2012 IEEE 13th International Conference on Mobile Data Management

Using On-the-move Mining for Mobile Crowdsensing

Wanita Sherchan*, Prem P. Jayaraman*, Shonali Krishnaswamy*†, Arkady Zaslavsky‡, Seng Loke§ and Abhijat Sinha*
*Faculty of Information Technology, Monash University, Email: firstname.lastname@monash.edu
†Institute for Infocomm Research (I2R), Singapore
‡CSIRO ICT Centre, Email: arkady.zaslavsky@csiro.au
§Department of Comp. Sc. and Comp. Eng., La Trobe University, Email: s.loke@latrobe.edu.au

***Abstract*—In this paper, we propose and develop a platform to support data collection for mobile crowdsensing from mobile device sensors that is under-pinned by real-time mobile data stream mining. We experimentally show that mobile data mining provides an efficient and scalable approach for data collection for mobile crowdsensing. Our approach results in reducing the amount of data sent, as well as the energy usage on the mobile phone, while providing comparable levels of accuracy to traditional models of intermittent/continuous sensing and sending. We have implemented our Context-Aware Real-time Open Mobile Miner (CAROMM) to facilitate data collection from mobile users for crowdsensing applications. CAROMM also collects and correlates this real-time sensory information with social media data from both Twitter and Facebook. CAROMM supports delivering real-time information to mobile users for queries that pertain to specific locations of interest. We have evaluated our framework by collecting real-time data over a period of days from mobile users and experimentally demonstrated that mobile data mining is an effective and efficient strategy for mobile crowdsensing.**

I. Introduction

Mobile devices are increasingly becoming the central computing and communication device in people's lives. Devices today are equipped with a growing number of sophisticated embedded sensors such as an accelerometer, digital compass, gyroscope, GPS, microphone, light intensity sensor, and camera. This creates the opportunity to develop applications that leverage on the sensing capability of these mobile devices. These applications can be broadly classified into two categories- personal and community sensing, based on the type of entity being monitored and the purpose of monitoring. In personal sensing applications, the focus is on monitoring an individual or the context surrounding an individual for the individual's benefit. For example, activity recognition (e.g., running, walking, exercising) of an individual for personal record-keeping or health monitoring. Typically, the sensed information is not shared with anyone. In community sensing, also referred to as group sensing [1] and mobile crowdsensing [2], the focus is on monitoring of large-scale phenomena that cannot be measured using information from a single individual. The purpose is to collect information from a large group of people and analyse and use that information for the benefit of the group. For example, intelligent transportation systems that use traffic congestion monitoring and air pollution level monitoring require speed and air quality information from a large number of individuals. Such systems would be able to provide accurate and useful information only when there is a critical mass of people providing information from their daily commutes, which can be aggregated to determine congestion and pollution levels in cities.

Personal sensing combined with social networks has given rise to 'mobile social networks' where sensed user context information is shared with the user's social network. Most of these applications use social networks as a means for disseminating sensed information such as in CenceMe [3], or obtaining user preferences such as in Serendipity [4], WhozThat [5] and SocialFusion [6]. To the best of our knowledge, using social networks/media themselves as a source of information for community sensing is an emerging focus in this area. Social media provides another source of information that can be valuable for information services, where the goal is to monitor not only sensory data regarding a location, but also user opinions and experiences regarding that location. This provides the opportunity to create a real-time holistic view of entities such as places of interest. To this regard, we propose a framework for mobile crowdsensing, Context- Aware Real-time Open Mobile Miner (CAROMM), to facilitate sensor data collection from mobile users and correlate this real-time information with social media data from both Twitter (http://twitter.com/) and Facebook (http://www.facebook.com/).

An integral part of a mobile crowdsensing framework such as CAROMM is sensing and sending of information. Both personal and community sensing require mobile devices to continuously sense, process and upload sensed data to the cloud/remote servers. Since mobile devices are continuously sensing, processing and/or uploading sensed data, several issues become significant due to the mobile device and its operational and computational context. There are several key factors that need to be considered and addressed in order for mobile crowdsensing to be effective [2]. Firstly, it is imperative that the *data collection* process from mobile devices is cost-efficient for both the device performing the sensing, as well as the networks that need to scale for large volumes of users sending sensed data. Secondly, mobile crowdsensing needs to have infrastructure to receive, manage and analyse large volumes of real-time data streams using the pay-per-use cloud

978-0-7695-4713-8/12 $26.00 © 2012 IEEE
DOI 10.1109/MDM.2012.58

computing platforms. Thirdly, sensing using mobile devices requires participation from the user and willingness to allow collection of sensor data, and hence use preferences and privacy-preserving operations for mobile crowdsensing need to considered. In this context, there need to be incentives in place to facilitate such large-scale mobile crowdsensing. For example, an incentive of reduced data transfer costs for supporting citizen surveillance operations and so on need to be considered.

The focus of this paper is on the data collection dimension as it is the first step for mobile crowdsensing. We propose and develop a mobile data mining driven approach for highly scalable and cost-efficient data collection for mobile crowdsensing. The local analytics that we perform results in reduced data transfer and reduced energy utilisation on the mobile device, and yet, captures information at the same level of granularity/accuracy as continuous sensing-transmission. Furthermore, the mobile analytics techniques that we leverage are themselves resource-aware and energy-efficient [7]. Furthermore, to minimise costs associated with frequent data transfer between mobile devices and the cloud, we explore intelligent techniques such as using sensor data to determine when to collect data, e.g., not capturing videos/pictures when the light intensity is too low. We implement our proposed CAROMM system to leverage cloud technologies to enable collation and processing of huge amounts of real-time data generated by mobile phone sensors, as well as correlation of information feeds from social media to specifically support real-time queries pertaining to locations of interest. The CAROMM system forms the basis for evaluating the feasibility and validity of mobile analytics as an effective mechanism for supporting large-scale mobile crowdsensing.

The rest of this paper is organised as follows. Section II discusses related work and compares and contrasts our work with the current state-of-the-art. Section III presents our proposed CAROMM framework. Section IV describes the data collection module of CAROMM which is the core contribution of this paper, and also includes cost models for bandwidth usage, energy usage and accuracy of data collection for mobile crowdsensing. Section V presents the implementation and evaluation of the data collection model. Finally, Section VI presents the conclusions and future directions.

II. Related Work

The aim of the review is to focus on context inference using mobile sensing and energy efficient strategies for sensing and uploading in mobile crowdsensing applications.

A variety of research projects have focused on extracting user context with the help of mobile devices enabling the emergence of personal, group, and community-scale sensing applications such as Citysense [8], Serendipity [4], WhozThat [5], CenceMe [3], and SocialFusion [6]. These mobile sensing applications can be broadly categorised into two types. The first type focuses on importing social context into the user's local context using mobile devices. Serendipity [4], WhozThat [5] and SocialFusion [6] can be considered of this type. In all of these works, use of social network information is limited to obtaining user's preferences from their social network profiles. The second category of mobile sensing applications export user's local context to their social networks, e.g., Citysense and CenceMe. Citysense [8] utilises users' GPS locations to provide a visualization of mobile user concentration in an area. No social media is involved in this work. CenceMe [3] mines mobile data provided by iPhone sensors to infer user actions and allows publishing of the mined data to the social networks. In contrast to the literature, CAROMM exports user's local context from mobile devices to social context in the cloud, where this information is aggregated with social media data to create a holistic view/context of the user as well as the user's environment. In addition, it performs mobile data mining (learning from the data on-board the device) to reduce the overheads of data collection.

With respect to addressing resource-constraints in mobile phone based continuous sensing systems, [9] proposes a framework called EEMSS that uses a hierarchical sensor management strategy to recognize user states and detect state transitions. It aims to improve device battery life by powering only a minimum set of sensors and optimizing the sensor duty cycles. Musolesi *et al.* [10] propose different techniques to optimize the user state uploading process. The focus is on maintaining stable/reliable user state updates regardless of network connectivity. Our approach is comparable to the online strategies. We use continuous on-device clustering of sensed data to identify changes in clusters and upload only when change is detected. We compare the efficiency and accuracy of this approach with upload of raw data streams. In contrast, in [10], uploading strategies are based on a set of fixed user state data and therefore not compared with upload of raw data. Techniques proposed in [9] and [10] are orthogonal to ours and can coexist with our solution to provide a holistic approach to energy-efficient mobile crowdsensing.

III. The Context Aware Real-time Open Mobile Miner (CAROMM) Framework

A. Motivational Scenario

Nowadays, everyone carries a mobile device with them. Most of these mobile devices come with increasingly sophisticated list of sensors that are able to capture various context information pertaining to the user and their environment. Prevalence and wide uptake of social media such as Facebook (http://www.facebook.com/) and Twitter (http://twitter.com/) and photo sharing sites such as Flickr (http://www.flickr.com/) have shown that users are willing to share information. Mobile crowdsensing aims to leverage these phenomena with a view to delivering real-time sensory information to a range of applications such as location-based service delivery, location-based social networking and citizen surveillance.

We propose and develop our CAROMM system to support the scenario where people use a mobile application to upload sensory data to the cloud. The sensory data could have mixed input such as multimedia/videos, twitter/social media streams, text, activities, location, temperature, time, device orientation,

speed, and movement of the device. An application on the cloud processes this mixed data to operate as a real-time location information service. Thus, CAROMM supports the provision of collated information to users who request for real-time information about specific locations of interest. The real-time information delivered includes aggregation of sensor feeds from mobile users in that location pertaining to physical phenomena such as light levels, temperature, estimates of crowd intensity, as well videos/photos (and the context in which those photos were taken such as day, night), and recent social media posts.

This scenario is motivated by future application scenarios where telecommunication providers could offer real-time information services to users. Such information services would require real-time sensory data from a critical mass of mobile users. To motivate users to share their context information using a mobile application, the provider could offer users access to personalised services, or promote offers such as free texting or discounts if the user allows the application to obtain a certain number of feeds within a given time period. The sensory information thus gathered can then be used by the provider to offer information services to other users who will be charged for accessing this information. Users of the information services will have access to real-time information pertaining to places of interest. For example, a user interested to go skiing in a particular location might want to have access to the real-time temperature, wind speed and humidity information, and real-time experience updates from people already in that location. The provider benefits because it is cheaper to collect real-time updates from mobiles/people already in that location, hence the set-up cost is low. Users of these information services benefit from access to real-time information that includes not only sensory data from the location but also user experience data. Other potential applications could be citizen surveillance where video analysis in the cloud can be used to detect unusual/suspicious activity in a location, and geo-fencing and tracking of personnel, for example policemen on the beat using this application to send information regularly to their headquarters.

B. Overview of the CAROMM System

In this section, we present an overview of the Context Aware Real-time Open Mobile Miner (CAROMM) framework for enabling mobile crowdsensing applications. CAROMM has several features: (i) capture different types of stream data from mobile devices, (ii) process, manage and analyse this data along with the relevant contextual information associated with them (e.g. associate light-intensity levels with pictures/videos, and social media information pertaining to locations of interest), and (iii) facilitate real-time queries from mobile users on the collected (and analysed) data.

While we have implemented the software system for collecting, collating and information retrieval of such real-time data for mobile crowdsensing, the core theoretical contribution of this paper is in the data collection model. Given the limited resources on the mobile devices and the fact that users need

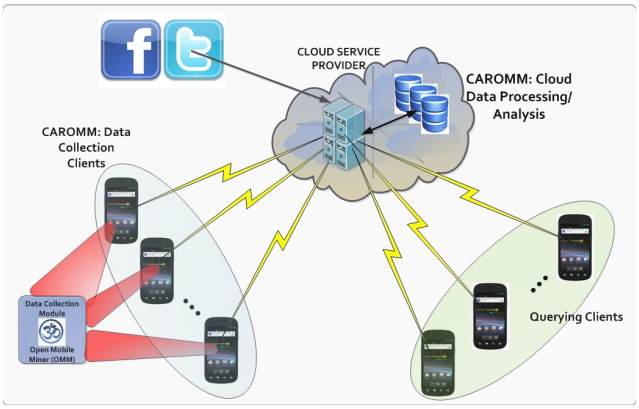

Fig. 1. The CAROMM Framework

the devices to perform their normal functionality along with data collection, the data collection needs to be highly resource efficient. Furthermore, given that mobile crowdsensing needs to be large scale in terms of the number of users involved, it also needs to be efficient in terms of the data transfer and bandwidth use. Therefore, simply sensing and uploading all data to the cloud is not a preferred solution for data collection. To address both of these issues, we propose to leverage real-time mobile data stream mining on the sensed data to reduce the amount of data sent and the energy consumption on the device. We describe our mobile data stream mining based solution for mobile crowdsensing in detail in Section IV. In fact in [2], "local analytics" has been suggested as a key enabler for mobile crowdsensing. Our work is the first evaluation of the feasibility of local analytics/mobile data mining for this domain.

Figure 1 shows the Context Aware Real-time Open Mobile Miner (CAROMM) framework. The framework consists of three main modules- a Data Collection Client and a Querying Client residing on the mobile devices, and a Data Processing Module residing on the cloud. The Data collection Module captures sensory data, performs local continuous real-time stream mining on the data and uploads analysed information to the Data Processing Module in the cloud where further analysis, management, and fusion of the incoming multiple streams needs to be performed. To intelligently send only analysed information from each device, we use resource-aware clustering on the sensory data to identify significant changes in the situation. This reduces the frequency and amount of data transferred from each mobile device to the cloud, while at the same time ensuring that important information is not lost. Clustering provides fine-grained control over when to send updates. For example, with GPS data, subtle changes which are not viewed as significant enough to warrant an update will be ignored but significant changes can be detected and used to perform updates. Furthermore, the resource-aware mobile data stream clustering technique that we have developed [7] can be controlled via its parameters to tune the sensitivity to change detection. This total control over 'granularity' of change is one major advantage of using CAROMM.

The Querying Client on mobile devices send user queries to

the Data Processing Module and receive and display the results obtained. The Data Processing Module consists of Social Media Data Collection and Query Processing. This module aggregates information obtained from all sources (mobile devices and social media) to provide contextual information in response to the user queries obtained from the Querying Client. Various approaches and analytics can be used for combining mixed media data. This opens interesting avenues for future research and the stream analytics for cloud platforms to support mobile crowdsensing is planned as further research in this project.

The proposed CAROMM framework aims to answer these research questions: How can different types of data obtained from different sources be used to define context for an entity such as a place of interest? How can such data be meaningfully integrated to infer what is happening in any given place of interest? What kinds of analytics can be done based on mobile sensor data and social media data? Does it make sense to do mobile mining to reduce the energy and bandwidth consumption? In answering these questions, the focus of this paper is on leveraging mobile data mining for mobile crowdsensing. We consider energy efficient and bandwidth efficient mobile data mining that can give an acceptable level of granularity. We compare the cost of sending raw sensed data at specified time intervals to the cloud for processing versus our mobile analytics based approach. We discuss the cost models for data transfer, energy usage and accuracy for these two approaches in Section IV-B. We implement and evaluate our data collection framework using these cost models in Section V.

IV. CAROMM Data Collection Module

A. Data Collection Module Architecture

In this section, we describe the architecture of the data collection module within the CAROMM framework. The data collection module of CAROMM addresses the challenges in collecting, processing/analysing and uploading data sensed from user environments. We take advantage of the mobile device's processing capabilities and the plethora of embedded sensors to perform on-the-move mining of collected sensor data. The proposed data collection module enables, as demonstrated later, cost-efficient collection and processing of mobile device sensor data using data stream mining. Figure 2 presents the architecture of the data collection module.

The data collection module has five main components, namely, Interface Controller, Data Analysis-Cluster engine, Data Collection Manager, Cloud Upload Manager, and Sensor and Media Manager. The data collection module runs on the mobile device interfacing with sensors available on the device. The proposed approach does not require any additional hardware sensors other than sensors available on the mobile device.

Interface Controller: This component is the graphical user component presented to the user. This component instantiates the Data Collection Manager. The interface controller provides the user with data collection options namely sensing interval,

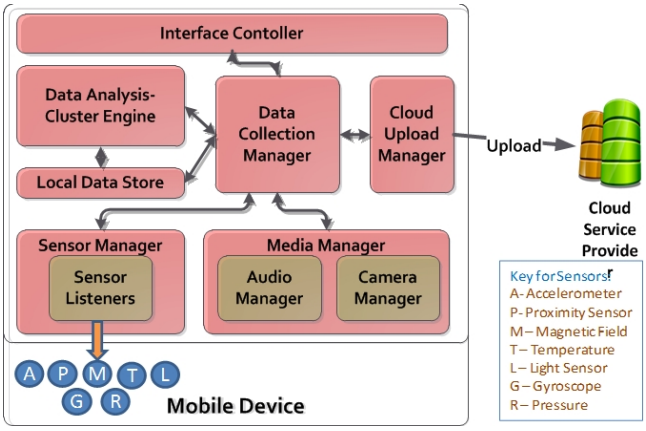

Fig. 2. CAROMM-Data Collection Module Architecture

upload interval and sensor selection (to choose which sensors are accessed by the module).

Data Analysis-Cluster Engine: The data analysis-cluster engine is the core component of the proposed data collection module. It handles all processing and analysis of data. The analysis engine performs continual mining over the sensed data. For continual data mining, we use the generic open source toolkit for mobile data mining (OMM)[11]. We have used the Light Weight Cluster (LWC) algorithm implemented in OMM toolkit to perform clustering over sensed data. OMM is a powerful resource aware mobile data miner. OMM adapts its functioning depending on resource availability on the mobile device. The LWC algorithm uses data adaptation techniques to match high-speed data streams and achieves optimum accuracy based on available resources [7]. The LWC algorithm is an outcome of our previous works in the area of mobile data stream mining. The LWC algorithm extracted from [7] is presented in a pseudo code format in Figure 3. The data analysis engine uploads periodic updates of clustered data to the cloud using the Cloud Manager. Moreover, the data analysis engine incorporates change detection, i.e., it has the ability to determine significant change in sensed data. Any significant change in the sensed data results in the data being uploaded to the cloud. Further, we have also implemented a timeout procedure that will upload clustered data to the cloud if no change is detected over a certain period of time. We chose a periodic upload interval to enable data uploads when no change is detected in the environment. The periodic upload interval for the clustering approach is set to a much higher value as against the raw data collection approach. The analysis engine performs data mining on multiple attributes. Hence, the change detection works across a multitude of sensed data. The use of on-the-move mining helps the data collection module upload data to the cloud only when a change in the environmental context is detected. We show in our experiments that this can result in significant savings in energy and bandwidth usage while still retaining a high level of data accuracy.

Data Collection Manager: The data collection manager acts as a coordinator between the components of the data

```
Begin LWC
2  Data items arrive in sequence with a data rate.
3  The algorithm starts by considering the first point as a center.
4  Compare any new data item with the centers to find the distance.
5  If the distance for all the centers is greater than a threshold, the
   new item is considered as a new center;
   else increase the weight for the center that has the shortest
   distance between the data item and the center by 1 and let the
   new center equals the weighted average.
6  Repeat 4 and 5.
7  If the number of centers = k (according to the available memory)
   then create a new centers      vector.
8  Repeat 4, 5, 6, and 7.
9  If memory is full then re-cluster (incrementally integrate the
   clusters)
   End
```

Fig. 3. LWC Algorithm [7]

collection module. It instantiates the various sensors on the phone using the sensor manager. It forwards sensed data to the analysis engine for on-the-move mining, and clustered datasets from the analysis engine to the cloud manager for further processing. The data collection manager also handles timers and asynchronous call backs from other components.

Cloud Storage Manager: The cloud storage manager is responsible for uploading data to the cloud using the mobile device's network connection.

Sensor and Media Manager: The sensor manager is responsible for interfacing with the device's sensors. It periodically queries the device's sensors for data and passes it to the data collection manager. The media manager handles any communication between the data collection module and the device's camera and microphone. The data collection module has the capability to collect various types of data such as sensory, photos, videos and voice. The data collected from the camera, namely photos and videos, are passed to the data analysis engine before uploading to the cloud.

The algorithms implemented in the proposed data collection module are presented in pseudo code format in Figure 4. The algorithm *dataCollection* includes the functions performed by the Data Collection Manager, Sensor and Media Manager and Cloud Manager initiated by the interface controller. The algorithms *dataAnalysis* and *changeDetect* include the functions of the data analysis module performing on-the-move mining.

A key objective of a good data collection architecture is to significantly reduce battery and network bandwidth usage by taking advantage of on-mobile device capabilities and at the same time attaining high level of data accuracy. The proposed data collection module of CAROMM framework achieves the above objectives. We now present cost models to validate the efficiency of the data collection module in terms of energy and bandwidth consumption and data accuracy.

B. Cost Models for Mobile Data Stream Mining

In this section, we develop cost models for two data collection approaches for mobile crowdsensing:

1) Model 1: all processing in the cloud: In this mode, the mobile devices sense context data periodically and upload to the cloud. No processing is done on the device.
2) Model 2: local mobile data analytics on-board the device: In this mode, each mobile device performs continuous sensing and local data stream mining on the collected sensor data and only mined data is uploaded to the cloud. This aims to reduce costs related to energy usage and data transmission.

```
Begin dataCollection
2  Instantiate sensors
3  Repeat 4,5,6,7 every x seconds
4        sense data
5        dataAnalysis(SensedData)
6        if data to upload:
7                upload SensedData to Cloud
   End
-----------------------------------------------------------------
Begin dataAnalysis(SensedData)
2    if no cluster exist:
3        create new cluster for SensedData
4    else:
5        changeDetect(SensedData)
6        if new cluster created
7                return SensedData
     End
-----------------------------------------------------------------
     Begin changeDetect(SensedData)
2    Iterate through existing list of cluster
3    For each cluster get their attributes
4    Compute distance between SensedData and existing clusters
5    If distance < threshold:
6     //member of existing cluster
7          Get the closest cluster for the SensedData
8     Increase the weight of the existing cluster
9     Return closest cluster
10   else:
11    // new cluster
12    if current number of clusters < maximum cluster size
13      Create a new cluster for the SensedData with attributes
14    else:
15      Re-cluster (incrementally integrate the clusters)
     End
```

Fig. 4. Data Collection Module Algorithms

We develop two cost models: a data transmission cost model and an energy usage cost model. These cost models aim to compare the cost related to the above two data collection approaches. Therefore, in these models we do not consider the cost associated with mining social media data. Social media data is only mined on the cloud.

Data Transmission Cost Model: We evaluate the cost of data transmission in terms of consumed bandwidth for any time period t. The cost of data transfer is directly proportional to the amount of data transferred between the mobile device (M) and the cloud (C).

$$Cost_{dt} \propto \textit{total data transferred from M to C} \quad (1)$$

Therefore, the data transmission cost $Cost_{dt}$ can be represented as follows:

$$Cost_{dt} = x * \textit{total data transferred from M to C} \quad (2)$$

where x is a constant.

- Model 1: All processing in the cloud: amount of data transferred *total data transferred from M to C* is high as all sensed data is uploaded.
- Model 2: Local processing on the mobile should result in lower *total data transferred from M to C*, and hence, lower data transmission cost.

Let, $Cost_{dt^{raw}}$ be the data transmission cost for Model 1 (i.e., using raw data), and $Cost_{dt^{clust}}$ be the data transmission cost for Model 2,(i.e., using on-device clustering). Then, assuming that for similar devices the constant x is same in the case of both raw and clustering approaches, the savings

on the data transmission cost for mobile data mining can be evaluated as

$$\begin{aligned} \textit{Bandwidth Gain} &= \frac{Cost_{dt^{raw}}}{Cost_{dt^{clust}}} \\ &= \frac{\textit{(total data transferred from M to C)}^{raw}}{\textit{(total data transferred from M to C)}^{clust}} \end{aligned} \tag{3}$$

As part of the evaluation of CAROMM data collection module, we compute the average data transmission cost savings using Model 2 versus Model 1 in Section V-B.

Energy Usage Cost Model: We model the cost of energy usage in terms of battery drain for any time period t. Cost incurred due to energy drain is composed of drain due to sensing, drain due to processing/mining in the device and drain due to data transfer.

$$Cost_{ed} = Cost_{edS} + Cost_{edPr} + Cost_{edDt} \tag{4}$$

$Cost_{edS}$ represents energy drain due to sensing. It is directly proportional to the frequency of sensing. Energy expended due to sensing is the same in both modes of operation and can therefore be discounted.

$$Cost_{edS} \propto \textit{freq. of sensing} \tag{5}$$

$$Cost_{edS} = \textit{ a * freq. of sensing} \tag{6}$$

where a is a constant.

Using the same bandwidth, larger amount of data transfer requires more time and hence hence results in more energy drain. Similarly, more frequent data transfer requires more energy. These relationships can be expressed as follows:

$$Cost_{edDt} \propto \textit{total data transferred from M to C} \tag{7}$$

$$Cost_{edDt} \propto \textit{num. of data transfers from M to C} \tag{8}$$

$$\begin{aligned} Cost_{edDt} = y \textit{ * total data transferred from M to C} \\ \textit{* num. of data transfers from M to C} \end{aligned} \tag{9}$$

where y is a constant. From 4, 6 and 9, the energy drain cost can be represented as follows:

$$\begin{aligned} Cost_{ed} = \textit{ a * freq. of sensing} + Cost_{edPr} \\ +y \textit{ * total data transferred from M to C} \\ \textit{* num. of data transfers from M to C} \end{aligned} \tag{10}$$

- Model 1: All processing on the cloud: In this mode, there is no processing on the mobile, therefore, $Cost_{edPr} \simeq 0$. Therefore, the energy drain cost consists of only the data transfer cost and the sensing cost. In this instance, the cost can be represented as:

$$\begin{aligned} Cost_{ed}^{raw} = \textit{ a * freq. of sensing} \\ +y \textit{ * total data transferred from M to C} \\ \textit{* num. of data transfers from M to C} \end{aligned} \tag{11}$$

- Model 2: Local processing on the mobile. In this case, the energy drain cost due to mobile data mining $Cost_{edPr}$ becomes significant. Typically, energy drain due to processing is directly proportional to the amount of data being processed. Energy drain may also be affected by the clustering algorithm used, however, we do not consider this in these cost models.

$$Cost_{edPr}^{clust} \propto \textit{total size of data accumulated on M} \tag{12}$$

$$Cost_{edPr}^{clust} = z \textit{ * total size of data accumulated on M} \tag{13}$$

where z is a constant. From 10 and 13, the energy drain cost for clustering can be computed as:

$$\begin{aligned} Cost_{ed}^{clust} = \textit{ a * freq. of sensing} \\ +z \textit{ * total size of data accumulated on M} \\ +y \textit{ * total data transferred from M to C} \\ \textit{* num. of data transfers from M to C} \end{aligned} \tag{14}$$

The *total size of data accumulated on M* depends on the *freq. of sensing* and caching mechanisms used. This variable may not vary much in the two models. Similarly, for fair comparison, *freq. of sensing* would be the same in the two models. Therefore, significant reduction of *total size of data accumulated on M* and *num. of data transfers from M to C* in any given time period t will result in reduction of energy drain cost. Performing mobile data mining aims to reduce these variables.

For energy usage evaluation of the CAROMM data collection model, we use the ratio of energy usage cost for raw approach versus the clustering approach. This is evaluated as:

$$\textit{Energy Gain} = \frac{Cost_{ed}^{raw}}{Cost_{ed}^{clust}} \tag{15}$$

We run several experiments to test and validate our cost models. The experimental evaluations are presented in Section V-B

In addition to the cost models, we also develop data accuracy model for evaluating CAROMM data collection model. Let sf be the sensing frequency, and uf be the data upload frequency. With raw approach, all sensed data are uploaded, i.e., $sf = uf$. If sf is high enough, it results in high accuracy as sensing and upload of data are real-time. However, this results in higher data transmission cost $Cost_{dt}^{raw}$ and higher energy usage cost $Cost_{ed}^{raw}$. For the same sf, the role of mobile data mining is to reduce the upload frequency uf such that there is no significant reduction in accuracy. In the raw approach, if uf is reduced significantly, it will decrease $Cost_{dt}^{raw}$ and $Cost_{ed}^{raw}$, but it may also reduce the data accuracy as the uploaded data is no longer current. This is especially applicable in cases of frequent changes in sensed data. With the use of clustering, the uf is affected only by changes in the sensed values, and therefore, may result in higher accuracy as all major changes are detected and reflected. We compare the data accuracy of the CAROMM clustering approach with the raw approach in Section V-B.

V. Implementation

In this section, we present the implementation and the experimental evaluation of the data collection module with respect to the cost models presented in Section IV-B.

A. Implementation Details

The CAROMM data collection module has been implemented on android-based smart phones using android SDK v2.2 (http://developer.android.com/sdk/index.html). The software development was done in Eclipse using the Android Development Tool (ADT) plug-in. In this section, we use the term collector to refer to the data collection module that performs

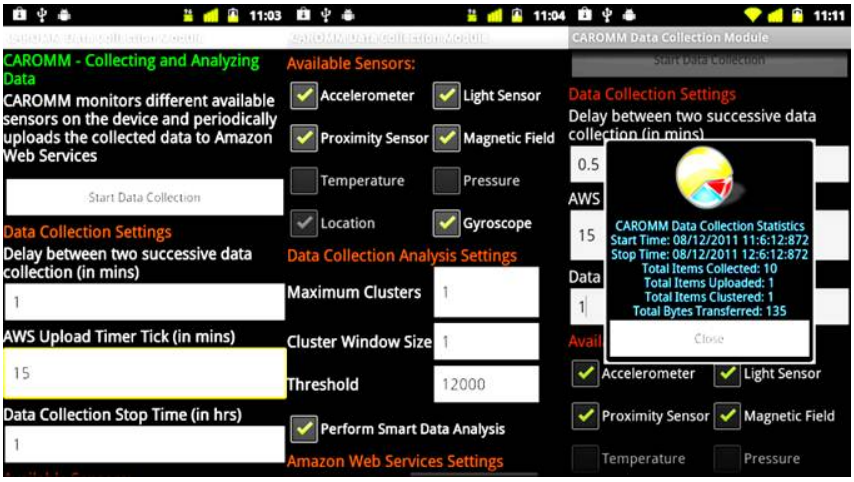

Fig. 5. Data Collection Module Main Screen Screenshots

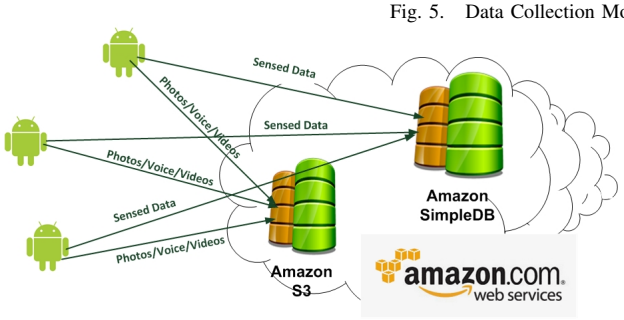

Fig. 6. Amazon Cloud Service Upload Architectures

the operations of interfacing with smart phone sensor to collect sensor data, process and analyse the sensed data and upload the analysed data to the cloud. Figure 5 shows a set of screen dumps of the collector's graphical user interface. As illustrated, the user has the option to choose different parameters including collection interval, upload interval, clustering parameters and cloud setting. The interface also provides an option to take photos and view data collection statistics when required. The clustering parameter threshold is used to control the granularity of the cluster. By changing the granularity of the cluster, we can control the cluster's sensitivity to change in sensed data. For example, having a very high threshold will make the system insensitive to significant change in sensed data and having a low threshold will make it very sensitive to minor changes in sensed data. The threshold value directly influences the data accuracy.

The cloud service provider used for our implementation is Amazon. We used Amazon SimpleDB (http://aws.amazon.com/simpledb/), a non-relational highly scalable data store. To store objects namely photos, videos and voice data, we used Amazon Simple Storage Service (S3) (http://aws.amazon.com/s3/). Our implementation uploads objects to S3 and maintains a key to the upload in SimpleDB. Figure 6 shows the architecture of the interaction between the mobile device's data collection module and Amazon cloud services.

As highlighted in the beginning of this paper, the key focus of this paper is to propose, evaluate and validate the on-the-move mining-based data collection module of CAROMM. Hence, we focus on the analysis on the mobile device and do not detail the analysis on the cloud. Our current implementation uses Amazon Elastic Compute Cloud (EC2)(http://aws.amazon.com/ec2/) to host a web service that can be used to query information pertaining to a location consisting of data from different sources namely mobile device sensors collected by users and social media including Facebook and Twitter. The web service answering user queries is a REST web service developed in JAVA using Net beans and Glassfish v3. This service is hosted on the cloud to answer queries in real-time. It performs simple data aggregation before returning the query results. Further, we have also developed a simple querying client to validate the accuracy of our proposed approach for performing on-the-move mining. The query client is developed for android-based smart phones using android SDK v2.2 (http://developer.android.com/sdk/index.html) and has the capability to query the web service to retrieve and visualise the results. Figure 7 shows some screen dumps of the android client making a query to the Amazon cloud service. The screen dumps in Figure 7 show the result of a query for the suburb *Carnegie* in Melbourne, Australia. The current implementation has some basic analytics which converts sensor readings into some meaningful values. For example, based on the light sensor values, photos are classified as taken on *Good*, *Moderate* and *Poor* lighting conditions.

B. Experimental Evaluation

The key challenge in mobile crowdsensing is to enable cost-efficient collection of environmental data from multiple sources over long periods of time. The term cost-efficiency is used to represent energy spent in collecting, analysing and uploading data, bandwidth usage while uploading data and data accuracy. A good approach to save bandwidth and energy usage is to reduce the number of uploads as research shows that communication is a major factor affecting battery usage. On the other hand, reducing data uploads might result in loss of data accuracy. Hence, it is imperative for a data

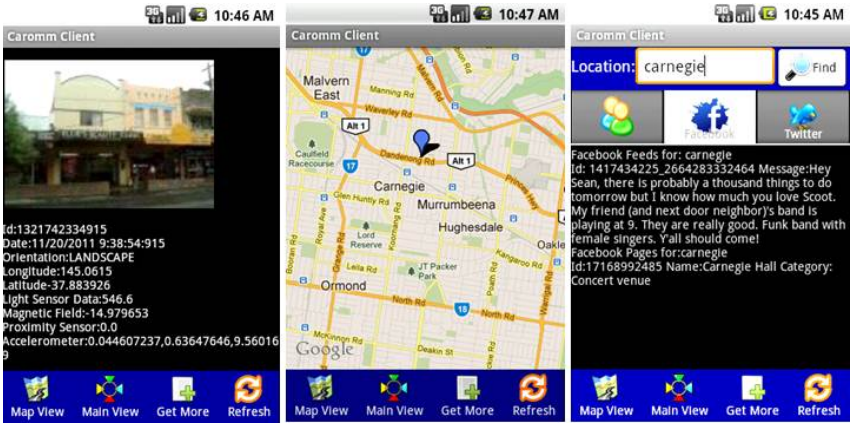

Fig. 7. Screenshots of CAROMM Client Querying the Cloud (Web service)

collection module to have a balance between these two factors ensuring high data accuracy while reducing battery and bandwidth usage. In this section, we evaluate and validate the data collection module's ability to perform on-the-move mining resulting in reduced battery and bandwidth usage, at the same time maintain a high level of accuracy. To this end, we have performed elaborate experiments involving real-time data collection from users over a period of several days under varying environmental conditions to validate the cost-efficiency of the proposed data collection module. We conducted experiments over 5 days involving 5 devices and 5 users. The devices comprised of two Acer Iconia tablets, two Google Nexus S smartphones and one Samsung Galaxy Tab 7.1. The devices were all running the same version of the data collection client compiled using android SDK v2.2. The results of the experimental evaluations are presented in two parts. In the first part, we present the results of experiments observing battery and bandwidth usage. In the second part, we present the results of the data accuracy using the proposed data collection module. The results of the proposed data collection module (we use the term *CAROMM data collection*) is compared against a continuous data collection approach (we use the term *raw data collection* to represent data collection on mobile devices without any device-based processing). The data collected from both approaches are uploaded to the cloud at consecutive intervals.

1) Data Transmission and Energy Usage: The results presented in this section is computed from data collected over a period of 5 days. Table I presents the parameters used for the experiments. These experiments were performed on identical devices, and heterogeneous devices, i.e., the device performing raw collection and device running CAROMM data collection were not identical. In most cases, one of the devices was a tablet and the other was a mobile phone.

Figures 8, 9 and 10 present the results of our experiments comparing CAROMM approach to the raw data collection approach. The results presented here are outcomes of 4 experiments conducted over 5 days. For each trial, we repeated the experiment two times and the outcomes presented are the average of those experiments. Experiments 1 and 2 were

TABLE I
CAROMM EXPERIMENT PARAMETERS

Parameter	Raw Data Collection	CAROMM Data Collection
Duration	2 and 3 Hours	2 and 3 Hours
Sense Interval	30 seconds	30 seconds
Upload Interval	1 minute	15 minute (when no change in cluster is detected)
Cluster Threshold	NA	12000

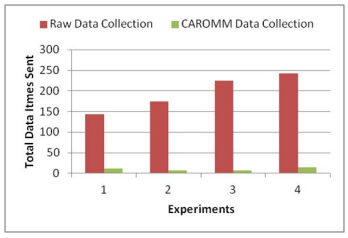

Fig. 8. Data Items Sent (Raw vs. CAROMM)

conducted for a duration of 2 hours on different mobile devices (a tablet and a mobile phone) and experiment 3 and 4 were conducted for a duration of 3 hours on identical mobile devices (two Google Nexus S smartphones). Each experiment was performed under varying real-time environment conditions including poor network conditions, user movement, device usage and same/different mobile devices. We chose different conditions to observe and validate the performance of the proposed data collection module within the CAROMM framework. Due to different conditions, we observe different amount of data collected and battery usage among experiments 1 and 2 (2 hour run) and experiments 3 and 4 (3 hour run). The cluster granularity threshold of 12000 was determined to be most suitable by experimental trials with good sensitivity to change.

Figure 8 presents the number of data items sent by the mobile device. Figure 9 presents the total amount of data sent (in bytes) from the device to the cloud and Figure 10 presents the battery consumption. The battery consumption is presented as a percentage of battery drain from the start to the end of the experiment. We note, for experiments 3 and 4, the mobile devices were also used by the user to perform other activities

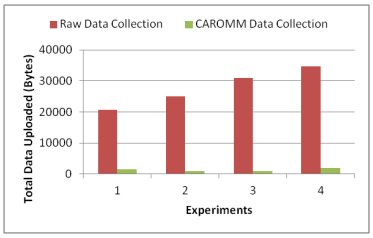

Fig. 9. Total Data Uploaded (Raw vs. CAROMM)

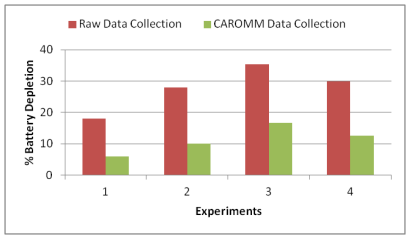

Fig. 10. Battery Depletion (Raw vs. CAROMM)

like browsing, listening to music etc. Hence, to avoid skewing of battery depletion, we normalised the results for experiment 3 and 4 by assigning weights to each experiment depending on device usage by the user. The data size computation was within the application and hence was not affected by other data-intensive activities. The weights were assigned in the range of 1 to 5 with 1 representing no user usage while 5 representing very high user usage. The value 1 to 5 was chosen by questioning the user participating in the trial and based on experimental trials where the device was only uploading data to the cloud. In experiments 3 and 4, for CAROMM we used a weight of 3 and for the raw approach we used a weight of 1.5

The results clearly validate the significant gain in energy and reduced bandwidth usage using the proposed CAROMM approach. We compute *Energy Gain* for CAROMM from Equation 15 of the cost model defined in Section IV-A. We adapt the equation to represent energy drain cost as the % of battery depletion, as follows:

$$Energy\ Gain = \frac{Cost_{ed}^{raw}}{Cost_{ed}^{clust}}$$
$$= \frac{\%\ battery\ depletion\ using\ raw}{\%\ battery\ depletion\ using\ clustering} * 100$$

This computation showed that the reduction in battery usage is 3 times the raw data collection approach, *Energy Gain* is 300%. This is a significant savings in battery usage achieved by the proposed data collection module. Similarly, we use Equation 3 to experimentally evaluate the *Bandwidth Gain*.

$$Bandwidth\ Gain = \frac{Cost_{dt^{raw}}}{Cost_{dt^{clust}}}$$
$$= \frac{(total\ data\ transferred\ from\ M\ to\ C)^{raw}}{(total\ data\ transferred\ from\ M\ to\ C)^{clust}}$$

This computation showed that on average, the *Bandwidth Gain* is 17 times that of the raw approach. This is due to lower size and number of uploads using clustering in CAROMM data collection.

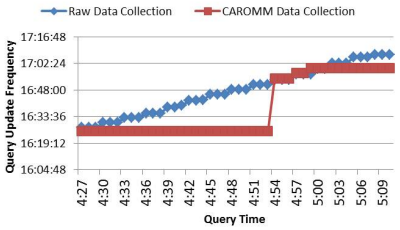

Fig. 11. Difference in Sensed Data (Longitude) - Raw vs. CAROMM

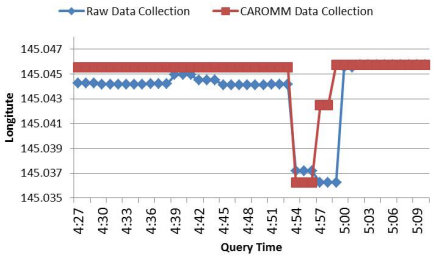

Fig. 12. Difference in Sensed Data (Longitude) - Raw vs. CAROMM

2) Data Accuracy: In the previous section, we experimentally evaluated and validated the energy efficiency and the reduced bandwidth usage of CAROMM data collection module primarily attributed to lesser number of data uploads. Another important evaluation is to validate the proposed data collection approach's ability to maintain high level of accuracy while reducing battery and bandwidth usage. To this end, we performed experiments by executing queries in real-time over the raw and the CAROMM datasets. The queries were issued while the data was being collected using the raw and the CAROMM approaches. The queries were issued every 3 minutes over a period of 40 minutes. At each interval, three queries were executed on the raw and the CAROMM datasets. In all our experiments, we assume the raw data represents the most accurate value of the phenomenon being sensed. We note that the clustering performed by CAROMM was over multiple dimensions including latitude, longitude, accelerometer and light sensor. If any of these sensor values change significantly, the data collection module detects this change. Due to space constraints, we restrict our results to longitude only. To perform our comparison, we analyse the result of the queries by computing the difference in timestamp and the actual data of the query response between the raw and CAROMM datasets. The results of two independent experiments are presented in Figures 11 and 12.

The results presented in Figures 11 and 12 show the plot of timestamps and actual longitude data returned for a query made at a given time. We made some interesting observations during our experiments. We note that the longitude and latitude values reported by devices in the same location had a minor difference. This is well observed in Figure 12 between times 4.27 and 4.51. we see that the timestamp of the query response from CAROMM has not changed while the raw query response has changed. This is due to the fact that there is no significant change in the longitude value during these time periods. When a change is detected at time 4:54, we note that the timestamps

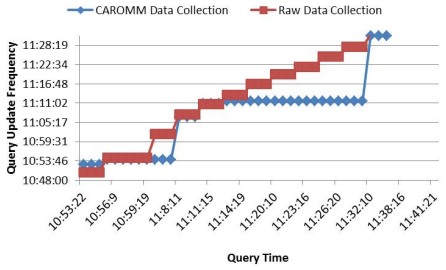

Fig. 13. Timestamps of query result (Raw vs. CAROMM)

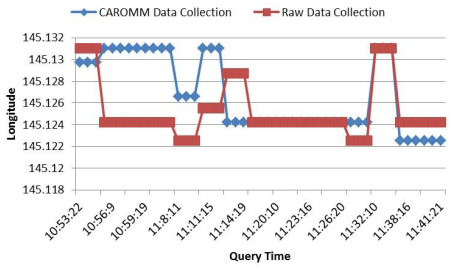

Fig. 14. Difference in Sensed Data (Longitude) - Raw vs. CAROMM

in both CAROMM and raw data are same. The results in Figures 11 and 12 validate the fact that our proposed CAROMM data collection module significantly reduces communication bandwidth without losing data accuracy. The results presented in Figures 11 and 12 are outcomes of experiments where the user movement was moderate over time (note time 4:54 - 5:00). Further validation of the ability of CAROMM data collection module to detect change and thereby maintain high levels of data accuracy is demonstrated in results presented in Figures 13 and 14. Figures 13 and 14 present the results of the experiments where the user was changing locations frequently. As stated earlier and again observed in the results, there is a small difference in the sensed longitude value due to GPS error. The most interesting observation is the ability of the CAROMM data collection module to detect changes indicated at time intervals 11:8 - 11:17 and 11:29 - 11:41. This change detection further validates CAROMM's ability to work efficiently without loss of data accuracy.

Finally, to validate the outcomes of our experiments statistically, we chose F-Test to determine if there is any significant change in longitude computed by raw and CAROMM data collection approaches. The F-Test is designed to compare if two population variances are equal [12]. We used a confidence interval of 95%. In both experiments, the null hypothesis was accepted with a p value of 0.05 concluding that, statistically, the variance in the observed datasets is not significantly different. This further validates the performance of CAROMM data collection module's ability to maintain a high level of data accuracy while consuming less battery and bandwidth.

VI. Conclusions and Future Directions

We have presented our CAROMM system to support mobile crowdsensing and focused on developing an efficient and scalable data collection model that aims to reduce energy and bandwidth consumption related to continuous sensing and uploading in such applications. We implemented and evaluated our system for a location-based information service application. Our evaluation demonstrated that our mobile data mining approach is able to achieve 300% reduction in energy usage and 17 times reduction in bandwidth usage with the same level of data accuracy as traditional sensing-uploading techniques.

The work presented in this paper is the first step and an important component of the overall CAROMM framework. We now intend to enhance our work by investigating cloud data management approaches for mobile crowdsensing. This includes data analysis and query processing on the cloud. Further, we plan to extend our work by incorporating mobile activity recognition using sensor data on mobile devices to understand the context in which the sensing is occurring. Finally, we aim to investigate and address privacy issues surrounding participatory and opportunistic data sensing applications.

References

[1] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 140–150, sept. 2010.

[2] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[3] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, , and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proceedings of SenSys 2008*. New York, NY, USA: ACM, 2008.

[4] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social software," *IEEE Pervasive Computing*, vol. 4, no. 2, 2005.

[5] A. Beach, M. Gartrell, S. Akkala, J. Elston, J. Kelley, K. Nishimoto, B. Ray, S. Razgulin, K. Sundaresa, B. Surendar, M. Terada, , and R. Han, "Whozthat? evolving an ecosystem for context-aware mobile social networks," *IEEE Network*, vol. 22, no. 4, pp. 50–55, 2008.

[6] A. Beach, M. Gartrell, X. Xing, R. Han, Q. Lv, S. Mishra, and K. Seada, "Fusing mobile, sensor, and social data to fully enable context-aware computing," in *Proceedings of HotMobile 2010*. ACM, 2010.

[7] M. M. Gaber, S. Krishnaswamy, and A. B. Zaslavsky, "Cost-efficient mining techniques for data streams." in *Proceedings of ACSW Frontiers'04*, 2004, pp. 109–114.

[8] "Citysense," available from: http://www.sensenetworks.com/citysense.php.

[9] Y. Wang, J. Lin, M. Annavaram, Q. Jacobson, J. I. Hong, B. Krishnamachari, and N. M. Sadeh, "A framework of energy efficient mobile sensing for automatic user state recognition." in *Proceedings of MobiSys'09*, 2009, pp. 179–192.

[10] M. Musolesi, M. Piraccini, K. Fodor, A. Corradi, and A. Campbell, "Supporting energy-efficient uploading strategies for continuous sensing applications on mobile phones," in *Pervasive Computing*, ser. Lecture Notes in Computer Science, P. Floren, A. Krger, and M. Spasojevic, Eds. Springer Berlin Heidelberg, 2010, vol. 6030, pp. 355–372.

[11] S. Krishnaswamy, M. Gaber, M. Harbach, C. Hugues, A. Sinha, B. Gillick, P. Haghighi, and A. Zaslavsky, "Open mobile miner: a toolkit for mobile data stream mining," in *Proceedings of ACM KDD?09*, 2009, pp. 109–114.

[12] "Stat: F-test," available from: http://people.richland.edu/james/lecture/m170/ch13-f.html. Accessed on Dec 2011.