Chapter 14

INTEGRATING SENSORS AND SOCIAL NETWORKS

Charu C. Aggarwal
IBM T. J. Watson Research Center
Hawthorne, NY 10532, USA
charu@us.ibm.com

Tarek Abdelzaher
University of Illinois at Urbana Champaign
Urbana, IL, USA
zaher@cs.uiuc.edu

Abstract A number of sensor applications in recent years collect data which can be directly associated with human interactions. Some examples of such applications include GPS applications on mobile devices, accelerometers, or location sensors designed to track human and vehicular traffic. Such data lends itself to a variety of rich applications in which one can use the sensor data in order to model the underlying relationships and interactions. It also leads to a number of challenges, since such data may often be private, and it is important to be able to perform the mining process without violating the privacy of the users. In this chapter, we provide a broad survey of the work in this important and rapidly emerging field. We also discuss the key problems which arise in the context of this important field and the corresponding solutions.

Keywords: Sensor Networks, Social Sensors

1. Introduction

Social networks have become extremely popular in recent years, because of numerous online social networks such as *Facebook*, *LinkedIn* and *MySpace*.

In addition, many chat applications can also be modeled as social networks. Social networks provide a rich and flexible platform for performing the mining process with different kinds of data such as text, images, audio and video. Therefore, a tremendous amount of research has been performed in recent years on mining such data in the context of social networks [25, 40, 43, 56]. In particular, it has been observed that the use of a combination of linkage structure and different kinds of data can be a very powerful tool for mining purposes [65, 68]. The work in [65, 68] discusses how one can combine the text in social networks with the linkage structure in order to implement more effective classification models. Other recent work [38] uses the linkage structure in image data in order to perform more effective mining and search in information networks. Therefore, it is natural to explore whether sensor data processing can be tightly integrated with social network construction and analysis. Most of the afore-mentioned data types on a social network are static and change slowly over time. On the other hand, sensors collect vast amounts of data which need to be stored and processed in real time. There are a couple of important drivers for integrating sensor and social networks:

- One driver for integrating sensors and social networks is to allow the actors in the social network to both publish their data and subscribe to each other's data either directly, or indirectly after discovery of useful information from such data. The idea is that such collaborative sharing on a social network can increase real-time awareness of different users about each other, and provide unprecedented information and understanding about global behavior of different actors in the social network. The vision of integrating sensor processing with the real world was first proposed in [67].

- A second driver for integrating sensors and social networks is to better understand or measure the aggregate behavior of self-selected communities or the external environment in which these communities function. Examples may include understanding traffic conditions in a city, understanding environmental pollution levels, or measuring obesity trends. Sensors in the possession of large numbers of individuals enable exploiting the crowd for massively distributed data collection and processing. Recent literature reports on several efforts that exploit individuals for data collection and processing purposes such as collection of vehicular GPS trajectories as a way for developing street maps [34], collectively locating items of interest using cell-phone reports, such as mapping speed traps using the Trapster application [76], use of massive human input to translate documents [55], and the development of protein folding games that use competition among players to implement the equivalent of global optimization algorithms [14].

The above trends are enabled by the emergence of large-scale data collection opportunities, brought about by the proliferation of sensing devices of every-day use such as cell-phones, piedometers, smart energy meters, fuel consumption sensors (standardized in modern vehicles), and GPS navigators. The proliferation of many sensors in the possession of the common individual creates an unprecedented potential for building services that leverage massive amounts data collected from willing participants, or involving such participants as elements of distributed computing applications. Social networks, in a sensor-rich world, have become inherently multi-modal data sources, because of the richness of the data collection process in the context of the network structure. In recent years, sensor data collection techniques and services have been integrated into many kinds of social networks. These services have caused a computational paradigm shift, known as *crowd-sourcing* [15], referring to the involvement of the general population in data collection and processing. Crowd-sourcing, arguably pioneered by programs such as SETI, has become remarkably successful recently due to increased networking, mobile connectivity and geo-tagging [1]. Some examples of integration of social and sensor networks are as follows:

- The *Google Latitude* application [69] collects mobile position data of uses, and shares this data among different users. The sharing of such data among users can lead to significant events of interest. For example, proximity alerts may be triggered when two linked users are within geographical proximity of one another. This may itself trigger changes in the user-behavior patterns, and therefore the underlying sensor values. This is generally true of many applications, the data on one sensor can influence data in the other sensors.

- The *City Sense application* [70] collects sensor data extracted from fixed sensors, GPS-enabled cell phones and cabs in order to determine where the people are, and then carries this information to clients who subscribe to this information. The information can also be delivered to clients with mobile devices. This kind of social networking application provides a "sense" as to where the people in the city are, and can be used in order to effectively plan activities. A similar project, referred to as *WikiCity*, [13] developed at MIT, uses the mobile data collected from cell phones in order to determine the spatial trends in a city, and which the social streets might be.

- This general approach of collecting individual location data from mobile phones can also be used in order to generate interesting business decisions. For example, the project *MacroSense* [73] analyzes customer location behaviors, in order to determine individuals which behave in a similar way to a given target. The application is able to perform real time recommendations, personalization and discovery from real time location data.

- *Automotive Tracking Application:* A number of real-time automotive tracking applications determine the important points of congestion in the city by pooling GPS data from the vehicles in the city. This can be used by other drivers in order to avoid points of congestion in the city. In many applications, such objects may have implicit links among them. For example, in a military application, the different vehicles may have links depending upon their unit membership or other related data.

  Another related application is that of sharing of bike track paths by different users [53]. The problem of finding bike routes is naturally a trial-and-error process in terms of finding paths which are safe and enjoyable. The work in [53] designs *Biketastic*, which uses GPS-based sensing on a mobile phone application in order to create a platform which enables rich sharing of biker experiences with one another. The microphone and the accelerometer embedded on the phone are sampled to infer route noise level and roughness. The speed can also be inferred directly from the GPS sensing abilities of the mobile phone. The platform combines this rich sensor data with mapping and visualization in order to provide an intuitive and visual interface for sharing information about the bike routes.

- *Animal Tracking:* In its most general interpretation, an actor in a social network need not necessary be a person, but can be any living entity such as an animal. Recently, animal tracking data is collected with the use of radio-frequency identifiers. A number of social links may exist between the different animals such as group membership, or family membership. It is extremely useful to utilize the sensor information in order to predict linkage information and vice-versa. A recent project called *MoveBank* [71] has made tremendous advances in collecting such data sets. We note that a similar approach may be used for commercial product-tracking applications, though social networking applications are generally relevant to living entities, which are most typically people.

Social sensors provide numerous research challenges from the perspective of analysis.

- Since the collected data typically contains sensitive personal data (eg. location data), it is extremely important to use privacy-sensitive techniques [28, 52] in order to perform the analysis. A recent technique called *PoolView* [28] designs privacy-sensitive techniques for collecting and using mobile sensor data.

- The volume of data collected can be very large. For example, in a mobile application, one may track the location information of millions of users simultaneously. Therefore, it is useful to be able to design techniques which can compress and efficiently process the large amounts of collected data.
- Many of the applications require *dynamic* and *real time* responses. For example, applications which trigger alerts are typically time-sensitive and the responses may be real-time. The real-time challenges of such applications are quite challenging, considering the large number of sensors which are tracked at a given time.

This chapter is organized as follows. Section 2 briefly discusses some key technological advances which have occurred in recent years, which have enabled the design of such dynamic and embedded applications. Section 3 introduces techniques for social network modeling from dynamic links which are naturally created by the sensor-based scenario. Section 4 discusses a broad overview of the key system design questions which arise in these different contexts. One of the important issues discussed in this section is privacy, which is discussed in even greater detail in a later section. Section 5 discusses the database design issues which are essential for scalability. Section 6 discusses some important privacy issues which arise in the context of social networks with embedded sensors. Section 7 introduces some of the key applications associated with integrated social and sensor networks. Section 8 discusses the conclusions and research directions.

2. Sensors and Social Networks: Technological Enablers

A number of recent technological advances in hardware and software have enabled the integration of sensors and social networks. One such key technological advance is the development is small mobile sensors which can collect a variety of user-specific information such as audio or video. Many of the applications discussed are based on *user-location*. Such location can easily be computed with the use of mobile GPS-enabled devices. For example, most of the recent smart-phones typically have such GPS technology embedded inside them. Some examples of such mobile sensor devices may be found in [45, 42].

Sensors typically collect large amounts of data, which must be continuously stored and processed. Furthermore, since the number of users in a social network can be very large, this leads to natural scalability challenges for the storage and processing of the underlying streams. For example, many naive solutions such as the centralized storage and processing of the raw streams are not very practical, because of the large number of streams which are continuously received. In order to deal with this issue, a number of recent hardware and software advances have turned out to be very useful.

- *Development of Fast Stream Processing Platforms:* A number of fast stream processing platforms, such as the IBM System S platform [72] have been developed in recent years, which are capable of storing and processing large volumes of streams in real time. This is a very useful capability from the perspective of typical cyber-physical applications which need a high level of scalability for real-time processing.

- *Development of Stream Synopsis Algorithms and Software:* Since the volume of the data collected is very large, it often cannot be collected explicitly. This leads to the need for designing algorithms and methods for stream synopsis construction [2]. A detailed discussion of a variety of methods (such as sketches, wavelets and histograms) which are used for stream synopsis construction and analysis is provided in [2].

- *Increased Bandwidth:* Since sensor transmission often requires large wireless bandwidth, especially when the data is in the form of audio or video streams, it is critical to be able to transmit large amounts of data in real time. The increases in available bandwidth in recent years, have made such real time applications a reality.

- *Increased Storage:* In spite of the recently designed techniques for compressing the data, the storage challenges for stream processing continue to be a challenge. Recent years have seen tremendous advances in hardware, which allow much greater storage, than was previously possible.

In addition, the development of miniaturized sensors and batteries have allowed their use and deployment in a number of different social settings. For example, the development of miniaturized sensors, which can be embedded within individual attire can be helpful in a wide variety of scenarios [42, 31, 18, 19]. For example, the spec mote is an extremely small sensor device, which can be embedded in the clothing of a user, while remaining quite unobtrusive.

In addition, the sensing abilities of such devices have also increased considerably in recent years. For example, the *sociometer* [18, 19] is a small wearable device, which can measure the following kinds of interactions:

- Detection of people nearby

- Motion information and accelerometers

- Microphone for speech information

In addition, the device has the flexibility to allow for the addition of other kinds of sensors such as GPS sensors and light sensors. The ability to pack larger and larger amounts of functionality into small and unobtrusive devices has been a recent innovation, which has encouraged the development of an ever-increasing number of social-centered applications. The aim of collecting a large number of such interactive behaviors is to be able to effectively model interactions, between different users, and then model the dynamics of the interaction with the use of the collected information.

3. Dynamic Modeling of Social Networks

In the case of an explicitly linked social network, the relationships between different entities are quite clear, and therefore the dynamics of the interaction can be modeled relatively easily. However, in the case of a sensor network, the links between different entities may or may not be available depending upon the application. For example, the *Google Latitude* application allows for explicit links between different agents. On the other hand, in many social applications [19], the links and communities between different agents may need to be derived based on their location and behavior. In such cases, the structure of the social network itself and the underlying communities [20, 21] can be derived directly from the details of the underlying interaction. This is a challenging problem, especially when the number of agents are large, and the number of interactions between them is even larger and dynamically evolving.

Many sensing platforms such as those discussed in [18], yield sensor data which is varied, and is of a multi-modal nature. For example, the data could contain information about interactions, speech or location. It is useful to be able analyze such data in order to summarize the interactions and make inferences about the underlying interactions. Such multi-modal data can also be leveraged in order to make predictions about the nature of the underlying activities and the corresponding social interactions. This also provides a virtual technique to perform link inferences in the underlying network.

For example, the collection of activity sensing data is not very useful, unless it can be leveraged to summarize the nature of the activities among the different participants. For example,in the case of the techniques discussed in [19], the IR transceiver is used to determine which people are in proximity of one another. However, this cannot necessarily be used in order to determine whether the corresponding people are interacting with another. A knowledge of such interactions can be determined with the use of *speech segmentation* techniques in which it is determined which participants are interacting with one another. The speech portions are segmented out of the ambient noise, and then segmented into conversations. The knowledge of such face-to-face interactions can be used to build dynamic and virtual links among the different participants.

We note that a dynamically linked social network can be modeled in two different ways:

- The network can be modeled as a group of dynamic interacting agents. The stochastic properties of these agents can be captured with the use of hidden markov models in order to characterize various kinds of behaviors. This is the approach used for community modeling as discussed in [12, 21].

- The interactions of the participants can be modeled as links which are continuously created or destroyed depending upon the nature of the underlying interactions. as a graph stream, in which the nodes represent the participants, and the edges represent the interactions among these different participants. Recently, a number of analytical techniques have been designed in order to determine useful knowledge-based patterns in graph streams [4]. These include methods for dynamically determining shortest-paths, connectivity, communities or other topological characteristics of the underlying network.

The inherently dynamic nature of such interactions in an evolving and dynamic social network leads to a number of interesting challenges from the perspective of social network analysis. Some examples of such challenges are discussed below.
(1) Determination of dynamic communities in graph streams: Communities are defined as dense regions of the social network in which the participants frequently interact with one another over time. Such communities in a dynamically evolving social network can be determined by using agent-based stochastic analysis or link-based graph stream analysis. Methods for modeling such a social network as a group of dynamically evolving agents are discussed in [12, 21]. In these techniques, a hidden markov model is used in conjunction with an influence matrix in order to model the evolving social network.

A second approach is to model the underlying face-to-face interactions as dynamic links. This creates an inherently dynamic network scenario in which the structure of the communities may continuously evolve over time. Therefore, a key challenge is to determine such communities in dynamic networks, when the clustering patterns may change significantly over time. Methods for determining evolving clusters and communities in networks have been discussed in [6, 7, 17, 57]. Graph streams pose a special challenge because of the rapid nature of the incoming edges, and their use for determination of evolving communities.
(2) Mining Structural Patterns in Time-Evolving Social Networks: Aside from the common problem of community detection, another interesting problem is that of mining structural patterns of different kinds in time evolving graphs. Some common methods for finding such patterns typically use matrix and tensor-based tools, which are comprehensively described in a tutorial in [27]. Common problems in time-evolving graphs include those of frequent pattern determination, outlier detection, proximity tracking [58], and subgraph change detection [46].

(3) Modeling spatio-temporal dynamics: Many of the approaches discussed above model the dynamics of the interactions as dynamic links. While this provides greater generality, it does not capture the spatio-temporal nature of the underlying agents. For example, the data received in a GPS application often contains spatio-temporal information such as the positions of different agents, and their underlying interactions. Therefore, an interesting and important challenge is to model the aggregate spatio-temporal dynamics in order to determine the underlying patterns and clusters. Such spatio-temporal dynamics can be used in order to make interesting *spatial predictions* such as future regions of activity or congestions. For example, methods for determining clusters and communities from such mobile applications have been discussed in [44].

Ad discussed earlier, the determination of dynamic interactions requires the real-time modeling of face-to-face interactions, which can sometimes be sensitive information. This also leads to numerous privacy challenges, especially since the interactions between the participants may be considered personal information. A number of privacy-sensitive approaches for face-to-face activity modeling and conversation segmentation have been discussed in [61–64]. We will discuss more details on privacy-issues in a later section of this chapter.

4. System Design and Architectural Challenges

The aforementioned monitoring and social computing opportunities present a need for a new architecture that encourages data sharing and efficiently utilizes data contributed by users. The architecture should allow individuals, organizations, research institutions, and policy makers to deploy applications that monitor, investigate, or clarify aspects of *socio-physical* phenomena; processes that interact with the physical world, whose state depends on the behavior of humans in the loop.

An architecture for social data collection should facilitate distillation of concise actionable information from significant amounts of raw data contributed by a variety of sources, to inform high-level user decisions. Such an architecture would typically consist of components that support (i) privacy-preserving sensor data collection, (ii) data model construction, and (iii) real-time decision services. For example, in an application that helps drivers improve their vehicular fuel-efficiency, data collection might involve upload of fuel consumption data and context from the vehicle's on-board diagnostics (OBD-II) interface and related sensors; a model might relate the total fuel consumption for a vehicle on a road segment as a function of readily available parameters (such as average road speed, degree of congestion, incline, and vehicle weight); the decision support service might provide navigation assistance to find the most fuel-efficient route to a given destination (as opposed to a fastest or shortest route). Below, we elaborate on the above functions.

4.1 Privacy-preserving data collection

In a grassroots application that is not managed by a globally trusted authority, an interesting challenge becomes ensuring the privacy of data shared. Anonymity is not a sufficient solution because the data themselves (such as GPS traces) may reveal the identity of the owner even if shared anonymously. One interesting direction is to allow individuals to “lie" about their data in a way that protects their privacy, but without degrading application quality. For example, in a traffic speed monitoring application reconstruction of community statistics of interest (such as average traffic speed on different streets) should remain accurate, despite use of perturbed data (“lies" about actual speed of individual vehicles) as input to the reconstruction process. This is possible thanks to deconvolution techniques that recover the statistical distribution of the original signals, given the statistical distribution of perturbed data and the statistical distribution of noise. Solutions to this and related problems can be found in literature on privacy-preserving statistics [5]. Recently, special emphasis was given to perturbing time-series data [28], since sensor data typically comprise a correlated series of samples of some continuous phenomenon. Perturbing time-series data is challenging because correlations among nearby samples can be exploited to breach privacy. Recent results demonstrate that the frequency spectrum of the perturbation signal must substantially overlap with the frequency spectrum of the original data time-series for the latter to be effectively concealed [28]. Generalizations to perturbation of correlated multi-dimensional time-series data were proposed in [52]. The main challenge addressed in this work was to account for the fact that data shared by different sensors are usually not independent. For example, temperature and location data can be correlated, allowing an attacker to make inferences that breach privacy by exploiting cross-sensor correlations.

A related interesting problem is that of perturbation (i.e., noise) energy allocation. Given a perturbation signal of a particular energy budget (dictated perhaps by reconstruction accuracy requirements), how to allocate this energy budget across the frequency spectrum to optimally conceal an original data signal? A recent technique defines privacy as the amount of mutual information between the original and perturbed signals. Optimality is defined as perturbation that minimizes the upper bound on such (leaked) mutual information. The technique describes how optimal perturbation is computed, and demonstrates the fundamental trade-off between the bound on information leak (privacy) and the bound on reconstruction accuracy [51].

4.2 Generalized Model Construction

Many initial participatory sensing applications, such as those giving rise to the above privacy concerns, were concerned with computing community statistics out of individual private measurements. The approach inherently assumes richly-sampled, low-dimensional data, where many low-dimensional measurements (e.g., measurements of velocity) are redundantly obtained by individuals measuring the same variable (e.g., speed of traffic on the same street). Only then can good statistics be computed. Many systems, however, do not adhere to the above model. Instead, data are often high-dimensional, and hence sampling of the high-dimensional space is often sparse. The more interesting question becomes how to generalize from high-dimensional, sparsely-sampled data to cover the entire input data space? For instance, consider a fuel-efficient navigation example, where it is desired to compute the most fuel-efficient route between arbitrary source and destination points, for an arbitrary vehicle and driver. What are the most important generalizable predictors of fuel efficiency of current car models driven on modern streets? A large number of predictors may exist that pertain to parameters of the cars, the streets and the drivers. These inputs may be static (e.g., car weight and frontal area) or dynamic (e.g., traveled road speed and degree of congestion). In many cases, the space is only sparsely sampled, especially in conditions of sparse deployment of the participatory sensing service. It is very difficult to predict *a priori* which parameters will be more telling. More importantly, the key predictors might differ depending on other parameters. For example, it could be that the key predictors of fuel efficiency for hybrid cars and gas-fueled cars are different. It is the responsibility of the model construction services to offer not only a general mechanism for applications to build good models quickly from the data collected, but also a mechanism for identifying the scope within which different predictors are dominant. A single “one-side-fits-all" prediction model, computed from all available data, is not going to be accurate. Similarly computing a model for each special case (e.g., a model for each type of car) is not going to be useful because, as stated above, the sampling is sparse. Hence, it is key to be able to generalize from experiences of some types of vehicles to predictions of others. Recent work combined data mining techniques based on regression cubes and sampling cubes to address the model generalization problem for sparse, high-dimensional data [32].

4.3 Real-time Decision Services

Ultimately, a generalized model, such as that described above, may be used as an input to an application-specific optimization algorithm that outputs some *decisions* for users in response to user queries. For example, estimates of fuel consumption on different roads on a map can be input to Dijkstra’s algorithm to find the minimum fuel route between points specified by the user. This route constitutes a decision output. Hence, support for real-time stream processing and decision updates must be provided as part of the social sensing architecture.

A key property of real-time decision services is the involvement of humans in the loop. A significant challenge is therefore to design appropriate user interfaces. End-user devices will act as data custodians who collect, store, and share user data. The level at which these custodians interact with the user, as well as the nature of interactions, pose significant research problems with respect to minimizing inconvenience to the user while engaging the user appropriately. Context sensing, collaborative learning, persuasion, and modeling of socio-sensing systems (with humans in the loop) become important problems. Participation incentives, role assignment, and engagement of users in modeling and network learning become important application design criteria that motivate fundamental research on game theoretic, statistical, machine learning, and economic paradigms for application design.

4.4 Recruitment Issues

The quality of the social experience gained from a sensor-based framework is dependent on the ability to recruit high quality participants for sensor collection and sharing. The work in [54] observes that the process of recruiting volunteers for participatory sensing campaigns is analogous to recruiting volunteers or employees in non-virtual environments. This similarity is used in order to create a 3-stage process for recruitment:

- **Qualifier:** This refers to the fact that the participants must meet minimum requirements such as availability and reputation.

- **Assessment:** Once participants that meet minimum requirements are found, the recruitment system then determines which candidates are most appropriate based on both diversity and coverage.

- **Progress Review:** Once the sensing process starts, the recruitment system must check participants' coverage and data collection reputation to determine if they are consistent with their base profile. This check can occur periodically, and if the similarity of profiles is below a threshold, this is used as a feedback to an additional recruitment process.

4.5 Other Architectural Challenges

Proper design of the above system components gives rise to other important challenges that must be solved in order to enable development and deployment of successful mobile sensing applications that adequately meet user needs. The following relates challenges described in a recent NSF-sponsored report on social sensing [77].

From the application perspective, mobile sensing applications depend significantly on social factors (user adoption, peer pressure, social norms, social networks, etc) as well as the nature of physical phenomena being monitored or controlled. Exciting interdisciplinary research challenges exist in describing the properties of distributed socio-physical applications. For example, what are the dynamics of information propagation in such systems? What are closed-loop properties of interaction involving social and physical phenomena? What are some fundamental bounds on capacity, delivery speed, and evolution of socio-sensing systems? Answering such questions is fundamental to informed design and performance analysis of sensing applications involving crowd-sourcing.

From the underlying physical network perspective, mobile sensing applications herald an era where many network clients are embedded devices. This motivates the investigation of a network architecture, where the main goal from networking shifts from offering a mere communication medium to offering *information distillation services*. These services bridge the gap between myriads of heterogeneous data feeds and the high-level human decision needs. In a network posed as an information service (as opposed to a communication medium), challenges include division of responsibilities between the end-device (e.g., phone) and network; paradigms for data collection on mobile devices, architectural support for data management, search, and mining; scalability to large-scale real-time information input and retrieval; improved context-awareness; support for predictability; and investigation of network and end-system support for reduction of cognitive overload of the information consumer. Other challenges in the design of network protocols for mobile sensing include energy management, integration of network storage, personalized search and retrieval, support for collaborative sensing, and exploitation of a rich realm of options in information transfer modalities and timing, including deferred information sharing and delay-tolerant communication.

While several social sensing applications are already deployed, exciting research opportunities remain in order to help understand their emergent behavior, optimize their performance, redesign the networks on which they run, and provide guarantees to the user, such as those on bounding unwanted information leakage.

5. Database Representation: Issues and Challenges

A number of database challenges naturally arise in such massive real time applications. For example, it is often the case that millions of streams may be collected simultaneously. It is useful to compress, process and store these streams in real-time. Furthermore,

such compression techniques need to be real-time in order to enable effective processing. In this section, we briefly review some stream compression and processing techniques which may be useful for such applications. In recent years a number of *synopsis structures* have been developed, which can be used in conjunction with a variety of mining and query processing techniques [30]. Some key synopsis methods include those of sampling, wavelets, sketches and histograms. The key challenges which arise in the context of synopsis construction of data streams are as follows:

Broad Applicability: The synopsis structure is typically used as an intermediate representation, which should be usable for a variety of analytical scenarios in the context of social networking applications.

One-pass constraint: The one-pass constraint is critical to synopsis construction algorithms, especially when the volume of the incoming data is large. This is especially true for a social networking application in which millions of sensors are simultaneously transmitting data.

Time and Space Efficiency: Since data streams have a very large volume, it is essential to create the synopsis in a time- and space-efficient way. In this sense, some of the probabilistic techniques such as sketches are extremely effective for counting-based applications, since they require constant-space for provable probabilistic accuracy. In other words, the time- and space-efficiency depends only upon the accuracy of the approach rather than the length of the data stream.

Data Stream Evolution: Since the incoming stream patterns may evolve over time, a synopsis structure which is constructed from the overall behavior of the data stream is not quite as effective as one which uses recent history. Consequently, it is often more effective to create synopsis structures which either work with sliding windows, or use some decay-based approach in order to weight the data stream points.

One key characteristic of many of the above methods is that while they work effectively in the 1-dimensional case, they often lose their effectiveness in the multi-dimensional case either because of data sparsity or because of inter-attribute correlations. The multi-dimensional case is especially for the social networking scenario, because millions of streams may be collected and processed at a given time. Next, we will discuss the broad classes of techniques which are used for synopsis construction in data streams. Each of these techniques have their own advantages in different scenarios, and we will take care to provide an overview of the different array of methods which are used for synopsis construction in data streams. The broad techniques which are used for synopsis construction in data streams are as follows:

Reservoir Sampling: Sampling methods are widely used for traditional database applications, and are extremely popular because of their broad applicability across a wide array of tasks in data streams. A further advantage of sampling methods is that unlike many other synopsis construction methods, they maintain their inter-attribute correlations across samples of the data. It is also often possible to use probabilistic inequalities in order to bound the effectiveness of a variety of applications with sampling methods.

However, a key problem in extending sampling methods to the data stream scenario, is that one does not know the total number of data points to be sampled in advance. Rather, one must maintain the sample in a dynamic way over the entire course of the computation. A method called reservoir sampling was first proposed in [59], which maintains such a sample dynamically. This technique was originally proposed in the context of one-pass access of data from magnetic-storage devices. However, the techniques also naturally extend to the data stream scenario.

Let us consider the case, where we wish to obtain an unbiased sample of size n from the data stream. In order to initialize the approach, we simply add the first n points from the stream to the reservoir. Subsequently, when the $(t + 1)$th point is received, it is added to the reservoir with probability $n/(t + 1)$. When the data point is added to the reservoir, it replaces a random point from the reservoir. It can be shown that this simple approach maintains the uniform sampling distribution from the data stream. We note that the uniform sampling approach may not be very effective in cases where the data stream evolves significantly. In such cases, one may either choose to generate the stream sample over a sliding window, or use a decay-based approach in order to bias the sample. An approach for sliding window computation over data streams is discussed in [48].

A second approach [3] uses biased decay functions in order to construct synopsis from data streams. It has been shown in [3] that the problem is extremely difficult for arbitrary decay functions. In such cases, there is no known solution to the problem. However, it is possible to design very simple algorithms for some important classes of decay functions. One of these classes of decay functions is the *exponential decay function*. The exponential decay function is extremely important because of its *memory less property*, which guarantees that the future treatment of a data point is independent of the past data points which have arrived. An interesting result is that by making simple implementation modifications to the algorithm of [59] in terms of modifying the probabilities of insertion and deletion, it is possible to construct a robust algorithm for this problem. It has been shown in [3] that the approach is quite effective in practice, especially when there is significant evolution of the underlying data stream.

While sampling has several advantages in terms of simplicity and preservation of multi-dimensional correlations, it loses its effectiveness in the presence of data sparsity. For example, a query which contains very few data points is unlikely to be accurate with the use of a sampling approach. However, this is a general problem with most techniques which are effective at counting frequent elements, but are not quite as effective at counting rare or distinct elements in the data stream.

Sketches: Sketches use some properties of random sampling in order to perform counting tasks in data streams. Sketches are most useful when the *domain size* of a data stream is very large. In such cases, the number of possible distinct elements become very large, and it is no longer possible to track them in space-constrained scenarios. There are two broad classes of sketches: *projection based* and *hash based*. We will discuss each of them in turn.

Projection based sketches are constructed on the broad idea of random projection [39]. The most well known projection-based sketch is the AMS sketch [10, 11], which we will discuss below. It has been shown in [39], that by by randomly sampling subspaces from multi-dimensional data, it is possible to compute ϵ-accurate projections of the data with high probability. This broad idea can easily be extended to the massive domain case, by viewing each distinct item as a dimension, and the counts on these items as the corresponding values. The main problem is that the vector for performing the projection cannot be maintained explicitly since the length of such a vector would be of the same size as the number of distinct elements. In fact, since the sketch-based method is most relevant in the distinct element scenario, such an approach defeats the purpose of keeping a synopsis structure in the first place.

Let us assume that the random projection is performed using k sketch vectors, and r_i^j represents the jth vector for the ith item in the domain being tracked. In order to achieve the goal of efficient synopsis construction, we store the random vectors implicitly in the form of a seed, and this can be used to dynamically generate the vector. The main idea discussed in [35] is that it is possible to generate random vectors with a seed of size $O(\log(N))$, provided that one is willing to work with the restriction that $r_i^j \in \{-1, +1\}$ should be 4-wise independent. The sketch is computed by adding r_i^j to the jth component of the sketch for the ith item. In the event that the incoming item has frequency f, we add the value $f \cdot r_i^j$. Let us assume that there are a total of k sketch components which are denoted by $(s_1 \dots s_k)$. Some key properties of the pseudo-random number generator approach and the sketch representation are as follows:

- A given component r_i^j can be generated in poly-logarithmic time from the seed. The time for generating the seed is poly-logarithmic in the domain size of the underlying data.

- A variety of approximate aggregate functions on the original data can be computed using the sketches.

Some example of functions which can be computed from the sketch components are as follows:

- **Dot Product of two streams:** If $(s_1 \dots s_k)$ be the sketches from one stream, and $(t_1 \dots t_k)$ be the sketches from the other stream, then $s_j \cdot t_j$ is a random variable whose expected value of the dot product.

- **Second Moment:** If $(s_1 \dots s_k)$ be the sketch components for a data stream, it can be shown that the expected value of s_j^2 is the second moment. Furthermore, by using Chernoff bounds, it can be shown that by selecting the median of $O(\log(1/\delta))$ averages of $O(1/\epsilon^2)$ copies of $s_j \cdot t_j$, it is possible to guarantee the accuracy of the approximation to within $1 \pm \epsilon$ with probability at least $1 - \delta$.

- **Frequent Items:** The frequency of the ith item in the data stream is computed by by multiplying the sketch component s_j by r_i^j. However, this estimation is accurate only for the case of frequent items, since the error is estimation is proportional to the overall frequency of the items in the data stream.

More details of computations which one can perform with the AMS sketch are discussed in [10, 11].

The second kind of sketch which is used for counting is the *count-min* sketch [26]. The count-min sketch is based upon the concept of hashing, and uses $k = \ln(1/\delta)$ pairwise-independent hash functions, which hash onto integers in the range $(0 \dots e/\epsilon)$. For each incoming item, the k hash functions are applied and the frequency count is incremented by 1. In the event that the incoming item has frequency f, then the corresponding frequency count is incremented by f. Note that by hashing an item into the k cells, we are ensuring that we maintain an overestimate on the corresponding frequency. It can be shown that the minimum of these cells provides the ϵ-accurate estimate to the frequency with probability at least $1 - \delta$. It has been shown in [26] that the method can also be naturally extended to other problems such as finding the dot product or the second-order moments. The count-min sketch is typically more effective for problems such as frequency-estimation of individual items than the projection-based AMS sketch. However, the AMS sketch is more effective for problems such as second-moment estimation.

Wavelet Decomposition: Another widely known synopsis representation in data stream computation is that of the wavelet representation. One of the most widely used representations is the *Haar Wavelet*. We will discuss this technique in detail in this section. This technique is particularly simple to implement, and is widely used in the literature for hierarchical decomposition and summarization. The basic idea in the wavelet technique is to create a decomposition of the data characteristics into a set of wavelet functions and basis

Granularity (Order k)	**Averages** Φ values	**DWT Coefficients** ψ values
$k = 4$	(8, 6, 2, 3, 4, 6, 6, 5)	-
$k = 3$	(7, 2.5, 5, 5.5)	(1, -0.5,-1, 0.5)
$k = 2$	(4.75, 5.25)	(2.25, -0.25)
$k = 1$	(5)	(-0.25)

Table 14.1. An Example of Wavelet Coefficient Computation

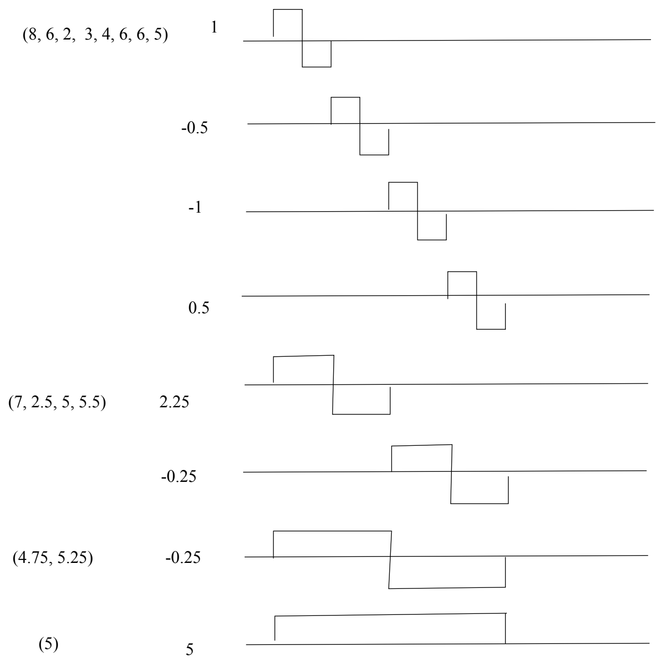

Figure 14.1. Illustration of the Wavelet Decomposition

functions. The property of the wavelet method is that the higher order coefficients of the decomposition illustrate the broad trends in the data, whereas the more localized trends are captured by the lower order coefficients.

We assume for ease in description that the length q of the series is a power of 2. This is without loss of generality, because it is always possible to decompose a series into segments, each of which has a length that is a power of two. The Haar Wavelet decomposition defines 2^{k-1} coefficients of order k. Each of these 2^{k-1} coefficients corresponds to a contiguous portion of the time series of length $q/2^{k-1}$. The ith of these 2^{k-1} coefficients corresponds to the segment in the series starting from position $(i-1) \cdot q/2^{k-1} + 1$ to position $i * q/2^{k-1}$. Let us denote this coefficient by ψ_k^i and the corresponding time series segment by S_k^i. At the same time, let us define the average value of the first half of the S_k^i by a_k^i and the second half by b_k^i. Then, the value of ψ_k^i is given by $(a_k^i - b_k^i)/2$. More formally, if Φ_k^i denote the average value of the S_k^i, then the value of ψ_k^i can be defined recursively as follows:

$$\psi_k^i = (\Phi_{k+1}^{2 \cdot i-1} - \Phi_{k+1}^{2 \cdot i})/2 \tag{14.1}$$

The set of Haar coefficients is defined by the Ψ_k^i coefficients of order 1 to $\log_2(q)$. In addition, the global average Φ_1^1 is required for the purpose of perfect reconstruction. We note that the coefficients of different order provide an understanding of the major trends in the data at a particular level of granularity. For example, the coefficient ψ_k^i is half the quantity by which the first half of the segment S_k^i is larger than the second half of the same segment. Since larger values of k correspond to geometrically reducing segment sizes, one can obtain an understanding of the basic trends at different levels of granularity. We note that this definition of the Haar wavelet makes it very easy to compute by a sequence of averaging and differencing operations. In Table 14.1, we have illustrated how the wavelet coefficients are computed for the case of the sequence $(8, 6, 2, 3, 4, 6, 6, 5)$. This decomposition is illustrated in graphical form in Figure 14.1. We also note that each value can be represented as a sum of $\log_2(8) = 3$ linear decomposition components. In general, the entire decomposition may be represented as a tree of depth 3, which represents the hierarchical decomposition of the entire series. This is also referred to as the *error tree*. In Figure 14.2, we have illustrated the error tree for the wavelet decomposition illustrated in Table 14.1. The nodes in the tree contain the values of the wavelet coefficients, except for a special *super-root* node which contains the series average. This super-root node is not necessary if we are only considering the relative values in the series, or the series values have been normalized so that the average is already zero. We further note that the number of wavelet coefficients in this series is 8, which is also the length of the original series. The original series has been replicated just below the error-tree in Figure 14.2, and it can be reconstructed by adding or subtracting the values in the nodes

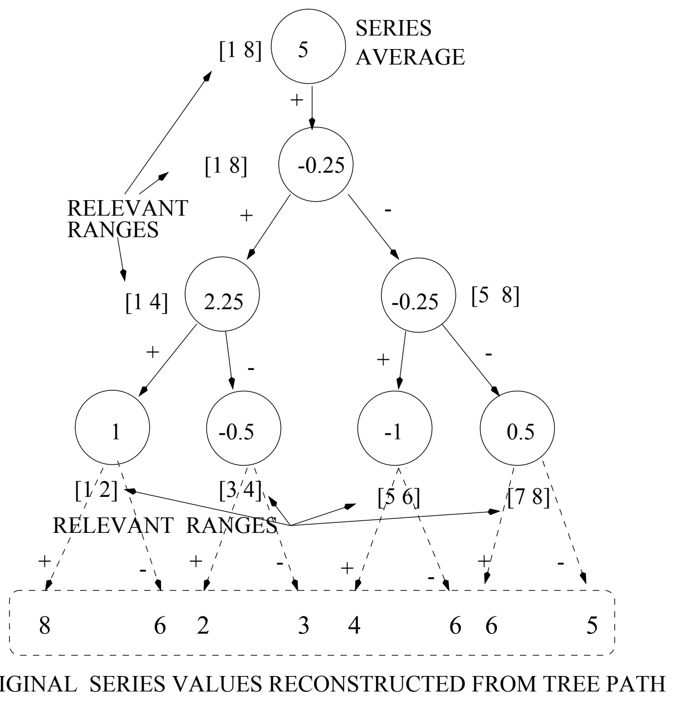

Figure 14.2. The Error Tree from the Wavelet Decomposition

along the path leading to that value. We note that each coefficient in a node should be added, if we use the left branch below it to reach to the series values. Otherwise, it should be subtracted. This natural decomposition means that an entire contiguous range along the series can be reconstructed by using only the portion of the error-tree which is relevant to it. Furthermore, we only need to retain those coefficients whose values are significantly large, and therefore affect the values of the underlying series. In general, we would like to minimize the reconstruction error by retaining only a fixed number of coefficients, as defined by the space constraints.

While wavelet decomposition is easy to perform for multi-dimensional data sets, it is much more challenging for the case of data streams. This is because data streams impose a one-pass constraint on the wavelet construction process. A variety of one-pass algorithms for wavelet construction are discussed in [30]. **Histograms:** The technique of histogram construction is closely related to that of wavelets. In histograms the data is binned into a number of intervals along an attribute. For any given query, the counts from the bins can be utilized for query resolution. A simple representation of the histogram method would simply partition the data into equi-depth or equi-width intervals. The main inaccuracy with the use of histograms is that the distribution of data points within a bucket is not retained, and is therefore assumed to be uniform. This causes inaccuracy because of extrapolation at the query boundaries. A natural choice is to use an equal number of counts in each bucket. This minimizes the error variation across different buckets. However, in the case of data streams, the boundaries to be used for equi-depth histogram construction are not known a-priori. We further note that the design of equi-depth buckets is exactly the problem of quantile estimation, since the equi-depth partitions define the quantiles in the data. Another choice of histogram construction is that of minimizing the variance of frequency variances of different values in the bucket. This ensures that the uniform distribution assumption is approximately held, when extrapolating the frequencies of the buckets at the two ends of a query. Such histograms are referred to as V-optimal histograms. Algorithms for V-optimal histogram construction are proposed in [36, 37]. A more detailed discussion of several algorithms for histogram construction may be found in [2].

6. Privacy Issues

Social sensing offers interesting new challenges pertaining to privacy assurances on data. General research on privacy typically focuses on electronic communication as opposed to ramifications of increasing sensory instrumentation in a socio-physical world. In contrast, traditional embedded systems research typically considers computing systems that interact with physical and engineering artifacts and belong to the same trust domain. A need arises to bridge the gap in privacy research by formulating and solving privacy-motivated research challenges in the emerging social sensing systems, where users interact in the context of social networks with embedded sensing devices in the physical world.

Sharing sensor data creates new opportunities for loss of privacy (and new privacy attacks) that exploit physical-side channels or a priori known information about the physical environment. Research is needed on both privacy *specification* and *enforcement* to put such specification and enforcement on solid analytic foundations, much like specification and enforcement of safety requirements of high-confidence software.

Specification calls for new physical privacy specification interfaces that are easy to understand and use for the non-expert. *Enforcement* calls for two complementary types of privacy mechanisms; (i) *protection mechanisms from involuntary physical exposure*, and (ii) *control of voluntary information sharing*. The former enforce *physical privacy*. They are needed to prevent “side-channel" attacks that exploit physical and spatio-temporal properties, characteristic of embedded sensing systems, to make inferences regarding private information. Control of voluntary information sharing must facilitate privacy-preserving exchange of time-series data. A predominant use of data in social sensing applications is for aggregation purposes such as computing statistical information from many sources. Mathematically-based data perturbation and anonymization schemes are needed to hide user data but allow fusion operations on perturbed or partial data to return correct results to a high degree of approximation.

While privacy-preserving statistics and privacy-preserving data mining are mature fields with a significant amount of prior research, sharing of sensor data offers the additional challenge of dealing with *correlated multi-dimensional time-series data* represented by sensory data streams. Correlations within and across sensor data streams and the spatio-temporal context of data offer new opportunities for privacy attacks. The challenge is to perturb a user's sequence of data values such that (i) the individual data items and their trend (i.e., their changes with time) cannot be estimated without large error, whereas (ii) the distribution of the data aggregation results at any point in time is estimated with high accuracy. For instance, in a health-and-fitness social sensing application, it may be desired to find the average weight loss trend of those on a particular diet or exercise routine as well as the distribution of weight loss as a function of time on the diet. This is to be accomplished without being able to reconstruct any individual's weight and weight trend without significant error.

Examples of data perturbation techniques can be found in [9, 8, 16]. The general idea is to add random noise with a known distribution to the user's data, after which a reconstruction algorithm is used to estimate the distribution of the original data. Early approaches relied on adding independent random noise. These approaches were shown to be inadequate. For example, a special technique based on random matrix theory has been proposed in [24] to recover the user data with high accuracy. Later approaches considered hiding individual data values collected from different private parties, taking into account that data from different individuals may be correlated [23]. However, they do not make assumptions on the model describing the *evolution* of data values from a given party over time, which can be used to jeopardize privacy of data streams. Perturbation techniques must specifically consider the data evolution model to prevent attacks that extract regularities in correlated data such as spectral filtering [24] and Principal Component Analysis (PCA) [23].

In work discussed earlier in this chapter [28], it was shown that privacy of time-series data can be preserved if the noise used to perturb the data is itself generated from a process that approximately models the measured phenomenon. For instance, in the weight watchers example, we may have an intuitive feel for the time scales and ranges of weight evolution when humans gain or lose weight. Hence, a noise model can be constructed that exports realistic-looking parameters for both the direction and time-constant of weight changes. The resulting perturbed stream can be aggregated with that of others in the community. Since the distributions of noise model parameters are statistically known, it is possible to estimate the sum, average and distribution of added noise (of the entire community) as a function of time. Subtracting that known average noise time series from the sum of perturbed community curves will thus yield the true community trend. The distribution of community data at a given time can similarly be estimated (using deconvolution methods) since the distribution of noise (i.e., data from virtual users) is known. The estimate improves with community size.

The approach preserves individual user privacy while allowing accurate reconstruction of community statistics. Several research questions arise that require additional work. For example, what is a good upper bound on the reconstruction error of the data aggregation result as a function of the noise statistics introduced to perturb the individual inputs? What are noise generation techniques that minimize the former error (to achieve accurate aggregation results) while maximizing the noise (for privacy)? How to ensure that data of individual data streams cannot be inferred from the perturbed signal? What are some bounds on minimum error in reconstruction of individual data streams? What noise generation techniques maximize such error for privacy? Privacy challenges further include the investigation of attack models involving corrupt noise models (e.g., ones that attempt to deceive non-expert users into using perturbation techniques that do not achieve adequate privacy protection), malicious clients (e.g., ones that do not follow the correct perturbation schemes or send bogus data), and repeated server queries (e.g., to infer additional information about evolution of client data from incremental differences in query responses). For example, given that it is fundamentally impossible to tell if a user is sharing a properly perturbed version of their real weight or just some random value, what fractions of malicious users can be accommodated without significantly affecting reconstruction accuracy of community statistics? Can damage imposed by a single user be bounded using outlier detection techniques that exclude obviously malicious users? How does the accuracy of outlier detection depend on the scale of allowable perturbation? In general, how to quantify the tradeoff between privacy and robustness to malicious user data? How tolerant is the perturbation scheme to collusion among users that aims to bias community statistics? Importantly, how does the *time-series* nature of data affect answers to the above questions compared to previous solutions to similar problems in other contexts (e.g., in relational databases)?

Furthermore, how can the above perturbation techniques, defense solutions, and bounds be extended to the sharing of multiple correlated data streams, or data streams with related context? For example, consider a social sensing application where users share vehicular GPS data to compute traffic speed statistics in a city. In this case, in order to compute the statistics correctly as a function of time and location, each vehicle's speed must be shared together with its current GPS location and time of day. Perturbing the speed alone does not help privacy if the correct location of the user must be revealed at all times. What is needed is a perturbation and reconstruction technique that allows a user to “lie" about their speed, location, and time of day, altogether, in a manner that makes it impossible to reconstruct their true values, yet allow an aggregation service to average out the added multi-dimensional noise and accurately map the true aggregate traffic speed as a function of actual time and space. This problem is related to the more general concern of privacy-preserving classification [60, 66], except that it is applied to the challenging case of aggregates of time-series data. Preliminary work shows that the problem is solvable. Understanding the relation between multi-dimensional error bounds on reconstruction accuracy and bounds on privacy, together with optimal perturbation algorithms in the sense of minimizing the former while maximizing the latter, remains an open research problem.

7. Sensors and Social Networks: Applications

In this section, we will discuss a number of recent applications which have been designed in the context of sensors and social networks. Many of these applications are related to storage and processing of mobile data which is continuously collected over time. Such mobile data can be used in order to provide real time knowledge of the different users to one another, trigger alerts, provide an understanding of social trends, and enable a variety of other applications. In this section, we will discuss a number of social-centric applications, which have been developed in recent years. These include specific systems which have been designed by companies such as Google, Microsoft, and SenseNetworks, as well as a number of generic applications, which have not yet been fully commercialized.

7.1 The Google Latitude Application

The Google Latitude Application uses GPS data which is collected from Google map users on mobile cell phones. It is also possible to collect more approximate data with the use of cell phone tower location data (in case the mobile phones are not GPS enabled), or with the use of IP addresses of a computer which is logged into the personalized google page called *iGoogle*. The Latitude application enables the creation of *virtual friends*, who are essentially other users that carry the same location-enabled device, or use other devices such as personal computers which can transmit approximate location data such as IP-addresses. A number of other applications which enabled by the Google Latitude master application are as follows:

- **Location Alerts:** The application allows the triggering of alerts when someone is near their latitude friends. The alerts are triggered only when something interesting is being done. This is done on the basis of both time and location. For example, an alert could be triggered when two friends are at a routine place, but an unusual time. Alternatively, it could be triggered when two friends are at a routine time but unusual place.

- **Public Location Badge:** It is possible to post one's location directly on blog or social network. This in turn increases the visibility of one's information to other users of the site.
- **Use with Chat Applications:** The mobile location can also be used in conjunction with the *Google Talk* application which allows users to chat with one another. Users who are chatting with one another can see each other's location with the use the embedded latitude functionality.

It is clear that all of the above techniques change the nature and dynamics of social interactions between users. For example, the triggering of alerts can itself lead to a changed pattern of interaction among the different users. The ability to mine the dynamics of such interactions is a useful and challenging task for a variety of applications.

While *Google Latitude* is perhaps the most well known application, it is by no means the only one. A number of recent applications have been designed which can track mobile devices on the internet through GPS tracking. Some of these applications have been designed purely for the purpose of tracking a device which might be lost, whereas others involve more complex social interactions. Any software and hardware combination which enables this has the potential to be used for social sensing applications. Some examples of such applications are as follows:

- **Navizon Application:** This application [74] uses GPS in order to allow social interactions between people with mobile phones. It allows the tracking of mobile friends, coverage of particular areas, and trails followed by a particular user.
- **iLocalis Application:** This application [75] is currently designed only for particular mobile platforms such as the iPhone, and it allows the tracking of family and friends. In addition, it is also designed for corporate applications in which a group of mobile employees may be tracked using the web. Once friendship links have been set up, the application is capable of sending a message to the friends of a particular user, when they are nearby.

7.2 The Citysense and Macrosense Applications

The Citysense and Macrosense applications both collect real-time data from a variety of GPS-enabled cell phones, cell phone tower triangulation, and GPS-enabled cabs. The two applications share a number of similarities in terms of the underlying methodology, but they have different features which are targeted towards different kinds of audiences. We describe them below:

7.2.1 CitySense Application.

The citysense application is designed for the broad consumer base which carries mobile cell phones. The Citysense application is designed to track important trends in the behavior of people in the city. For example, the application has been deployed in San Francisco, and it can show the busiest spots in the city on a mobile map.

The CitySense application also has a social networking version of a collaborative filtering application. The application stores the personal history of each user, and it can use this personal history in order to determine where other similar users might be. Thus, this can provide recommendations to users about possible places to visit based on their past interests.

A very similar application is the WikiCity project [13] which collects real time information with the use of GPS and mobile devices. These are then used to collect the location patterns of users, and their use in a variety of neighborhoods.

7.2.2 MacroSense Application.

The MacroSense application is similar in terms of the data it collects and kind of functionality it provides; however it is focussed towards the commercial segment in predicting consumer behavior. The application can predict the behavior of customers based on their location profile and behavior. The application can predict what a particular customer may like next. The broad idea is to segment and cluster customers into marketing groups based on their behavior, and use this information in order to make predictions. For example, the popularity of a product with users who are most like the target can be used for predictive purposes. Thus, this approach is somewhat like collaborative filtering, except that it uses the behavior of customers rather than their feedback. The effectiveness of particular behaviors which predict the interests are also used. This analysis can be performed in real time, which provides great value in terms of predictive interactions. The analytics can also be used in order to predict group influences for the behaviors of the underlying subjects.

7.3 Green GPS

Green GPS [32] is a participatory sensing navigation service that allows drivers to find the most fuel-efficient routes for their vehicles between arbitrary end-points. Green GPS relies on data collected by individuals from their vehicles as well as on mathematical models to compute fuel efficient routes.

The most fuel efficient route may depend on the vehicle and may be different from the shortest or fastest route. For example, a fast route that uses a freeway may consume more fuel because fuel consumption increases non-linearly with speed. Similarly, the shortest route that traverses busy city streets may be suboptimal because of downtown traffic.

The service exploits measurements of standard vehicular sensor interfaces that give access to most gauges and engine instrumentation. Vehicles that have been sold in the United States after 1996 are mandatorily equipped with a sensing subsystem called the On-Board Diagnostic (OBD-II) system. The OBD-II is a diagnostic system that monitors the health of the automobile using sensors that measure approximately 100 different engine parameters. Examples of monitored measurements include fuel consumption, engine RPM, coolant temperature and vehicle speed.

To build its fuel efficiency models, Green GPS utilizes a vehicle's OBD-II system and a typical scanner tool in conjunction with a participatory sensing framework. The team is collecting data from vehicles driven by research participants to determine what factors influence fuel consumption. The data collected by the participants is driving the creation of a mathematical model that enable computing fuel consumption of different cars on different road segments. Early studies have shown that a 13% reduction in consumer gas consumption is possible over the shortest path and 6% over the fastest path.

7.4 Microsoft SensorMap

Most of the applications discussed above are based on location data, which is automatically collected based on user behavior. The SensorMap project [49] at Microsoft allows for a more general framework in which users can *choose to publish any kind of sensor data*, with the understanding that such shared knowledge can lead to interesting inferences from the data sets. For example, the sensor data published by a user could be their location information, audio or video feeds, or text which is typed on a keyboard. The goal of the SensorMap project is to store and index the data in a way such that it is efficiently searchable. The application also allows users to index and cache data, so that users can issue spatio-temporal queries on the shared data.

The SensorMap project is part of the SenseWeb project, which allows sharing and exploring of sensor streams over geo-centric interfaces. A number of key design challenges for managing such sensor streams have been discussed in [47]. Other key challenges, which are associated with issues such as the privacy issues involved with continuously collecting and using the sensors which are only intermittently available is discussed in [41].

7.5 Animal and Object Tracking Applications

While social networks are generally defined for the case of people, a similar analysis can be applied to the case of online tracking of animals. For example, animals which are drawn from the same community or family may be considered to have implicit links among them. Such links can be utilized for the perspective of detailed understanding of how community and family membership affects geographical patterns. Such information can be very useful for a variety of applications, such as building disease propagation models among animals. This general idea can also be extended to applications which are outside the domain of social networks. For example, commercial products can be tracked with the use of RFID tags. The products may have implicit links among them which correspond to shared batches or processes during the production and transportation process. Such tracking data can be used in conjunction with linkage analysis in order to determine the causality and origin of tainted products. It can also be used to track the current location of other products which may be tainted.

7.6 Participatory Sensing for Real-Time Services

A variety of *participatory sensing techniques* can be used for enabling real-time services. In participatory sensing, users agree to allow data about them to be transmitted in order to enable a variety of services which are enabled in real time. We note that the main challenges to participatory sensing include the *privacy-issues* associated with such sensing techniques. Therefore participatory sensing is usually utilized only for applications which are *real-time* and *time-critical*; in many cases such services may involve emergencies or health-care applications. Some examples are as follows:

- **Vehicular Participatory Sensing:** In vehicular participatory sensing, a variety of sensor data from vehicles such as mobile location, or other vehicular performance parameters may be continuously transmitted to users over time. Such data may be shared with other users *in the aggregate* in order to preserve privacy. This is the social aspect of such applications, since they enable useful individual decisions based on global patterns of behavior. In addition, vehicular participatory sensing may be used in order to enable quick responses in case of emergencies involving the vehicle operation.

- **Elderly Healthcare:** The ability to carry such devices allows its use for a variety of healthcare applications involving the elderly. For example, elderly patients can use this in order to call for care when necessary. Similarly, such sensing devices can be utilized for a variety of safety and health-care related applications.

8. Future Challenges and Research Directions

In this chapter, we examined the emerging area of integrating sensors and social networks. Such applications have become more commonplace in recent years because of new technologies which allow the embedding of small and unobtrusive sensors in clothing. The main challenges of using such technologies are as follows:

- Such applications are often implemented on a very large scale. In such cases, the database scalability issues continue to be a challenge. While new advances in stream processing have encouraged the development of effective techniques for data compression and mining, mobile applications continue to be a challenge because of the fact that *both* the *number* of streams and rate of data collection may be extremely large.

- A major challenge in sensor-based social networking are the privacy issues inherent in the underlying applications. For example, individuals may not be willing to disclose their locations [33] in order to enable applications such as proximity alerts. In many cases, such constraints can greatly reduce the functionality of such applications. A major challenge in such applications is to provide individual hard guarantees on their privacy level, so that they become more willing to share their real time information.

- The architectural challenges for such systems continue to be quite extensive. For example, the use of centralized processing methods for such large systems does not scale well. Therefore, new methods [47, 49] have moved away from the centralized architecture for stream collection and processing.

The future challenges of such research include the development of new algorithms for large scale data collection, processing and storage. Some advancements [2, 47, 49] have already been made in this direction.

Acknowledgements

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, J. Reich. Mobiscopes for Human Spaces. *IEEE Pervasive*, 6 (2), pp. 20-29, April 2007.

[2] C. C. Aggarwal (ed.) Data Streams: Models and Algorithms, *Springer*, 2007.

[3] C. C. Aggarwal. On Biased Reservoir Sampling in the presence of Stream Evolution. *VLDB Conference*, 2006.

[4] C. C. Aggarwal, H. Wang (ed.) Managing and Mining Graph Data, *Springer*, 2010.

[5] C. C Aggarwal, P. Yu (ed.) Privacy-Preserving Data Mining: Models and Algorithms, *Springer*, 2008.

[6] C. C. Aggarwal, P. S. Yu. Online Analysis of Community Evolution in Data Streams, *SIAM Conference on Data Mining*, 2005.

[7] C. C. Aggarwal, Y. Zhao, P. Yu. On Clustering Graph streams, *SIAM Conference on Data Mining*, 2010.

[8] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the 20th ACM SIGMOD Symposium on Principles of Database Systems*, pages 247–255, 2001.

[9] R. Agrawal and R. Srikant. Privacy preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 439–450, Dallas, TX, May 2000.

[10] N. Alon, P. Gibbons, Y. Matias, M. Szegedy. Tracking Joins and Self-Joins in Limited Storage. *ACM PODS Conference*, 1999.

[11] Alon N., Matias Y., Szegedy M. (The Space Complexity of Approximating Frequency Moments. *ACM STOC Conference*, pp. 20–29, 1996.

[12] C. Asavathiratham, The lnfluence Model: A Tractable Representation for the Dynamics of Networked Markov Chains, *TR, Dept. of EECS, MIT*, Cambridge, 2000.

[13] F. Calabrese, K. Kloeckl, C. Ratti. Wikicity: Real-Time Urban Environments. *IEEE Pervasive Computing*, 6(3), 52-53, 2007. http://senseable.mit.edu/wikicity/rome/

[14] A. Beberg and V. S. Pande. Folding@home: lessons from eight years of distributed computing. *IEEE International Parallel and Distributed Processing Symposium*, pp. 1-8, 2009

[15] D. Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *The Journal of Research into New Media Technologies*, 14(1), pp. 75-90, 2008.

[16] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the SIGMOD/PODS Conference*, pages 211–222, 2003.

[17] D. Chakrabarti, R. Kumar, A. Tomkins. Evolutionary clustering. *KDD Conference*, 2006.

[18] T. Choudhury A. Pentland. The Sociometer: A Wearable Device for Understanding Human Networks. *International Sunbelt Social Network Conference*, February 2003.

[19] T. Choudhury, A. Pentland. Sensing and Modeling Human Networks using the Sociometer, *International Conference on Wearable Computing*, 2003.

[20] T. Choudhury, A. Pentland. Characterizing Social Networks using the Sociometer. *North American Association of Computational Social and Organizational Science*, 2004.

[21] T. Choudhury, B. Clarkson, S. Basu, A. Pentland. Learning Communities: Connectivity and Dynamics of Interacting Agents. *International Joint Conference on Neural Networks*, 2003.

[22] T. Choudhury, M. Philipose, D. Wyatt, J. Lester. Towards Activity Databases: Using Sensors and Statistical Models to Summarize People's Lives. *IEEE Data Engineering Bulletin*, Vol. 29 No. 1, March 2006.

[23] Z. Huang, W. Du, and B. Chen. Deriving private information from randomized data. In *Proceedings of the 2005 ACM SIGMOD Conference*, pages 37–48, Baltimore, MD, June 2005.

[24] H. Kargutpa, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the IEEE International Conference on Data Mining*, pages 99–106, 2003.

[25] A. Clauset, M. E. J. Newman, C. Moore. Finding community structure in very large networks. *Phys. Rev. E 70, 066111*, 2004.

[26] G. Cormode, S. Muthukrishnan. An Improved Data Stream Summary: The Count-Min Sketch and its Applications. *LATIN*, pp. 29–38, 2004.

[27] C. Faloutsos, T. Kolda, J. Sun. Mining Large Time-Evolving Data using Matrix and Tensor Tools, *ICDM Conference*, 2007.

[28] R. K. Ganti, N. Pham, Y.-E. Tsai, T. F. Abdelzaher. PoolView: Stream Privacy in Grassroots Participatory Sensing, *SenSys*, 2008.

[29] R. K. Ganti, Y.-E. Tsai, T. F. Abdelzaher. SenseWorld: Towards Cyber-Physical Social Networks. *IPSN*, pp. 563-564, 2008.

[30] M. Garofalakis, J. Gehrke, R. Rastogi. Querying and mining data streams: you only get one look (a tutorial). *SIGMOD Conference*, 2002.

[31] R. K. Ganti, P. Jayachandran, T. F. Abdelzaher, J. A. Stankovic, SATIRE: A Software Architecture for Smart AtTIRE. *Mobisys*, 2006.

[32] Raghu Ganti, Nam Pham, Hossein Ahmadi, Saurabh Nangia, Tarek Abdelzaher. GreenGPS: A Participatory Sensing Fuel-Efficient Maps Application. *Mobisys*, San Francisco, CA, June 2010.

[33] B. Gedik, L. Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. *ICDCS Conference*, 2005.

[34] T. Guo, K. Iwamura, and M. Koga. Towards high accuracy road maps generation from massive GPS traces data. *Proc. of IGARSS*, pp. 667-670, 2007.

[35] P. Indyk. Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation. *IEEE FOCS*, 2000.

[36] Y. Ioannidis, V. Poosala. Balancing Histogram Optimality and Practicality for Query Set Size Estimation. *ACM SIGMOD Conference*, 1995.

[37] H. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. Sevcik, T. Suel. Optimal Histograms with Quality Guarantees. *VLDB Conference*, 1998.

[38] Y. Jing, S. Baluja. Pagerank for product image search. *WWW Conference*, pages 307–316, 2008.

[39] W. Johnson W., J. Lindenstrauss. Extensions of Lipshitz mapping onto Hilbert Space. *Contemporary Mathematics*, Vol 26, pp. 189–206, 1984.

[40] R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins. Trawling the web for emerging cyber-communities. *WWW Conference*, 1999.

[41] A. Krause, E. Horvitz, A. Kansal, F. Zhao. Toward Community Sensing. *IPSN*, pp. 481–492, 2008.

[42] M. Laibowitz, N.-W. Gong, J. A. Paradiso, Wearable Sensing for Dynamic Management of Dense Ubiquitous Media. *BSN*, 2009.

[43] J. Leskovec, K. J. Lang, A. Dasgupta, M. W. Mahoney. Statistical properties of community structure in large social and information networks. *WWW Conference*, 2008.

[44] Y. Li, J. Han, J. Yang. Clustering Moving Objects, *ACM KDD Conference*, 2004.

[45] J. Lifton, M. Feldmeier, Y. Ono, C. Lewis, J. A. Paradiso. A platform for ubiquitous sensor deployment in occupational and domestic environments. *IPSN*, 2007.

[46] Z. Liu, J. Xu Yu, Y. Ke, X. Lin, L. Chen. Spotting Significant Changing Subgraphs in Evolving Graphs, *ICDM Conference*, 2008.

[47] L. Luo, A. Kansal, S. Nath, F. Zhao. Sharing and exploring sensor streams over geocentric interfaces, *ACM SIGSPATIAL international conference on Advances in geographic information systems*, 2008.

[48] S. Babcock, M. Datar, R. Motwani. Sampling from a Moving Window over Streaming Data. *SIAM Symposium on Discrete Algorithms (SODA)*, 2002.

[49] S. Nath, J. Liu, F, Zhao. SensorMap for Wide-Area Sensor Webs. *IEEE Computer*, 40(7): 90-93, 2008.

[50] M. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 2006.

[51] N. Pham, T. Abdelzaher, S. Nath. On Bounding Data Stream Privacy in Distributed Cyber-physical Systems, *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (IEEE SUTC)*, Newport Beach, CA, June, 2010.

[52] N. Pham, R. K. Ganti, Y. S. Uddin, S. Nath, T. Abdelzaher, Privacy preserving Reconstruction of Multidimensional Data Maps in Vehicular Participatory Sensing, *EWSN*, 2010.

[53] S. Reddy, K. Shilton, G. Denisov, C. Cenizal, D. Estrin, M. B. Srivastava. Biketastic: sensing and mapping for better biking. *CHI*, pp. 1817–1820, 2010.

[54] S. Reddy, D. Estrin, M. B. Srivastava. Recruitment Framework for Participatory Sensing Data Collections.*Pervasive*, pp. 138–155, 2010.

[55] M. Romaine, J. Richardson. State of the Translation Industry. *Translation Industry Report*, My Gengo, 2009.

[56] V. Satuluri, S. Parthasarathy. Scalable graph clustering using stochastic flows: Applications to community discovery. *KDD Conference*, 2009.

[57] J. Sun, S. Papadimitriou, P. Yu, C. Faloutsos. Graphscope: Parameter-free Mining of Large Time-Evolving Graphs, *KDD Conference*, 2007.

[58] H. Tong, S. Papadimitriou, P. Yu, C. Faloutsos. Proximity-Tracking on Time-Evolving Bipartite Graphs, *SDM Conference*, 2008.

[59] J. S. Vitter. Random Sampling with a Reservoir. *ACM Transactions on Mathematical Software*, Vol 11(1), pp. 37–57, 1985.

[60] Z. Yang, S. Zhong, and R. N. Wright. Privacy-preserving classification without loss of accuracy. In *Proceedings of the Fifth SIAM International Conference on Data Mining*, pages 92–102, 2005.

[61] D. Wyatt, T. Choudhury, J. Bilmes. Creating Social Network Models from Sensor Data, *NIPS Network Workshop*, 2007.

[62] D. Wyatt, T. Choudhury, J. Bilmes. Conversation Detection and Speaker Segmentation in Privacy Sensitive Situated Speech Data. *Proceedings of Interspeech*, 2007.

[63] D. Wyatt, T. Choudhury, H. Kautz. Capturing Spontaneous Conversation and Social Dynamics: A Privacy-Sensitive Data Collection Effort. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[64] D. Wyatt, T. Choudhury, J. Bilmes. Learning Hidden Curved Exponential Random Graph Models to Infer Face-to-Face Interaction Networks from Situated Speech Data. *Proceedings of AAAI*, 2008.

[65] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *KDD*, pages 821–826, 2006.

[66] N. Zhang, S. Wang, and W. Zhao. A new scheme on privacy-preserving data classification. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 374–383, New York, NY, USA, 2005. ACM.

[67] F. Zhao, L. Guibas. *Wireless Sensor Networks: An Information Processing Approach*, Morgan Kaufmann, 2004.

[68] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. *PVLDB*, 2(1):718–729, 2009.

[69] http://latitude.google.com

[70] http://www.citysense.com

[71] http://www.movebank.org

[72] http://www-01.ibm.com/software/data/infosphere/streams/

[73] http://www.sensenetworks.com/macrosense.php

[74] http://www.navizon.com

[75] http://ilocalis.com

[76] http://www.trapster.com

[77] National Science Foundation Workshop Report on Future Directions in Networked Sensing Systems: Fundamentals and Applications, The Westin Arlington Gateway, Arlington, VA, November 12-13, 2009.