transactions on Mobile Computing*, vol. 18, no. 12, 2018.

[40] M. De Domenico, A. Lima, and M. Musolesi, "Interdependence and predictability of human mobility and social interactions," *Pervasive and Mobile Computing*, vol. 9, no. 6, 2013.

[41] P. O. V. de Melo, A. C. Viana, M. Fiore *et al.*, "Recast: Telling apart social and random relationships in dynamic networks," *Performance Evaluation*, vol. 87, 2015.

[42] M. P. Deisenroth and J. W. Ng, "Distributed gaussian processes," in *International Conference on Machine Learning*, 2015.

[43] J. Demšar, T. Curk, A. Erjavec *et al.*, "Orange: data mining toolbox in python," *Journal of Machine Learning Research*, vol. 14, no. 1, 2013.

[44] N. Do, Y. Zhao, C.-H. Hsu *et al.*, "Crowdsourced mobile data transfer with delay bound," *ACM Transactions on Internet Technology*, vol. 16, no. 4, 2016.

[45] T. M. Do and D. Gatica-Perez, "Human interaction discovery in smartphone proximity networks," *Personal and Ubiquitous Computing*, vol. 17, no. 3, 2013.

[46] Y. Du, "In-network collaborative mobile crowdsensing," in *IEEE International Conference on Pervasive Computing and Communications PhD Forum*, 2020.

[47] Y. Du, V. Issarny, and F. Sailhan, "User-centric context inference for mobile crowdsensing," in *ACM International Conference on Internet of Things Design and Implementation*, 2019.

[48] ——, "When the power of the crowd meets the intelligence of the middleware: The mobile phone sensing case," *ACM SIGOPS Operating Systems Review*, vol. 53, no. 1, 2019.

[49] Y. Du, F. Sailhan, and V. Issarny, "Let opportunistic crowdsensors work together for resource-efficient, quality-aware observations," in *IEEE International Conference on Pervasive Computing and Communications*, 2020.

[50] J. Dutta, P. Pramanick, and S. Roy, "Noisesense: Crowdsourced context aware sensing for real time noise pollution monitoring of the city," in *IEEE International Conference on Advanced Networks and Telecommunications Systems*, 2017.

[51] P. Dutta, P. M. Aoki, N. Kumar *et al.*, "Common sense: participatory urban sensing using a network of handheld air quality monitors," in *ACM Conference on Embedded Networked Sensor Systems*, 2009.

[52] K. Eldrandaly and A. Abdelmouty, "Spatio-temporal interpolation: Current practices and future prospects," *International Journal of Digital Content Technology and its Applications*, vol. 11, 06 2017.

[53] M. Elhoushi, J. Georgy, A. Noureldin *et al.*, "A survey on approaches of motion mode recognition using sensors," *IEEE Transactions on intelligent transportation systems*, vol. 18, no. 7, 2016.

[54] M. Ester, H.-P. Kriegel, J. Sander *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *ACM Conference on Knowledge Discovery and Data Mining*, 1996.

[55] K. Farrahi and D. Gatica-Perez, "Probabilistic mining of socio-geographic routines from mobile phone data," *Journal of Selected Topics in Signal Processing*, vol. 4, no. 4, 2010.

[56] ——, "Discovering routines from large-scale human locations using probabilistic topic models," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 1, 2011.

[57] A. Farshad, M. K. Marina, and F. Garcia, "Urban wifi characterization via mobile crowdsensing," in *IEEE Network Operations and Management Symposium*, 2014.

[58] M. Faulkner, R. Clayton, T. Heaton *et al.*, "Community sense and response systems: Your phone as quake detector," *Communications of the ACM*, vol. 57, no. 7, 2014.

[59] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, 2011.

[60] H. Gao, C. H. Liu, J. Tang *et al.*, "Online quality-aware incentive mechanism for mobile crowd sensing with extra bonus," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, 2018.

[61] M. Girolami, S. Chessa, G. Adami *et al.*, "Sensing interpolation strategies for a mobile crowdsensing platform," in *IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, 2017.

[62] M. Girolami, S. Chessa, M. Dragone *et al.*, "Using spatial interpolation in the design of a coverage metric for mobile crowdsensing systems," in *IEEE International Symposium on Computers and Communication*, 2016.

[63] L. Grasedyck, D. Kressner, and C. Tobler, "A literature survey of low-rank tensor approximation techniques," *GAMM-Mitteilungen*, vol. 36, no. 1, 2013.

[64] S. Grubeša, A. Petošić, M. Suhanek *et al.*, "Mobile crowdsensing accuracy for noise mapping in smart cities," *Automatika*, vol. 59, no. 3-4, 2018.

[65] J. Gubbi, R. Buyya, S. Marusic *et al.*, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, 2013.

[66] R. E. Guinness, "Beyond where to how: A machine learning approach for sensing mobility contexts using smartphone sensors," *Sensors*, vol. 15, no. 5, 2015.

[67] B. Guo, Z. Wang, Z. Yu *et al.*, "Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm," *ACM Computing Surveys*, vol. 48, no. 1, 2015.

[68] B. Guo, Z. Yu, X. Zhou *et al.*, "From participatory sensing to mobile crowd sensing," in *IEEE International Conference on Pervasive Computing and Communication Workshops*, 2014.

[69] S. Hachem, V. Mallet, R. Ventura *et al.*, "Monitoring noise pollution using the urban civics middleware," in *IEEE International Conference on Big Data Computing Service and Applications*, 2015.

[70] S. Hachem, A. Pathak, and V. Issarny, "Probabilistic registration for large-scale mobile participatory sensing," in *IEEE International Conference on Pervasive Computing and Communications*, 2013.

[71] ——, "Service-oriented middleware for large-scale mobile participatory sensing," *Pervasive and Mobile Computing*, vol. 10, 2014.

[72] K. Hänsel, H. Haddadi, and A. Alomainy, "Awsense: A framework for collecting sensing data from the apple watch," in *ACM International Conference on Mobile Systems, Applications, and Services*, 2017.

[73] P. Harris, A. Fotheringham, R. Crespo *et al.*, "The use of geographically weighted regression for spatial prediction: an evaluation of models using simulated data sets," *Mathematical Geosciences*, vol. 42, no. 6, 2010.

[74] D. Hasenfratz, O. Saukh, S. Sturzenegger *et al.*, "Participatory air pollution monitoring using smartphones," in *ACM International Workshop on Mobile Sensing*, 2012.

[75] A. Hassani, P. D. Haghighi, and P. P. Jayaraman, "Context-aware recruitment scheme for opportunistic mobile crowdsensing," in *IEEE International Conference on Parallel and Distributed Systems*, 2015.

[76] J. Hicks, N. Ramanathan, D. Kim *et al.*, "Andwellness: an open mobile system for activity and experience sampling," in *ACM Wireless Health*, 2010.

[77] T. Higuchi, H. Yamaguchi, and T. Higashino, "Context-supported local crowd mapping via collaborative sensing with mobile phones," *Pervasive and Mobile Computing*, vol. 13, 2014.

[78] T. Higuchi, H. Yamaguchi, T. Higashino *et al.*, "A neighbor collaboration mechanism for mobile crowd sensing in opportunistic networks," in *IEEE International Conference on Communications*, 2014.

[79] R. V. Hogg, E. A. Tanis, and D. L. Zimmerman, *Probability and statistical inference*. Pearson/Prentice Hall, 2010.

[80] S. Hyuga, M. Ito, M. Iwai *et al.*, "Estimate a user's location using smartphone's barometer on a subway," in *ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, 2015.

[81] A. Ignatov, R. Timofte, W. Chou *et al.*, "Ai benchmark: Running deep neural networks on android smartphones," in *European Conference on Computer Vision*, 2018.

[82] V. Issarny, V. Mallet, K. Nguyen *et al.*, "Dos and don'ts in mobile phone sensing middleware: Learning from a large-scale experiment," in *ACM International Middleware Conference*, 2016.

[83] P. P. Jayaraman, J. B. Gomes, H.-L. Nguyen *et al.*, "Scalable energy-efficient distributed data analytics for crowdsensing applications in mobile environments," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 3, 2015.

[84] P. P. Jayaraman, C. Perera, D. Georgakopoulos *et al.*, "Efficient opportunistic sensing using mobile collaborative platform mosden," in *IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2013.

[85] P. Jesus, C. Baquero, and P. S. Almeida, "A survey of distributed data aggregation algorithms," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, 2014.

[86] C. Jiang, L. Gao, L. Duan *et al.*, "Scalable mobile crowdsensing via peer-to-peer data sharing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 4, 2017.

[87] H. Jin, L. Su, D. Chen *et al.*, "Thanos: Incentive mechanism with quality awareness for mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, 2018.

[88] H. Jin, L. Su, H. Xiao *et al.*, "Incentive mechanism for privacy-aware data aggregation in mobile crowd sensing systems," *IEEE Transactions on Networking*, vol. 26, no. 5, 2018.

[89] J. Jin, J. Gubbi, S. Marusic *et al.*, "An information framework for creating a smart city through internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 2, 2014.

[90] O. Juhlin and M. Östergren, "Time to meet face-to-face and device-to-device," in *ACM Conference on Human-Computer Interaction with Mobile Devices and Services*, 2006.

[91] C. Julien, "Opportunistic crowds: A place for device-to-device collaboration in pervasive crowd applications," in *IEEE International Conference on Pervasive Computing and Communications Workshops*, 2019.

[92] C. Julien, C. Liu, A. L. Murphy *et al.*, "Blend: practical continuous neighbor discovery for bluetooth low energy," in *ACM International Conference on Information Processing in Sensor Networks*. ACM, 2017.

[93] T. Kalbarczyk and C. Julien, "Omni: An application framework for seamless device-to-device interaction in the wild," in *ACM International Middleware Conference*, 2018.

[94] G. Kalic, I. Bojic, and M. Kusek, "Energy consumption in android phones when using wireless communication technologies," in *IEEE International Convention MIPRO*, 2012.

[95] T. Kandappu, A. Mehrotra, A. Misra, M. Musolesi *et al.*, "Pokeme: Applying context-driven notifications to increase worker engagement in mobile

crowd-sourcing," in *ACM Conference on Human Information Interaction and Retrieval*, 2020.

[96] X. Kang, L. Liu, and H. Ma, "Data correlation based crowdsensing enhancement for environment monitoring," in *IEEE International Conference on Communications*, 2016.

[97] ——, "Enhance the quality of crowdsensing for fine-grained urban environment monitoring via data correlation," *Sensors*, vol. 17, no. 1, 2017.

[98] S. S. Kanhere, "Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces," in *International Conference on Distributed Computing and Internet Technology*. Springer, 2013.

[99] M. Karaliopoulos, O. Telelis, and I. Koutsopoulos, "User recruitment for mobile crowdsensing over opportunistic networks," in *IEEE International Conference on Computer Communications*, 2015.

[100] K. Katevas, H. Haddadi, and L. Tokarchuk, "Sensingkit: A multi-platform mobile sensing framework for large-scale experiments," in *ACM International Conference on Mobile Computing and Networking*, 2014.

[101] M. A. A. H. Khan and N. Roy, "Untran: Recognizing unseen activities with unlabeled data using transfer learning," in *IEEE/ACM International Conference on Internet-of-Things Design and Implementation*, 2018.

[102] M. A. Khan, W. Cherif, and F. Filali, "Group owner election in wi-fi direct," in *IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*, 2016.

[103] M. A. Khan, W. Cherif, F. Filali *et al.*, "Wi-fi direct research-current status and future perspectives," *Journal of Network and Computer Applications*, vol. 93, 2017.

[104] S. Kim, C. Robson, T. Zimmerman *et al.*, "Creek watch: pairing usefulness and usability for successful citizen science," in *ACM Conference on Human Factors in Computing Systems*, 2011.

[105] N. Kiukkonen, J. Blom, O. Dousse *et al.*, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *ACM International Conference on Pervasive Services*, 2010.

[106] L. Kong, M. Xia, X.-Y. Liu *et al.*, "Data loss and reconstruction in sensor networks," in *IEEE International Conference on Computer Communications*, 2013.

[107] I. Koukoutsidis, "Estimating spatial averages of environmental parameters based on mobile crowdsensing," *ACM Transactions on Sensor Networks*, vol. 14, no. 1, 2017.

[108] J. Kwak, J. Kim, and S. Chong, "Proximity-aware location based collaborative sensing for energy-efficient mobile devices," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, 2018.

[109] N. D. Lane, "Community-aware smartphone sensing systems," *Internet Computing*, vol. 16, no. 3, 2012.

[110] N. D. Lane, Y. Chon, L. Zhou *et al.*, "Piggyback crowdsensing (pcs) energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities," in *ACM International Conference on Embedded Networked Sensor Systems*, 2013.

[111] N. D. Lane, S. B. Eisenman, M. Musolesi *et al.*, "Urban sensing systems: opportunistic or participatory?" in *ACM International Workshop on Mobile Computing Systems and Applications*, 2008.

[112] N. D. Lane, P. Georgiev, and L. Qendro, "Deepear: robust smartphone audio sensing in unconstrained acoustic environments using deep learning," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015.

[113] N. D. Lane, E. Miluzzo, H. Lu *et al.*, "A survey of mobile phone sensing," *IEEE Communications Magazine*, vol. 48, no. 9, 2010.

[114] J. K. Laurila, D. Gatica-Perez, I. Aad *et al.*, "The mobile data challenge: Big data for mobile computing research," *EPFL Infoscience*, 2012. [Online]. Available: http://infoscience.epfl.ch/record/192489

[115] B. Lefevre, R. Agarwal, V. Issarny *et al.*, "Mobile crowd-sensing as a resource for contextualized urban public policies: a study using three use cases on noise and soundscape monitoring," *Cities & Health*, 2019.

[116] B. Lefevre and V. Issarny, "Matching technological & societal innovations: the social design of a mobile collaborative app for urban noise monitoring," in *IEEE International Conference on Smart Computing*, 2018.

[117] H. Li, T. Li, W. Wang *et al.*, "Dynamic participant selection for large-scale mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 12, 2019.

[118] H. Li, T. Li, and Y. Wang, "Dynamic participant recruitment of mobile crowd sensing for heterogeneous sensing tasks," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, 2015.

[119] M. Li, P. Zhou, Y. Zheng *et al.*, "Iodetector: A generic service for indoor/outdoor detection," *ACM Transactions on Sensor Networks*, vol. 11, no. 2, 2014.

[120] S. Li, Z. Qin, H. Song *et al.*, "A lightweight and aggregated system for indoor/outdoor detection using smart devices," *Future Generation Computer Systems*, 2017.

[121] Z. Li, S. Yang, F. Wu *et al.*, "Holmes: Tackling data sparsity for truth discovery in location-aware mobile crowdsensing," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, 2018.

[122] C. Liu, J. Hua, and C. Julien, "Scents: Collaborative sensing in proximity iot networks," in *IEEE International Conference on Pervasive Computing and Communications Workshops*, 2019.

[123] C. H. Liu, B. Zhang, X. Su *et al.*, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *Systems Journal*, vol. 11, no. 3, 2017.

[124] H. Liu, Y.-S. Ong, X. Shen *et al.*, "When gaussian process meets big data: A review of scalable gps," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[125] J. Liu, D. Sacchetti, F. Sailhan *et al.*, "Group management for mobile ad hoc networks: Design, implementation and experiment," in *ACM International Conference on Mobile Data Management*, 2005.

[126] J. Liu, H. Shen, H. S. Narman *et al.*, "A survey of mobile crowdsensing techniques: A critical component for the internet of things," *ACM Transactions on Cyber-Physical Systems*, vol. 2, no. 3, 2018.

[127] K. Liu, W. Shen, B. Yin *et al.*, "Development of mobile ad-hoc networks over wi-fi direct with off-the-shelf android phones," in *IEEE International Conference on Communications*, 2016.

[128] S. Liu, Z. Zheng, F. Wu *et al.*, "Context-aware data quality estimation in mobile crowdsensing," in *IEEE International Conference on Computer Communications*, 2017.

[129] T. Liu, J. Nicholas, M. M. Theilig *et al.*, "Machine learning for phone-based relationship estimation: The need to consider population heterogeneity," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 4, 2019.

[130] Y. Liu, L. Kong, and G. Chen, "Data-oriented mobile crowdsensing: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, 2019.

[131] Z. Liu, H. Park, Z. Chen *et al.*, "An energy-efficient and robust indoor-outdoor detection method based on cell identity map," *Procedia Computer Science*, vol. 56, 2015.

[132] H. Lu, W. Pan, N. D. Lane *et al.*, "Soundsense: scalable sound sensing for people-centric applications on mobile phones," in *ACM International Conference on Mobile Systems, Applications, and Services*, 2009.

[133] H. Lu, J. Yang, Z. Liu *et al.*, "The jigsaw continuous sensing engine for mobile phone applications," in *ACM International Conference on Embedded Networked Sensor Systems*, 2010.

[134] D. Luo, H. Luo, and C. Zili, "An indoor scene recognition algorithm based on pressure change pattern," in *IEEE International Conference on Intelligent Computation Technology and Automation*, 2015.

[135] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Communications Magazine*, vol. 52, no. 8, 2014.

[136] S. Madhani, M. Tauil, and T. Zhang, "Collaborative sensing using uncontrolled mobile devices," in *IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2005.

[137] N. Maisonneuve, M. Stevens, M. E. Niessen *et al.*, "Noisetube: Measuring and mapping noise pollution with mobile phones," in *Information Technologies in Environmental Engineering*. Springer, 2009.

[138] M. K. Marina, V. Radu, and K. Balampekos, "Impact of indoor-outdoor context on crowdsourcing based mobile coverage analysis," in *ACM International Workshop on All Things Cellular: Operations, Applications and Challenges*, 2015.

[139] M. Marjanović, L. Skorin-Kapov, K. Pripužić *et al.*, "Energy-aware and quality-driven sensor management for green mobile crowd sensing," *Journal of Network and Computer Applications*, vol. 59, 2016.

[140] T. J. Matarazzo, P. Santi, S. N. Pakzad *et al.*, "Crowdsensing framework for monitoring bridge vibrations using moving smartphones," *Proceedings of the IEEE*, vol. 106, no. 4, 2018.

[141] G. Melo, L. Oliveira, D. Schneider *et al.*, "Towards an observatory for mobile participatory sensing applications," in *IEEE International Conference on Computer Supported Cooperative Work in Design*, 2017.

[142] D. Mendez, M. Labrador, and K. Ramachandran, "Data interpolation for participatory sensing systems," *Pervasive and Mobile Computing*, vol. 9, no. 1, 2013.

[143] E. Miluzzo, M. Papandrea, N. D. Lane *et al.*, "Tapping into the vibe of the city using vibn, a continuous sensing application for smartphones," in *ACM International Symposium on From Digital Footprints to Social and Community Intelligence*, 2011.

[144] B. Minasny and A. B. McBratney, "The matérn function as a general model for soil variograms," *Geoderma*, vol. 128, no. 3-4, 2005.

[145] F. Montori, L. Bedogni, and L. Bononi, "A collaborative internet of things architecture for smart cities and environmental monitoring," *IEEE Internet of Things Journal*, vol. 5, no. 2, 2017.

[146] V. F. Mota, D. F. Macedo, Y. Ghamri-Doudane *et al.*, "Managing the decision-making process for opportunistic mobile data offloading," in *IEEE Network Operations and Management Symposium*, 2014.

[147] V. F. Mota, T. H. Silva, D. F. Macedo *et al.*, "Towards scalable mobile crowd-sensing through device-to-device communication," *Journal of Network and Computer Applications*, vol. 122, 2018.

[148] J. Ni, K. Zhang, Y. Yu *et al.*, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[149] O. Ohashi and L. Torgo, "Spatial interpolation using multiple regression," in *IEEE International Conference on Data Mining*, 2012.

[150] V. Pankratius, F. Lind, A. Coster *et al.*, "Mobile crowd sensing in space weather monitoring: the mahali project," *IEEE Communications Magazine*, vol. 52, no. 8, 2014.

[151] F. Pedregosa, G. Varoquaux, A. Gramfort *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, 2011.

[152] D. Peng, F. Wu, and G. Chen, "Pay as how well you do: A quality based incentive mechanism for crowdsensing," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2015.

[153] C. Perera, D. S. Talagala, and C. H. a. Liu, "Energy-efficient location and activity-aware on-demand mobile distributed sensing platform for sensing as a service in iot clouds," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 4, 2015.

[154] C. Perera, A. Zaslavsky, P. Christen *et al.*, "Context-aware sensor search, selection and ranking model for internet of things middleware," in *IEEE International Conference on Mobile Data Management*, 2013.

[155] ——, "Sensing as a service model for smart cities supported by internet of things," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 1, 2014.

[156] A.-H. Phan, P. Tichavskỳ, and A. Cichocki, "Fast alternating ls algorithms for high order candecomp/parafac tensor factorizations," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, 2013.

[157] J. Phuttharak and S. W. Loke, "A review of mobile crowdsourcing architectures and challenges: Toward crowd-empowered internet-of-things," *IEEE Access*, vol. 7, 2018.

[158] I. Podnar Zarko, A. Antonic, and K. Pripužic, "Publish/subscribe middleware for energy-efficient mobile crowdsensing," in *ACM International Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, 2013.

[159] Z. Qin and Y. Zhu, "Noisesense: A crowd sensing system for urban noise mapping service," in *IEEE International Conference on Parallel and Distributed Systems*, 2016.

[160] V. Radu, P. Katsikouli, R. Sarkar *et al.*, "A semi-supervised learning approach for robust indoor-outdoor detection with smartphones," in *ACM International Conference on Embedded Network Sensor Systems*, 2014.

[161] R. Rana, C. T. Chou, N. Bulusu *et al.*, "Ear-phone: A context-aware noise mapping using smart phones," *Pervasive and Mobile Computing*, vol. 17, 2015.

[162] R. K. Rana, C. T. Chou, S. S. Kanhere *et al.*, "Ear-phone: an end-to-end participatory urban noise mapping system," in *ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010.

[163] C. E. Rasmussen, "Gaussian processes in machine learning," in *Springer Summer School on Machine Learning*, 2003.

[164] S. Reddy, A. Parker, J. Hyman *et al.*, "Image browsing, processing, and clustering for participatory sensing: lessons from a dietsense prototype," in *ACM Workshop on Embedded Networked Sensors*, 2007.

[165] F. Restuccia, P. Ferraro, S. Silvestri *et al.*, "Incentme: effective mechanism design to stimulate crowdsensing participants with uncertain mobility," *IEEE Transactions on Mobile Computing*, vol. 18, no. 7, 2018.

[166] F. Sailhan, V. Issarny, and O. Tavares-Nascimiento, "Opportunistic multiparty calibration for robust participatory sensing," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, 2017.

[167] M. Saloni, C. Julien, A. L. Murphy *et al.*, "Lasso: A device-to-device group monitoring service for smart cities," in *IEEE International Smart Cities Conference*, 2017.

[168] S. Sarma, N. Venkatasubramanian, and N. Dutt, "Sense-making from distributed and mobile sensing data: A middleware perspective," in *ACM Annual Design Automation Conference*, 2014.

[169] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, 2017.

[170] I. Schweizer, R. Bärtl, A. Schulz *et al.*, "Noisemap-real-time participatory noise maps," in *International Workshop on Sensing Applications on Mobile Phones*, 2011.

[171] X. Sheng, J. Tang, and W. Zhang, "Energy-efficient collaborative sensing with mobile phones," in *IEEE International Conference on Computer Communications*, 2012.

[172] W. Sherchan, P. P. Jayaraman, S. Krishnaswamy *et al.*, "Using on-the-move mining for mobile crowdsensing," in *IEEE International Conference on Mobile Data Management*, 2012.

[173] J. Shi, R. Zhang, Y. Liu *et al.*, "Prisense: privacy-preserving data aggregation in people-centric urban sensing systems," in *IEEE International Conference on Computer Communications*, 2010.

[174] W. Shi, J. Cao, Q. Zhang *et al.*, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, 2016.

[175] V. Singh, D. Chander, U. Chhaparia *et al.*, "Safestreet: An automated road anomaly detection and early-warning system using mobile crowdsensing," in *IEEE International Conference on Communication Systems & Networks*, 2018.

[176] V. Sivaraman, J. Carrapetta, K. Hu *et al.*, "Hazewatch: A participatory sensor system for monitoring air pollution in Sydney," in *IEEE Conference on Local Computer Networks-Workshops*, 2013.

[177] M. Stevens and E. D'Hondt, "Crowdsourcing of pollution data using smartphones," in *Workshop on Ubiquitous Crowdsourcing*, 2010.

[178] M. Talasila, R. Curtmola, and C. Borcea, "Mobile crowd sensing," in *Handbook of Sensor Networking: Advanced Technologies and Applications*. CRC Press, 2015.

[179] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, "Device-to-device communication in 5g cellular networks: challenges, solutions, and future directions," *IEEE Communications Magazine*, vol. 52, no. 5, 2014.

[180] D. d. C. Teixeira, A. C. Viana, M. S. Alvim *et al.*, "Deciphering predictability limits in human mobility," in *ACM International Conference on Advances in Geographic Information Systems*, 2019.

[181] G. Texier and V. Issarny, "Leveraging the power of the crowd and offloading urban iot networks to extend their lifetime," in *IEEE International Symposium on Local and Metropolitan Area Networks*, 2018.

[182] N. Thepvilojanapong, S. Konomi, Y. Tobe *et al.*, "Opportunistic collaboration in participatory sensing environments," in *ACM International Workshop on Mobility in the Evolving Internet Architecture*, 2010.

[183] M. Umer, L. Kulik, and E. Tanin, "Spatial interpolation in wireless sensor networks: localized algorithms for variogram modeling and kriging," *Geoinformatica*, vol. 14, no. 1, 2010.

[184] R. Ventura, V. Mallet, and V. Issarny, "Assimilation of mobile phone measurements for noise mapping of a neighborhood," *Journal of the Acoustical Society of America*, vol. 144, no. 3, 2018.

[185] R. Ventura, V. Mallet, V. Issarny *et al.*, "Evaluation and calibration of mobile phones for noise monitoring application," *Journal of the Acoustical Society of America*, vol. 142, no. 5, 2017.

[186] H. Wackernagel, *Multivariate geostatistics: an introduction with applications*. Springer Science & Business Media, 2013.

[187] E. Wang, Y. Yang, J. Wu *et al.*, "An efficient prediction-based user recruitment for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, 2017.

[188] J. Wang, F. Wang, Y. Wang *et al.*, "Social-network-assisted worker recruitment in mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 7, 2018.

[189] ——, "Allocating heterogeneous tasks in participatory sensing with diverse participant-side factors," *IEEE Transactions on Mobile Computing*, vol. 18, no. 9, 2019.

[190] J. Wang, Y. Wang, S. Helal *et al.*, "A context-driven worker selection framework for crowd-sensing," *International Journal of Distributed Sensor Networks*, vol. 12, no. 3, 2016.

[191] J. Wang, Y. Wang, D. Zhang *et al.*, "Learning-assisted optimization in mobile crowd sensing: A survey," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 1, 2018.

[192] L. Wang, D. Zhang, A. Pathak *et al.*, "Ccs-ta: Quality-guaranteed online task allocation in compressive crowdsensing," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015.

[193] L. Wang, D. Zhang, Y. Wang *et al.*, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Communications Magazine*, vol. 54, no. 7, 2016.

[194] L. Wang, D. Zhang, and H. Xiong, "effsense: energy-efficient and cost-effective data uploading in mobile crowdsensing," in *ACM International Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, 2013.

[195] L. Wang, D. Zhang, H. Xiong *et al.*, "ecosense: Minimize participants' total 3g data cost in mobile crowdsensing using opportunistic relays," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 6, 2016.

[196] L. Wang, D. Zhang, Z. Yan *et al.*, "effsense: A novel mobile crowd-sensing framework for energy-efficient and cost-effective data uploading," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 12, 2015.

[197] W. Wang, Q. Chang, Q. Li *et al.*, "Indoor-outdoor detection using a smart phone sensor," *Sensors*, vol. 16, no. 10, 2016.

[198] X. Wang, R. Jia, X. Tian *et al.*, "Location-aware crowdsensing: Dynamic task assignment and truth inference," *IEEE Transactions on Mobile Computing*, vol. 19, no. 2, 2018.

[199] D. Weir, S. Rogers, R. Murray-Smith *et al.*, "A user-specific machine learning approach for improving touch accuracy on mobile devices," in *ACM International Symposium on User Interface Software and Technology*, 2012.

[200] J. Wen, S. Loke, J. Indulska *et al.*, "Sensor-based activity recognition with dynamically added context," in *EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2015.

[201] J. Weppner and P. Lukowicz, "Bluetooth based collaborative crowd density estimation with mobile phones," in *IEEE International conference on pervasive computing and communications*, 2013.

[202] D. C. Wheeler and A. Páez, "Geographically weighted regression," in *Handbook of applied spatial analysis*. Springer, 2010.

[203] I. H. Witten, E. Frank, M. A. Hall *et al.*, *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann, 2017.

[204] C. Xiang, P. Yang, and S. Xiao, "Counter-strike: accurate and robust identification of low-level radiation sources with crowd-sensing networks," *Personal and Ubiquitous Computing*, vol. 21, no. 1, 2017.

[205] Y. Xiao, P. Simoens, P. Pillai *et al.*, "Lowering the barriers to large-scale mobile crowdsensing," in *ACM International Workshop on Mobile Computing Systems and Applications*, 2013.

[206] H. Xiong, D. Zhang, G. Chen *et al.*, "icrowd: Near-optimal task allocation for piggyback crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, 2015.

[207] L. Xu, X. Hao, N. D. Lane *et al.*, "More with less: Lowering user burden in mobile crowdsourcing through compressive sensing," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015.

[208] Q. Xu and R. Zheng, "Mobibee: a mobile treasure hunt game for location-dependent fingerprint collection," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016.

[209] X. Xu, R. Ansari, A. Khokhar *et al.*, "Hierarchical data aggregation using compressive sensing (hdacs) in wsns," *ACM Transactions on Sensor Networks*, vol. 11, no. 3, 2015.

[210] Y. Xu, Y. Zhu, and Z. Qin, "Urban noise mapping with a crowd sensing system," *Wireless Networks*, vol. 25, no. 5, 2019.

[211] J. Yang, E. Munguia-Tapia, and S. Gibbs, "Efficient in-pocket detection with mobile phones," in *ACM International Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, 2013.

[212] S. Yang, J. Bian, L. Wang *et al.*, "Edgesense: Edge-mediated spatial-temporal crowdsensing," *IEEE Access*, vol. 7, 2018.

[213] Y. Yang, W. Liu, E. Wang *et al.*, "A prediction-based user selection framework for heterogeneous mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, 2019.

[214] J.-W. Yoo and K. H. Park, "A cooperative clustering protocol for energy saving of mobile devices with wlan and bluetooth interfaces," *IEEE Transactions on Mobile Computing*, vol. 10, no. 4, 2010.

[215] Ö. Yürür, C. H. Liu, Z. Sheng *et al.*, "Context-awareness for mobile sensing: A survey and future directions," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, 2014.

[216] W. Zamora, C. T. Calafate, J.-C. Cano *et al.*, "A survey on smartphone-based crowdsensing solutions," *Mobile Information Systems*, vol. 2016, 2016.

[217] A. Zanella, N. Bui, A. Castellani *et al.*, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, 2014.

[218] Y. Zhan and H. Haddadi, "Towards automating smart homes: contextual and temporal dynamics of activity prediction," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2019.

[219] D. Zhang, L. Wang, H. Xiong *et al.*, "4w1h in mobile crowd sensing," *IEEE Communications Magazine*, vol. 52, no. 8, 2014.

[220] X. Zhang, Z. Yang, W. Sun *et al.*, "Incentives for mobile crowd sensing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, 2015.

[221] X. Zhang, L. Shu, Z. Huo *et al.*, "A short review of constructing noise map using crowdsensing technology," in *Springer International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2017.

[222] Y. Zheng, T. Liu, Y. Wang *et al.*, "Diagnosing new york city's noises with ubiquitous data," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2014.

[223] Z. Zhou, J. Feng, B. Gu *et al.*, "When mobile crowd sensing meets uav: Energy-efficient task assignment and route planning," *IEEE Transactions on Communications*, vol. 66, no. 11, 2018.

[224] Q. Zhu, M. Y. S. Uddin, N. Venkatasubramanian *et al.*, "Spatiotemporal scheduling for crowd augmented urban sensing," in *IEEE International Conference on Computer Communications*, 2018.