IAM - Interpolation and Aggregation on the Move: Collaborative Crowdsensing for Spatio-temporal Phenomena

Francoise Sailhan, Valérie Issarny, Yifan Du

IAM – Interpolation and Aggregation on the Move: Collaborative Crowdsensing for Spatio-temporal Phenomena

Yifan Du
yifan.du@inria.fr
Inria Paris
France

Françoise Sailhan
francoise.sailhan@cnam.fr
CNAM Paris
France

Valérie Issarny
valerie.issarny@inria.fr
Inria Paris
France

ABSTRACT

Crowdsensing allows citizens to contribute to the monitoring of their living environment using the sensors embedded in their mobile devices, e.g., smartphones. However, crowdsensing at scale involves significant communication, computation, and financial costs due to the dependence on cloud infrastructures for the analysis (e.g., interpolation and aggregation) of spatio-temporal data. This limits the adoption of crowdsensing by activists although sorely needed to inform our knowledge of the environment. As an alternative to the centralized analysis of crowdsensed observations, this paper introduces a fully distributed interpolation-mediated aggregation approach running on smartphones. To achieve so efficiently, we model the interpolation as a distributed tensor completion problem, and we introduce a lightweight aggregation strategy that anticipates the likelihood of future encounters according to the quality of the interpolation. Our approach thus shifts the centralized post-processing of crowdsensed data to distributed pre-processing on the move, based on opportunistic encounters of crowdsensors through state-of-the-art D2D networking. The evaluation using a dataset of quantitative environmental measurements collected from 550 crowdsensors over 1 year shows that our solution significantly reduces –and may even eliminate– the dependence on the cloud infrastructure, while it incurs a limited resource cost on end devices. Meanwhile, the overall data accuracy remains comparable to that of the centralized approach.

KEYWORDS

Crowdsensing, Ubiquitous Sensing, Pervasive Computing, Aggregation, Interpolation, Opportunistic Relay

1 INTRODUCTION

Mobile crowdsensing is an attractive sensing paradigm to monitor the urban environment such as noise level, ambient temperature, luminance, air pressure, relative humidity, etc. [11, 17, 26, 38]. Citizens may contribute valuable spatio-temporal observations using the low-cost, yet powerful, sensors embedded in their smartphones/tablets, with the GPS for positioning and the Internet access for uploading. Furthermore, pollution is an increasing societal concern, and through opportunistic crowdsensing, people are able to be aware of it in a mutually beneficial way. However, the collected observations often cover the urban space and time unevenly and sparsely, due to the dependence on the mobility of citizens throughout the city [33]. State-of-the-art crowdsensing systems address this shortcoming through the centralized analysis –incl. aggregation and interpolation– of observations provided to cloud infrastructure servers [8, 18, 23, 31]. The implemented centralization 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 inference) [29].

We argue that enabling fully decentralized crowdsensing systems, including the underlying large-scale data analysis, is key to the democratization of environmental monitoring using crowdsensors. This is the focus of our paper that introduces the IAM (*Interpolation and Aggregation on the Move*) distributed solution, which builds upon the two following trends to support crowdsensing-based environmental monitoring at scale:

(i) IAM views crowdsensors as "*social sensors*" that outfit human, as opposed to mere physical equipment that senses the environment and transfers data. In particular, mobile crowdsensors often encounter each other during sensing campaigns or as part of the daily routine of their owners [20]. We focus on *opportunistic* crowdsensing, where people follow their daily routine without being directly involved in the sensing task [6, 11]. At the micro level, behavioral signatures (i.e., routines) as well as recurrent encounter patterns reflect the underlying relational dynamics of organizations or communities to which the user is affiliated. Furthermore, end-users are more likely to share their data, especially in short encounters, as the risk of losing their anonymity is lower [14].

(ii) The collaborative and ubiquitous processing of crowdsensed observations improves the overall efficiency of the system in terms of resource consumption [6, 28, 35], regarding both the infrastructure and contributing devices. It also allows end users to collaborate and share knowledge effortlessly and with little or no cost. In particular, D2D collaboration enhances mobile edge computing by enabling the sharing of heterogeneous computing and communication resources between powerful mobile end devices [2, 20].

Concretely, IAM allows a fully distributed, collaborative approach to crowdsensing: crowdsensors interpolate their observation of the phenomenon, and they aggregate the respective data in an opportunistic 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. At the same time, P2P wireless ad hoc network technology (e.g., Wi-Fi Direct) enables the discovery of nearby mobile devices and the exchange of data between peers, making the collaboration possible [6, 20]. Some crowdsensing systems already exploit the P2P collaboration of crowdsensors as they meet [5, 34, 39]. 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.

In summary, the paper makes the following contributions to enable ubiquitous and collaborative crowdsensing:

(1) A *fully distributed approach to the aggregation and interpolation of crowdsensing data on the move* (§ 3), which exploits the smartphone's capability to perform 3D tensor completion. In particular, we thoroughly analyze and compare eligible state-of-the-art interpolation methods, which leads us to leverage the Gaussian Process Regression (§ 3.2) that produces the most cost-effective inference for spatio-temporal phenomena along with an estimation of the inference uncertainty. Another advantage compared to alternative interpolation approaches (i.e., ordinary kriging & tensor decomposition) is that all the specified parameters are learned and the most relevant kernel was selected following related assessment. 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* (§ 3.3) 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 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* (§ 4) of the IAM solution and its *performance evaluation using a 1-year crowdsensing dataset* (§ 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.

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.

2.1 Data Aggregation

Related crowdsensing observations are usually aggregated using an average [19, 29] or a weighted average [13, 28] value, although more complex functions may be found in the literature depending on the observed phenomenon [12]. 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) or an unstructured (e.g., flooding) routing; it may even implement a combination of the two. Our solution is based on the unstructured 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 § 4). Still, the distributed aggregation protocol must be complemented with the interpolation of missing values [32].

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.

Ordinary kriging [8, 23] is one of the methods that is widely applied as it supports a fine-grained interpolation at each location over a 2D space. It estimates the value at an unobserved location based on values at observed locations and it performs well as long as observations are uniformly distributed. In particular, performance greatly degrades for large amounts of missing data and the computational cost gets prohibitive [7].

Compressive sensing is a recent alternative approach to the interpolation of data for the production of observation maps, while dealing with sparse observations [7]. It has been used to infer/interpolate urban-scale physical phenomena from crowdsensed observations stored in a 2D matrix [8, 33, 36]. However, physical phenomena often have more than two modes of variation and are therefore best

represented as multi-dimensions observations, which leads to using 3D tensors [16, 26, 40]. The matrix/tensor completion problem is usually solved by matrix/tensor decomposition and one major drawback is that it trades uncertainty for efficiency, as there is no confidence interval associated with the inference.

Multiple regression is another approach to infer missing values [9, 25], which transforms the interpolation problem into regressions. This inspired us to use the Gaussian Process Regression (GPR) to solve the tensor completion problem. GPR is a well-known and general approach applied in machine learning [22], which supports a non-parametric and interpretable Bayesian model. GPR naturally supports multi-dimension and it does not require the data to follow Gaussian distribution. GPR enables estimating the level of uncertainty associated with the produced model. We compare the three interpolation approaches in the evaluation section (§ 5) where we show that GPR is the best suited to support distributed interpolation across mobile crowdsensors.

2.3 Centralized vs Distributed Data Fusion in Crowdsensing Platforms

The great majority of crowdsensing platforms applies centralized analysis to the sensing data: the cloud/edge server first aggregates the raw crowdsensor measurements and then interpolates the missing observations. Existing platforms adopt various interpolation and aggregation methods, while considering different types of sensors (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 [8]. Some platforms [16, 17, 26, 38] 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 analysis may leverage historical data to improve the accuracy of the generated map. 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 in [3].

The distributed interpolation and aggregation of sensing data have deserved less attention than the centralized counterparts, with most solutions targeting Wireless Sensor Networks (WSN). A localized, distributed interpolation & aggregation scheme based on kriging is introduced in [30] for a tree-structured WSN; it allows inferring a phenomenon over the holes that the static WSN does not cover. In [37], 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 [39]. 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 a stochastic gradient descent and they exchange factor vectors with the crowdsensors that are in the same WLAN. 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 is applied is less accurate than GPR, which we leverage within IAM.

Different to 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 aggregation, which benefits from the encounter with other crowdsensors and is hence not constrained by any hierarchical/tree network structure.

3 LIGHTWEIGHT DECENTRALIZED CROWDSENSING DATA FUSION

The IAM solution to decentralized crowdsensing takes benefits of today's smartphones capacity, while limiting the resulting additional resource consumption on devices. That is, IAM implements lightweight collaborative sensor data fusion across the participating crowdsensing devices as they are in D2D communication range.

3.1 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 [28] prior to contribute measurements). IAM supports the periodic monitoring of $\mathcal{P}$ over (a possibly large) area $\mathcal{A}$ and a given time period $\mathcal{D}$ using the contributions of the m crowdsensors.

We represent the data that each crowdsensor collects as a concise 3D tensor where the first two dimensions refer to the spatial space (i.e., latitude and longitude) and the third one to the temporal domain (time). In particular, we discretize the target region $\mathcal{A}$ into $I \times J$ areas, which are cells of equal spatial size. We also discretize $\mathcal{D}$ into K time slots of equal durations. 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 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 allows the opportunistic combination of the various tensors $\mathcal{Y}_s$ from the contributing crowdsensors s, using 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 $\mathcal{D}$. As a base design choice, crowdsensors first apply interpolation over their local tensor $\mathcal{Y}_s$ prior to engage in the collaborative aggregation, which is key to the –both local and global– resource-efficiency of the IAM approach, while supporting the computation of a globally accurate knowledge.

3.2 Spatio-temporal Interpolation using Gaussian Process Regression

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 $\mathcal{D}$, interpolation allows completing the tensor by estimating missing cells. The resulting (denser) tensor is denoted $\hat{\mathcal{Y}}_s$. The quality of the estimation can be established based on the approximation error (i.e., residual). The overall residual is then given by $\mathcal{E} = \|\mathcal{Y} - \hat{\mathcal{Y}}\|$.

Let Ω be the set of *observed cells* by 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 values 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 $\hat{\mathcal{Y}}$:

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

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 2.2, we leverage the *Gaussian Process Regression* [22] to compute $\hat{\mathcal{Y}}$ out of $\mathcal{Y}$. That is, we assume that y follows a Gaussian Process (Gaussian distribution over functions), i.e.:

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

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 close to each other (Euclidean distance) are likely to have 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 [27] for an overview). In our case, the Matérn [24] induced the lowest error and execution time compared to alternative kernels (see §5).

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 $\hat{\Sigma}^2 \in \mathbb{R}^{I \times J \times K}$ is also maintained, in which each element $\hat{\sigma}^2(x)$ refers to the variance of corresponding $\hat{y}(x)$.

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 relative low number of contributed cells (see § 5).

3.3 Opportunistic P2P 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 in selecting randomly one of the two crowdsensors to take in charge the aggregation and relay the resulting tensor, where we assume that all the crowdsensors have equal resource budgets. 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). 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'.

Algorithm 1
Asymmetric P2P aggregation at s

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

Algorithm 1 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., mobility behavior, available resource, and fault tolerance), for which we may leverage state of the art algorithms [34], while keeping the process energy-efficient.

Algorithm 2 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 $y_s(x)$

Algorithm 2
Crowdsensing data tensors merger (s, s') at the mainstay s

Input: data tensor $\mathcal{Y}_s$, mask tensor $\mathcal{M}_s$, variance tensor Σ^2_s, merge count n_s
Input: data tensor $\mathcal{Y}_{s'}$, mask tensor $\mathcal{M}_{s'}$, variance tensor $\Sigma^2_{s'}$, merge count $n_{s'}$
Output: aggregated data tensor $\mathcal{Y}_{ss'}$, mask tensor $\mathcal{M}_{ss'}$, variance tensor $\Sigma^2_{ss'}$, 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: $\mathcal{Y}_{ss'} += (\beta_s \mathcal{Y}_s / \Sigma^2_s + \beta_{s'} \mathcal{Y}_{s'} / \Sigma^2_{s'}) / (\beta_s \Sigma_s^{-2} + \beta_{s'} \Sigma_{s'}^{-2}) * (\neg \mathcal{M}_s * \neg \mathcal{M}_{s'})$
6: $\Sigma^2_{ss'} \leftarrow (\beta_s \Sigma_s^{-2} + \beta_{s'} \Sigma_{s'}^{-2})^{-1}$
7: $n_{ss'} \leftarrow n_s + n_{s'}$
8: $\mathcal{M}_{ss'} \leftarrow \mathcal{M}_s \vee \mathcal{M}_{s'}$
9: **return** $\mathcal{Y}_{ss'}, \mathcal{M}_{ss'}, n_{ss'}, \Sigma^2_{ss'}$

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:* a merged incremental average is applied.
- *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 and thus it is selected over the inference.
- *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 below.

3.4 Aggregating Multiple GPR Inferences

The *Generalized Product-of-Expert* is a method that allows combining estimated results that have been inferred by several experts (e.g., interpolations 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 [4]:

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

where β_s is a weighting parameter that allows tuning the relative importance of a crowdsensor s according to 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 [1], 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:

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

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

A generalized Product-of-Expert of a GPR allows merging several inferences (i.e., $m = 2$ inferences in our case) in a cost-effective way, as it is characterized by a $O(n)$ time complexity with n being the number of merged points. The opportunistic aggregation on the move is asymmetric, as captured by the loss function D. Our evaluation 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 (see § 5).

4 IAM-BASED CROWDSENSING SYSTEM DESIGN

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, etc.) may build upon IAM to produce such a 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 covers.

Figure 1 illustrates the resulting operation of a IAM-based crowdsensing system: crowdsensors sense, pre-process and interpolate the data, then 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 computation 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 composes the tensors. 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.

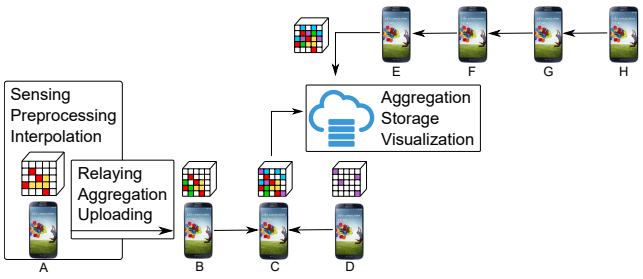

Figure 1: A IAM-based crowdsensing system

4.1 Aggregation Process at the Crowdsensors

Algorithm 3
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 3 outlines the periodic process that IAM runs on every participating crowdsensor s to compute and relay/upload $\mathcal{Y}_s$. The process iterates over 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 the one of the peer that s meets (Line 4 – See detail in § 3.3).

At the end of the current time window, the spatio-temporal interpolation is applied on the associated local tensor to infer missing values (Line 6 – See detail in § 3.2). We highlight that the interpolation runs locally, only once and prior to the aggregation process running over the next time window. This 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 node meets, which determines the mainstay.

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

4.2 Prototype Implementation

The IAM solution assumes an ad hoc framework supporting the opportunistic meeting and collaboration of peer nodes using D2D communication, such as our middleware presented in [6] or other solutions in [10, 20, 21]. The IAM prototype is thus specifically focused on the implementation of the distributed, collaborative interpolation and aggregation, further targeting Android smartphones/tablets as crowdsensors.

The prototype is available at https://github.com/sensetogether/IAM; it requires a Python 3 environment as well as the following third-party packages for data analytics: NumPy (https://numpy.org) handling multi-dimensional arrays (tensors) and Scikit-learn (https://scikit-learn.org) implementing various machine learning algorithms that is used for GPR training/inference. The key components of our prototype implementation are:

- *Pre-processing* reads the sensed 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 so as 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.
- *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 Σ^2. The interpolation has a $O(n^3)$ 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 Σ^2, respectively, following Algorithm 1. The aggregation process on each crowdsensor has a $O(p)$ time complexity, where p is the number of P2P meetings.

5 EVALUATION

IAM supports the opportunistic aggregation of the sensing data along with the interpolation of a physical phenomenon, provided relevant measurements from the crowdsensors. We assess the effectiveness of the IAM distributed approach using a dataset collected with the Ambiciti crowdsensing application (available on Google Play and App Store) for environmental monitoring.

5.1 Experiment Setup and Dataset

The experiment supporting our evaluation focuses on urban noise monitoring using crowdsensing. The dataset contains the measurements collected by the Ambiciti application in an opportunistic way [11]. The partner company Ambiciti provided us the dataset used for evaluation, which specifically relates to the contributions gathered in Paris over year 2017 and includes about 950k entries from 550 crowdsensors.

The dataset entries are tuples of the form: *<device ID, latitude index, longitude index, timestamp index, observation value>* with the *observation value* being the average 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. Instead, the sound levels in dB(A) are first converted into their energy equivalents, and then the energy equivalents are averaged algebraically, and finally the resulted energy equivalent is converted back to its dB(A) value.

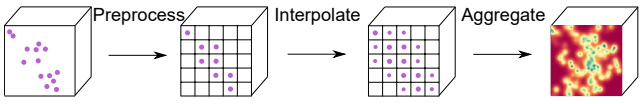

Figure 2: Producing a noise map using IAM (zoomed scale)

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 100×100×24 tensor,

which contains at most $240k$ of data entries. Figure 2 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. This highlights that the dataset is very sparse, which is the case with most crowdsensing applications and our reference application in particular.

We use the *Root Mean Square Error* (RMSE) to evaluate the accuracy of the interpolated and aggregated tensors. 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.

In the following evaluation, the experiments are run either on a DELL Precision 7520 workstation, used both as the centralized server (§ 5.2) and for simulation (§ 5.3), or on Android smartphones as end device testbed (§ 5.4). The D2D communication is assumed to be supported by the WiFi-Direct protocol, and we use the distance to detect P2P encounters [6].

5.2 Assessment of the Interpolation Methods

Figure 3 compares the robustness of the three inference strategies that are commonly used to interpolate physical phenomena (see § 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 largest amount of crowdsensing data were collected. The same 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 RMSE interquartile range and median. However, some wrong inferences lead to abnormal values as illustrated by the high RMSE mean value of 47. Similarly but to a lower extent, CP-ALS shows some abnormal RMSE. Instead, GPR-Matern provides both an accurate and robust inference: a stable RMSE without outliers –hence characterized by the lowest variance– is observed.

Figure 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 is constant in $O(RIJK)$ 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 $O(n^3)$ 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. 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. While the memory consumption associated with CP-ALS (resp. OK-Gaussian) is of $3.258MB$ with a variance of 0.422 (resp. $4.644MB$ 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. Overall, GPR-Matern is the most efficient in terms of accuracy and robustness, while its execution time is also relatively lower.

Focusing on GPR, we assess the accuracy of the following kernels in Figure 5: constant, RBF, 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.

5.3 Distributed vs Centralized 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 2), 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 2). 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 1 & 2, and we set $\beta_s = 1.5$ (resp. $\beta_{s'} = 0.5$) for crowdsensor s with lower D (resp. s' with higher D).

As illustrated in Figure 6, 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).

We further 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,

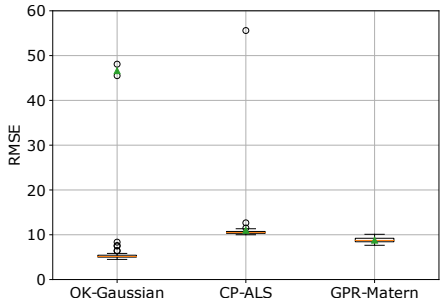

Figure 3: Interpolation accuracy - RMSE

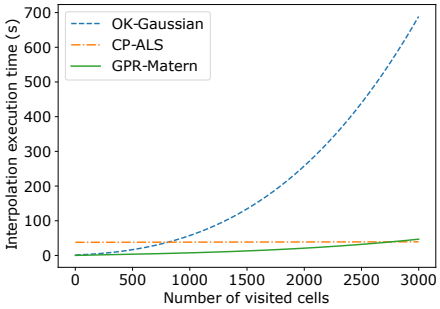

Figure 4: Interpolation execution time

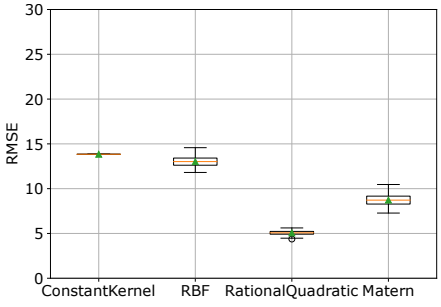

Figure 5: Kernel accuracy - RMSE

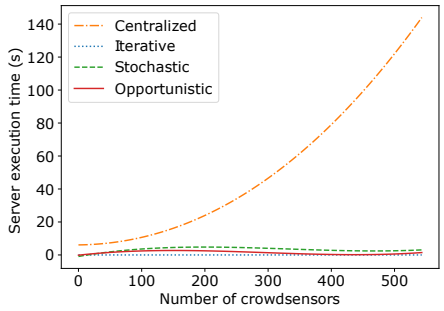

Figure 6: Execution time at the server only

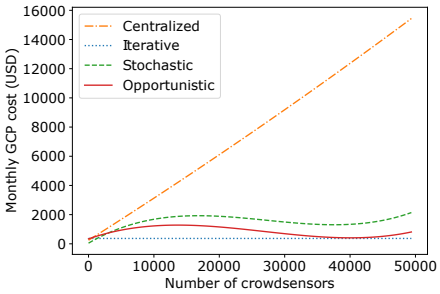

Figure 7: Monthly financial cost of a cloud-based deployment

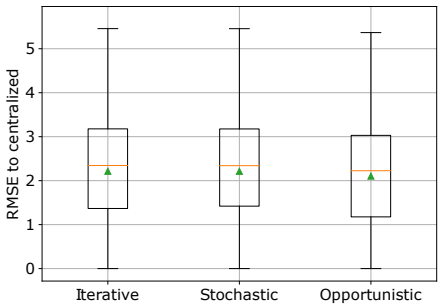

Figure 8: Aggregation accuracy - RMSE

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. In a nutshell, the price depends on the amount of network traffic, the amount of storage needed, and the load associated with the computation (e.g., execution time associated with the analysis). Figure 7 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 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 infrastructure server, thereby enabling the wider adoption of crowdsensing systems by communities of users concerned with environmental monitoring.

Figure 8 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 (resp. median) equals 2.213 (resp. 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 (resp. median) of 2.102 (resp. 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.

5.4 IAM Impact on the Device Resources

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.

Figure 9 compares the amount of traffic uploaded to the server 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. Furthermore, 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;

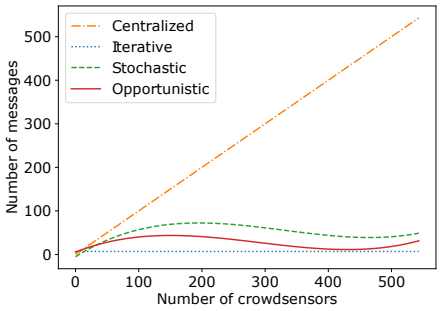

Figure 9: Directly uploaded messages

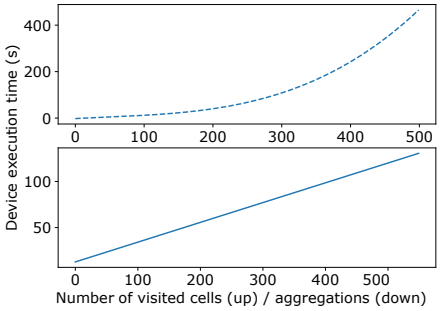

Figure 10: On-device execution time

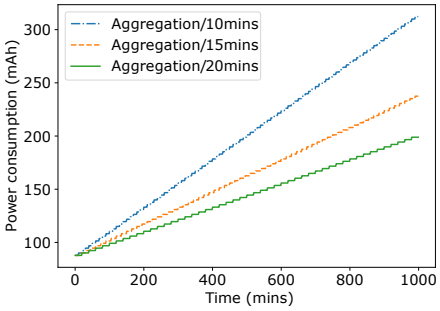

Figure 11: Energy consumed by IAM

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.

Then, we run experiments on a SAMSUNG GALAXY S7 smartphone embedding a 3000 mAh battery capacity. Figure 10 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 11 estimates the energy consumed by a smartphone that implements the interpolation and opportunistic aggregation when the 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 [15], uploading via cellular network consumes 8.9 (resp. 4) times of the D2D relay energy via Bluetooth (resp. Wi-Fi).

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 [2, 5, 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.

7 CONCLUSION

One of the major benefits of crowdsensing is the possibility to monitor environmental phenomena at the urban scale, simply leveraging the abundance and capacity of people's smartphones. 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.

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.

REFERENCES

[1] Yanshuai Cao and David J Fleet. 2014. Generalized product of experts for automatic and principled fusion of Gaussian process predictions. In *Modern Nonparametrics 3: Automating the Learning Pipeline workshop at NIPS*.

[2] Xu Chen, Lingjun Pu, Lin Gao, et al. 2017. Exploiting massive D2D collaboration for energy-efficient mobile edge computing. *IEEE Wireless Communications* 24, 4 (2017).

[3] Yun Cheng, Xiaoxi He, Zimu Zhou, et al. 2020. MapTransfer: Urban Air Quality Map Generation for Downscaled Sensor Deployments. In *ACM International Conference on Internet of Things Design and Implementation*.

[4] Marc Peter Deisenroth and Jun Wei Ng. 2015. Distributed Gaussian processes. In *International Conference on Machine Learning*.

[5] Ngoc Do, Ye Zhao, Cheng-Hsin Hsu, et al. 2016. Crowdsourced mobile data transfer with delay bound. *ACM Transactions on Internet Technology* 16, 4 (2016).

[6] Yifan Du, Francoise Sailhan, and Valerie Issarny. 2020. Let opportunistic crowdsensors work together for resource-efficient, quality-aware observations. In *IEEE International Conference on Pervasive Computing and Communications*.

[7] Khalid Eldrandaly and Ahmed Abdelmouty. 2017. Spatio-temporal Interpolation: Current Practices and Future Prospects. *International Journal of Digital Content Technology and its Applications* 11 (06 2017).

[8] Michele Girolami, Stefano Chessa, Gaia Adami, et al. 2017. Sensing interpolation strategies for a mobile crowdsensing platform. In *IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*.

[9] Paul Harris, AS Fotheringham, R Crespo, et al. 2010. The use of geographically weighted regression for spatial prediction: an evaluation of models using simulated data sets. *Mathematical Geosciences* 42, 6 (2010).

[10] Takamasa Higuchi, Hirozumi Yamaguchi, Teruo Higashino, et al. 2014. A neighbor collaboration mechanism for mobile crowd sensing in opportunistic networks. In *IEEE International Conference on Communications*.

[11] Valerie Issarny, Vivien Mallet, Kinh Nguyen, et al. 2016. Dos and don'ts in mobile phone sensing middleware: Learning from a large-scale experiment. In *ACM/IFIP International Middleware Conference*.

[12] Paulo Jesus, Carlos Baquero, and Paulo Sérgio Almeida. 2014. A survey of distributed data aggregation algorithms. *IEEE Communications Surveys & Tutorials* 17, 1 (2014).

[13] Haiming Jin, Lu Su, Houping Xiao, et al. 2018. Incentive mechanism for privacy-aware data aggregation in mobile crowd sensing systems. *IEEE Transactions on Networking* 26, 5 (2018).

[14] Oskar Juhlin and Mattias Östergren. 2006. Time to meet face-to-face and device-to-device. In *ACM Conference on Human-Computer Interaction with Mobile Devices and Services*.

[15] Goran Kalic, Iva Bojic, and Mario Kusek. 2012. Energy consumption in android phones when using wireless communication technologies. In *IEEE International Convention MIPRO*.

[16] Xu Kang, Liang Liu, and Huadong Ma. 2016. Data correlation based crowdsensing enhancement for environment monitoring. In *IEEE International Conference on Communications*.

[17] Xu Kang, Liang Liu, and Huadong Ma. 2017. Enhance the quality of crowdsensing for fine-grained urban environment monitoring via data correlation. *MDPI Sensors* 17, 1 (2017).

[18] Linghe Kong, Mingyuan Xia, Xiao-Yang Liu, et al. 2013. Data loss and reconstruction in sensor networks. In *IEEE International Conference on Computer Communications*.

[19] Ioannis Koukoutsidis. 2017. Estimating spatial averages of environmental parameters based on mobile crowdsensing. *ACM Transactions on Sensor Networks* 14, 1 (2017).

[20] Youngki Lee, Younghyun Ju, Chulhong Min, et al. 2012. Comon: Cooperative ambience monitoring platform with continuity and benefit awareness. In *ACM International Conference on Mobile Systems, Applications, and Services*.

[21] Chenguang Liu, Jie Hua, and Christine Julien. 2019. Scents: Collaborative sensing in proximity iot networks. In *IEEE International Conference on Pervasive Computing and Communications Workshops*.

[22] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, et al. 2020. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[23] Diego Mendez, Miguel Labrador, and Kandethody Ramachandran. 2013. Data interpolation for participatory sensing systems. *Pervasive and Mobile Computing* 9, 1 (2013).

[24] Budiman Minasny and Alex B McBratney. 2005. The Matérn function as a general model for soil variograms. *Geoderma* 128, 3-4 (2005).

[25] Orlando Ohashi and Luis Torgo. 2012. Spatial interpolation using multiple regression. In *IEEE International Conference on Data Mining*.

[26] Zhaokun Qin and Yanmin Zhu. 2016. NoiseSense: A crowd sensing system for urban noise mapping service. In *IEEE International Conference on Parallel and Distributed Systems*.

[27] Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Springer Summer School on Machine Learning*.

[28] Francoise Sailhan, Valérie Issarny, and Otto Tavares-Nascimiento. 2017. Opportunistic multiparty calibration for robust participatory sensing. In *IEEE International Conference on Mobile Ad Hoc and Sensor Systems*.

[29] Jing Shi, Rui Zhang, Yunzhong Liu, et al. 2010. Prisense: privacy-preserving data aggregation in people-centric urban sensing systems. In *IEEE International Conference on Computer Communications*.

[30] Muhammad Umer, Lars Kulik, and Egemen Tanin. 2010. Spatial interpolation in wireless sensor networks: localized algorithms for variogram modeling and Kriging. *Geoinformatica* 14, 1 (2010).

[31] Raphaël Ventura, Vivien Mallet, and Valérie Issarny. 2018. Assimilation of mobile phone measurements for noise mapping of a neighborhood. *The journal of the acoustical society of America* 144, 3 (2018).

[32] Jiangtao Wang, Yasha Wang, Daqing Zhang, et al. 2018. Learning-assisted optimization in mobile crowd sensing: A survey. *IEEE Transactions on Industrial Informatics* 15, 1 (2018).

[33] Leye Wang, Daqing Zhang, Yasha Wang, et al. 2016. Sparse mobile crowdsensing: challenges and opportunities. *IEEE Communications Magazine* 54, 7 (2016).

[34] Leye Wang, Daqing Zhang, Haoyi Xiong, et al. 2016. ecoSense: Minimize participants' total 3G data cost in mobile crowdsensing using opportunistic relays. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47, 6 (2016).

[35] Yu Xiao, Pieter Simoens, Padmanabhan Pillai, et al. 2013. Lowering the barriers to large-scale mobile crowdsensing. In *ACM International Workshop on Mobile Computing Systems and Applications*.

[36] Liwen Xu, Xiaohong Hao, Nicholas D Lane, et al. 2015. More with less: Lowering user burden in mobile crowdsourcing through compressive sensing. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing*.

[37] Xi Xu, Rashid Ansari, Ashfaq Khokhar, et al. 2015. Hierarchical data aggregation using compressive sensing (HDACS) in WSNs. *ACM Transactions on Sensor Networks* 11, 3 (2015).

[38] Yanan Xu, Yanmin Zhu, and Zhaokun Qin. 2019. Urban noise mapping with a crowd sensing system. *Wireless Networks* 25, 5 (2019).

[39] Sijia Yang, Jiang Bian, Licheng Wang, et al. 2018. EdgeSense: Edge-mediated spatial-temporal crowdsensing. *IEEE Access* 7 (2018).

[40] Yu Zheng, Tong Liu, Yilun Wang, et al. 2014. Diagnosing New York city's noises with ubiquitous data. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing*.