Towards Quality Aware Information Integration in Distributed Sensing Systems

Wenjun Jiang, Chenglin Miao, Lu Su, *Member, IEEE*, Qi Li, Shaohan Hu, Shiguang Wang, Jing Gao, *Member, IEEE*, Hengchang Liu, *Member, IEEE*, Tarek F. Abdelzaher, *Member, IEEE*, Jiawei Han, *Fellow, IEEE*, Xue Liu, *Member, IEEE*, Yan Gao, and Lance Kaplan, *Fellow, IEEE*

Abstract—In this paper, we present GDA, a generalized decision aggregation framework that integrates information from distributed sensor nodes for decision making in a resource efficient manner. Different from traditional approaches, our proposed GDA framework is able to not only estimate the reliability of each sensor, but also take advantage of its confidence information, and thus achieves higher decision accuracy. Targeting generalized problem domains, our framework can naturally handle the scenarios where different sensor nodes observe different sets of events whose numbers of possible classes may also be different. GDA also makes no assumption about the availability level of ground truth label information, while being able to take advantage of any if present. For these reasons, our approach can be applied to a much broader spectrum of sensing scenarios. In this paper, we also propose two extensions of the GDA framework, i.e., incremental GDA (I-GDA) and parallel GDA (P-GDA) to deal with streaming and large-scale data. The advantages of our proposed methods are demonstrated through both theoretic analysis and extensive experiments.

Index Terms—Information integration, distributed sensing system, participatory sensing, crowd sensing, social sensing, quality

1 Introduction

The proliferation of embedded sensing devices in the recent years has given rise to the fast development and wide deployment of distributed sensing systems. Designed to connect people and the physical world in ways previously unimaginable, such systems have become an integral part of people's everyday lives, hosting a whole spectrum of civilian and military applications, providing useful information to help people with decision making about events in the physical world.

One of the most widely used techniques for decision making is classification [1], [2], which is the task of assigning objects (data) to one of several predefined categories (classes). Its basic idea is to "learn" a function (also called classifier) from a set of training data, in which each object has feature values and a class label, and use the learned function to determine the class labels for newly-arrived data.

- *W. Jiang, C. Miao, L. Su, Q. Li, and J. Gao are with the State University of New York at Buffalo, Buffalo, NY 14260. E-mail: {wenjunji, cmiao, lusu, qli22, jing}@buffalo.edu.*
- *S. Hu, S. Wang, T.F. Abdelzaher, and J. Han are with the University of Illinois at Urbana-Champaign, Urbana, IL 61801. E-mail: {shu17, swang83, zaher, hanj}@illinois.edu.*
- *H. Liu is with University of Science and Technology of China, Hefei 230022, China. E-mail: hcliu@ustc.edu.cn.*
- *X. Liu is with McGill University, Montreal, Quebec, QC H3A 0G4, Canada. E-mail: xueliu@cs.mcgill.ca.*
- *Y. Gao is with LinkedIn, 2029 Stierlin Ct, Mountain View, CA 94043. E-mail: gaoyan.hrb@gmail.com.*
- *L. Kaplan is with Army Research Laboratory, Adelphi, MD 20783. E-mail: lance.m.kaplan@us.army.mil.*

Manuscript received 29 Mar. 2016; revised 4 Feb. 2017; accepted 13 Feb. 2017. Date of publication 6 June 2017; date of current version 8 Dec. 2017. (Corresponding author: Lu Su.) Recommended for acceptance by X. Wang. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TPDS.2017.2712630

The major challenge in applying classification techniques to solve decision making problems in distributed sensing systems lies in the *trade-off between decision accuracy and resource consumption*. On one hand, individual sensors are not reliable, due to various reasons such as incomplete observation, environment and circuit board noise, poor sensor quality, lack of sensor calibration, or even deceptive intent in the first place. To address this sensor reliability problem, one common approach is to integrate information from multiple sensors, as this will likely cancel out the errors of individual sensors and improve decision accuracy. On the other hand, distributed sensing systems usually have limited resources (energy, bandwidth, storage, time, money, or even human labor). Thus, it is often prohibitive to collect data from a large number of sensors due to the potential excessive resource consumption. Therefore, it is challenging to solve the *decision aggregation* problem, that is, to collect and integrate information from distributed sensors to reach a final decision in a resource efficient manner.

Recent efforts have been made to address this challenge under different sensing scenarios. Representative examples include decision aggregation approaches [3], [4] designed for remotely deployed sensing systems where unattended sensor nodes forward their findings through wireless ad hoc networks, and truth discovery schemes [5], [6], [7], [8], [9], [10] that target on social and crowd sensing applications where people themselves act as "sensors" and share their observations via social networks (e.g., in the application of crowdsourced road sensing, specific objects or events on the road, such as potholes, bumps, traffic congestions, and accidents, are detected and reported by the participating drivers through their smartphones). Both strategies dictate that each individual node report only its classification result (*decision*) as opposed to raw data, thus minimizing network transmission and leading to significant saving of system

resources. Individual decisions, upon arrival at the server, are further combined to produce the final decision.

Though yielding reasonably good performance in certain cases, these approaches suffer from a major limitation, that is, they only take as input discrete decision information. Sometimes individual sensors may not be quite confident about their decisions due to various reasons, such as incomplete or noisy observations. In this case, if each sensor's confidence information (the probability that it "believes" the observed event belongs to each candidate class) can also be taken into consideration, we should be able to further improve the final decision accuracy. For example, suppose a vehicle is observed by three sensors that try to determine whether it is a tank or a truck (assuming it is actually a tank). Each sensor then provides a confidence probability vector corresponding to its belief in the vehicle being a tank or a truck. Suppose the three probability vectors are (0.99, 0.01), (0.49, 0.51), and (0.49, 0.51). In this case, only one sensor predicts the vehicle to be a tank, however, deciding so with high confidence, as opposed to the other two that both vote for a truck but with confidences not far from that of random guess, as reflected by their decision probability vectors. Therefore, we expect a reasonable decision aggregation scheme to output tank as the aggregated decision, as opposed to the traditional decision aggregation or truth discovery schemes that only takes discrete decision labels as input, which would likely get the wrong answer, favoring the majority though incorrect decision in our example.

In practice, the decision probability vectors can be obtained from both traditional device-centric sensing systems where sensor nodes conduct explicit classification computations, and newly-emerged people-centric sensing paradigms where people conduct implicit classifications through their logical reasoning. For example, when working with hardware sensor nodes, we can adopt classification algorithms that derive decision probabilities through heuristic metrics like the distance between the observed data and the decision boundary learned from training data; When having people carry out sensing tasks, we can ask each participant to explicitly provide the confidence level of each decision made.

The goal of this paper is to develop a *generalized decision aggregation* (GDA) framework for distributed sensing systems that can address the above challenge, by taking as input each individual sensor's decision probability vectors and computing the aggregated decision (class label) for all events under observation. In pursuing the generalizability so that it is applicable to a full range of sensing scenarios, our proposed GDA framework bears the following properties.

1) Each individual sensor's reliability level is explicitly accounted for when GDA integrates individual decisions. A sensor's reliability information is important as it reflects the general quality of information it can provide. The aggregated decision should favor more reliable sensors and weigh less unreliable ones instead of treating all individuals equally. In reality, however, the reliability information is usually unknown a priori. To address this, in our GDA framework, the sensors' reliability is estimated along with the decision aggregation process and provided as part of the final outputs to the user.

   Please note that high confidence does not necessarily imply high reliability. For example, a sensor should be labeled unreliable if it is always confident about its decisions that are actually wrong. The ability of accounting for sensor reliability thus differentiates our GDA framework from prior information integration schemes (e.g., data fusion [11], [12], [13], [14], [15], [16]) that can also deal with continuous confidence probabilities.
2) Different from traditional decision aggregation schemes that assume all the events are observed by all the sensors, the proposed GDA framework is able to handle the scenarios where different sensor nodes observe different sets of events whose numbers of possible classes may also be different. Doing so enables us to seize more opportunities to estimate sensor reliability, thus leading to better final decision accuracy.
3) In order to be applied to newly emerged sensing scenarios where people are playing increasingly more critical roles, which implies more opportunities for ground truth label collection, we design our GDA framework from ground up to be able to cope with any availability level of label information, and do so in a dynamic and intelligent manner.
4) In addition to pursuing higher accuracy in aggregating decisions, we also want to do it in an efficient and scalable way. Towards this end, two extensions of the GDA framework are proposed in this paper. First, in order to deal with the data that arrive in a streaming manner, we come up with an incremental GDA (denoted as I-GDA) algorithm that can, upon the receipt of new decisions, calculate the true decision as well as update the reliability of each sensor without revisiting the old data. Second, to expedite the processing of massive sensory data, we develop a parallel GDA (denoted as P-GDA) scheme that adopts MapReduce [17] framework to process large-scale data in parallel.

The rest of the paper is organized as follows. We first summarize the related work in Section 2. Section 3 provides an overview of the system model and architecture. In Section 4, we formulate the generalized decision aggregation problem as an optimization program and propose an efficient algorithm to solve this problem, followed by two extensions of it. The proposed algorithms are evaluated in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

Classification techniques are widely used in a full range of sensing scenarios, such as habitat monitoring [18], [19], [20], [21], target surveillance and tracking [22], [23], environmental monitoring [24], activity recognition [25], [26], [27], road sensing and monitoring [28], and many others. Our generalized framework proposed in this paper can essentially be applied to all these sensing scenarios, addressing the decision aggregation problem by corroborating the scattered classification results and making the consolidated near-optimal final decision for the target events, and doing so in a resource efficient manner.

There are prior attempts on similar problems. For example, Su et al. [3], [4] study the decision aggregation problem for remotely deployed sensing systems where very limited label information can be accessed. Recently, the problem of truth discovery [5], [6], [7], [8], [9], [10], [29], [30] is investigated in the data mining and social sensing communities. Their goal is to identify the truth from claims made by

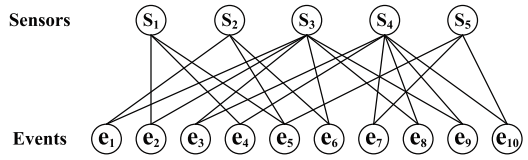


Fig. 1. An example of belief graph.

different information sources (e.g., websites, social network users). These approaches suffer from a major limitation, that is, they only take as input discrete decision information. In contrast, our proposed decision aggregation framework is able to take advantage of the confidence information of each sensor about its decision, and thus achieves higher decision accuracy. Recently, Li et al. [31], [32] propose a truth discovery scheme that can deal with continuous values. Different from this work, we customize the proposed GDA framework to account for the sensor confidence probability vectors that are subject to the decision constraint.

Moreover, the proposed problem and solution in this paper are different from the traditional data aggregation or data fusion schemes in wireless sensor networks. First, data aggregation techniques [33], [34] do not consider sensor reliability, and usually only involve applying simple operations (e.g., mean, min, and max) directly on the raw sensory data. Second, data fusion (or classifier fusion) schemes [11], [12], [13], [14], [15], [16] are designed to gather and combine information from multiple sensors in order to improve the accuracy of target detection and recognition. However, the aforementioned work does not take into consideration the reliability of individual sensor nodes. In contrast, the proposed GDA framework jointly optimizes aggregated decisions and sensor reliability, and can be applied in more general sensing scenarios where the sensor nodes, the observed event sets, and the possible candidate classes can all be different, which can thus be combined in arbitrarily complex manners. In addition to the above work, there are also some data fusion methods that capture the reliability of sensors. For example, optimal fusion rules based approaches [35] take into consideration the prior knowledge of sensory data's error distribution. However, in many cases, the reliability of each sensor is unknown *a priori*. To address this challenge, the proposed GDA framework can estimate sensor reliability in an unsupervised manner without any prior knowledge of sensor reliability. Moreover, the methods developed in [36], [37] can detect faulty sensors. GDA is also different from these methods because it can not only detect anomalous sensors (e.g., the sensors with extremely low reliability), but also estimate the reliability for all the sensors based on their decisions.

The basic GDA framework has been published in [38]. In this paper, we propose two extensions of GDA, i.e., incremental GDA (I-GDA) and parallel GDA (P-GDA), to deal with streaming data and large-scale data, respectively.

3 System Overview

We now give an overview of our system model and architecture.

3.1 System Model

We consider a sensing system consisting of n sensor nodes that are denoted by $\mathcal{S} = \{s_i | \ i = 1, 2, \ldots, n\}$. The sensor nodes collect information about the events taking place within their sensing ranges, and classify these events into predefined classes. Formally, we let $\mathcal{E} = \{e_i | i = 1, 2, \ldots, t\}$ denote the sequence of events (sorted in chronological order) observed by the sensor nodes. Generally, each sensor observes a subset of events, and each event is observed by a subset of sensors. The relationship between sensor nodes and events can be represented as a bipartite graph, called *belief graph*, where vertices are partitioned into sensors and events, and edges represent the observation relationship of sensor-event pairs, as illustrated in Fig. 1. We denote the belief graph by an affinity matrix called observation matrix $\mathbf{A} = (a_{ij})_{t \times n}$, where a_{ij} indicates whether event e_i is observed by sensor s_j.

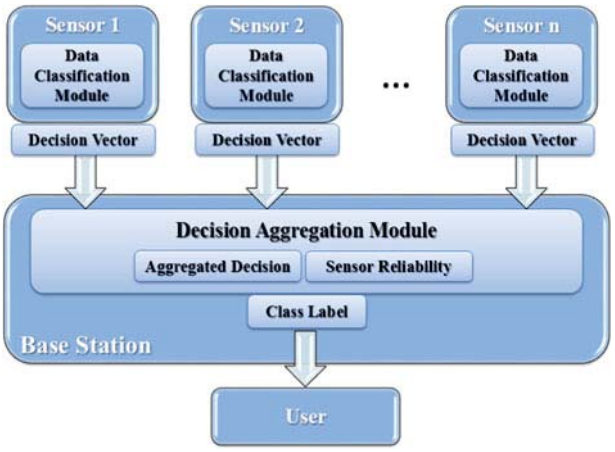


Fig. 2. System architecture.

In this case, suppose the mission of the sensing system is to classify different vehicle types, specifically, to find out whether an observed vehicle is a tank, a jeep, or a truck. As shown, 10 events are observed by 5 sensor nodes. Each event corresponds to a vehicle. Each sensor can be either a sensing device deployed on the roadside, or a pedestrian in the vicinity.

3.2 System Architecture

In this section, we provide an overview of the system architecture. The system contains two modules: a data classification module and a decision aggregation module. They are deployed on two different platforms: sensor nodes, and the base station. Fig. 2 illustrates the system architecture. We next discuss each of these two modules in more detail.

3.2.1 Data Classification Module

The data classification module runs on individual sensor nodes. It locally classifies the events observed by each sensor node, and uploads the classification results (i.e., *decision vector*) to the base station. The decision vector of node s_j for e_i is a probability vector denoted by $\mathbf{d}_i^j = (d_{i1}^j, \ldots, d_{im_i}^j)$, where m_i is the number of possible classes of event e_i. In this vector, each element probability, say d_{ik}^j, represents the confidence level in which s_j "believes" the observed event belongs to the kth class. For example, suppose sensor s_1 in Fig. 1 outputs a decision vector for event e_2 (i.e., $\mathbf{d}_2^1 = (0.8, 0.1, 0.1)$). This implies that s_1 believes that with 80 percent probability e_2 is a tank, and is a jeep or truck with 10 percent probability each.

3.2.2 Decision Aggregation Module

As shown in Fig. 2, the decision aggregation module resides on the base station. It combines the decision vectors from multiple

sensor nodes through iteratively updating the *aggregated decision* and *sensor reliability*, which are explained as following:

- Aggregated Decision: The aggregated decision for an event e_i is also a probability vector, denoted by $\mathbf{x}_i = (x_{i1}, \ldots, x_{im_i})$. It represents the consensus of the sensor nodes on the probability that e_i belongs to each candidate class.
- Sensor Reliability: As discussed in Section 1, the reliability levels of individual sensor nodes should be taken into account when aggregating the decisions of multiple nodes. To capture sensor reliability, we associate each node, say s_j, with a non-negative weight w_j, where higher weights indicate higher reliability.

Finally, as shown in Fig. 2, the decision aggregation module takes the highest probability in the aggregated decision vector as the predicted class label of each unlabeled event and sends the result to the user.

4 Generalized Decision Aggregation

In this section, we first formulate the generalized decision aggregation problem as an optimization program on the belief graph. Then a two-step block coordinate descent method [39] is proposed to solve the optimization problem. Based on this general GDA framework, two extensions are further developed to deal with streaming data and large-scale data respectively.

4.1 Optimization Program

With the previously introduced terminology and notions, we formulate the following optimization program:

$$\mathbf{P}: \min \quad \sum_{i=1}^{t}\sum_{j=1}^{n} a_{ij} w_j ||\mathbf{x}_i - \mathbf{d}_i^j||^2 \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} \exp(-w_j) = 1 \tag{2}$$

$$\mathbf{x}_i \geq \mathbf{0}, \quad |\mathbf{x}_i| = 1 \quad \text{for} \quad i = 1, 2, \ldots, t. \tag{3}$$

where the observation matrix $\mathbf{A} = (a_{ij})_{t \times n}$ and the decision vector $\mathbf{d}_i^j$ are the *constants* of the optimization program while the aggregated decision $\mathbf{x}_i$ and sensor reliability w_j are the *variables*.

Objective Function. The objective function (1) aims at minimizing the disagreement over the belief graph, namely, the weighted summation of the distances between the decisions of individual sensor nodes and the aggregated decision. In this case, we use squared L2 norm as the distance function

$$||\mathbf{x}_i - \mathbf{d}_i^j||^2 = (x_{i1} - d_{i1}^j)^2 + (x_{i2} - d_{i2}^j)^2 + \ldots + (x_{im_i} - d_{im_i}^j)^2.$$

Intuitively, the optimal aggregated decision should be close to the majority of individual decisions[1].

Furthermore, the sensors with higher reliability score (i.e., w_j) should have more impact on the weighted summation. In other words, more reliable sensors would incur higher penalties if they deviate far away from the aggregated decision, as compared to less reliable ones. This way, the objective function tends to be minimized when the aggregated decision agrees with that of reliable sensors.

1. Here we assume the majority of the sensor nodes are functioning appropriately and thus can make reasonable decisions.

Constraints. Next, we elaborate on the constraints that our objective function is subject to.

- Reliability Constraint (2) is a regularization function. It is used to prevent the sensor weight w_j from going to infinity, otherwise the optimization problem would become unbounded. In fact, the most straightforward choice of regularization function could be $\sum_{j=1}^{n} w_j = 1$, which is unsuitable for our purpose as an optimal solution is achieved when the aggregated decision is set to that of any single sensor, whose weight is set to 1 and the rest sensors 0. Therefore, we propose to formulate the regularization function using the sum over exponential value of weights. Exponential function is used to regularize weights so that they are rescaled by logarithm (the range of weights becomes smaller). One advantage of this regularization formulation is that a closed-form optimal solution can be derived.
- Decision Constraint (3) is used to guarantee that the elements of the decision probability vector $\mathbf{x}_i$ be non-negative, and sum to 1 (i.e., $\mathbf{x}_i$'s L1 norm $|\mathbf{x}_i| = \sum_{k=1}^{m_i} x_{ik} = 1$).

Unfortunately, $\mathbf{P}$ is not a convex program. This makes it difficult to find the global optimal solution. To address this challenge, we propose to solve $\mathbf{P}$ using the block coordinate descent method [39]. The basic idea is as follows: In each iteration, we update the values of sensor reliability and aggregated decision alternatively and separately. In particular, in the first step, we fix the weight (w_j) of each sensor node, and solve $\mathbf{P}$ with respect to the aggregated decision ($\mathbf{x}_i$) only. In the second step, $\mathbf{x}_i$ is fixed and $\mathbf{P}$ is solved with respect to w_j. The two-step process is repeated until convergence, which is guaranteed by the property of the block coordinate descent method. That is, if we can find the optimal aggregated decision (sensor reliability) when sensor reliability (aggregated decision) is fixed, convergence can be achieved [39]. Next, we give detailed explanation on these two steps, and show that each step itself is convex, and thus has a globally optimal solution.

4.2 Updating Aggregated Decision

When the reliability w_j of each sensor is fixed, we update the aggregated decision $\mathbf{x}_i$ for each event in order to minimize the weighted distances between $\mathbf{x}_i$ and the decisions $\mathbf{d}_i^j$ made by individual sensor nodes

$$\mathbf{P_x}: \min \quad \sum_{i=1}^{t}\sum_{j=1}^{n} a_{ij} w_j ||\mathbf{x}_i - \mathbf{d}_i^j||^2$$
$$\text{s.t.} \quad \mathbf{x}_i \geq \mathbf{0}, \quad |\mathbf{x}_i| = 1 \quad \text{for} \quad i = 1, 2, \ldots, t.$$

Different from $\mathbf{P}$, $\mathbf{P_x}$ has only one set of variables (i.e., $\mathbf{x}_i$'s), and thus is a convex program. This ensures that we can find globally optimal aggregated decisions. The detailed steps are as follows. First, we denote the objective function by

$$f(\mathbf{x}) = \sum_{i=1}^{t}\sum_{j=1}^{n} a_{ij} w_j \sum_{k=1}^{m_i} (x_{ik} - d_{ik}^j)^2,$$

then the optimal solution can be obtained through setting the partial derivative with respect to $\mathbf{x}$ to zero,

$$\frac{\partial f(\mathbf{x})}{\partial x_{ik}} = \sum_{j=1}^{n} 2 a_{ij} w_j (x_{ik} - d_{ik}^j) = 0,$$

for $i = 1, 2, \ldots, t$ and $k = 1, 2, \ldots, m_i$.

Solving this equation, we are able to get the optimal value of x_{ik}

$$x_{ik} = \frac{\sum_{j=1}^{n} a_{ij} w_j d_{ik}^j}{\sum_{j=1}^{n} a_{ij} w_j}.$$

Therefore, the optimal aggregated decision vector is actually the weighted average of individual decision vectors

$$\mathbf{x}_i = \frac{\sum_{j=1}^{n} a_{ij} w_j \mathbf{d}_i^j}{\sum_{j=1}^{n} a_{ij} w_j}. \tag{4}$$

One should note that when we solve for $\mathbf{x}_i$, we do not take into account the decision constraint (Eqn. (3)). This is because the aggregated decisions obtained from Eqn. (4) can automatically satisfy the constraint. In particular, each individual decision $\mathbf{d}_i^j$ is a probability vector, obviously we have $\mathbf{d}_i^j \geq 0$ and $|\mathbf{d}_i^j| = 1$. Thus, it can be derived that

$$|\mathbf{x}_i| = \sum_{k=1}^{m_i} x_{ik} = \frac{\sum_{j=1}^{n} a_{ij} w_j |\mathbf{d}_i^j|}{\sum_{j=1}^{n} a_{ij} w_j} = 1$$

Moreover, since both a_{ij} and w_j are nonnegative, it is clear that $x_{ik} \geq 0$. Therefore, $\mathbf{x}_i$'s automatically satisfy the decision constraint.

4.3 Updating Sensor Reliability

Next, we fix the values of the aggregated decision $\mathbf{x}_i$, and update the reliability of each sensor w_j through solving the following optimization program:

$$\mathbf{P_w}: \min \quad \sum_{i=1}^{t} \sum_{j=1}^{n} a_{ij} w_j ||\mathbf{x}_i - \mathbf{d}_i^j||^2$$
$$\text{s.t.} \quad \sum_{j=1}^{n} \exp(-w_j) = 1.$$

Similar to the previous step, $\mathbf{P_w}$ has only one set of variables, the w_j's. And the decision constraint (Eqn. (3)) in $\mathbf{P}$ is just constant here. $\mathbf{P_w}$ is clearly convex since the objective function is linear with respect to w_j, while the constraint is a convex function.

We use the method of Lagrange multipliers to solve $\mathbf{P_w}$. We first take a look at the Lagrangian of $\mathbf{P_w}$

$$L(\mathbf{w}, \lambda) = \sum_{i=1}^{t} \sum_{j=1}^{n} a_{ij} w_j ||\mathbf{x}_i - \mathbf{d}_i^j||^2 + \lambda \left(\sum_{j=1}^{n} \exp(-w_j) - 1 \right).$$

In $L(\mathbf{w}, \lambda)$, λ is a Lagrange multiplier, corresponding to the reliability constraint. It can be interpreted as the "shadow price" charged for the violation of the constraint.

Let the partial derivative of Lagrangian with respect to w_j be 0

$$\frac{\partial L(\mathbf{w}, \lambda)}{\partial w_j} = \sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2 - \lambda \exp(-w_j) = 0,$$

we can get

$$\frac{\sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2}{\lambda} = \exp(-w_j). \tag{5}$$

Summing both sides over j's, we have,

$$\frac{\sum_{j=1}^{n} \sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2}{\lambda} = \sum_{j=1}^{n} \exp(-w_j) = 1,$$

from which we can derive that

$$\lambda = \sum_{j=1}^{n} \sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2. \tag{6}$$

Plugging Eqn. (6) into Eqn. (5), we obtain a closed-form solution of reliability:

$$w_j = \log \left(\frac{\sum_{j=1}^{n} \sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2}{\sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i - \mathbf{d}_i^j||^2} \right). \tag{7}$$

As can be seen, the reliability of a sensor is the log ratio between the summed decision deviation (i.e., the difference between a sensor's decision and the aggregated decision) of all the sensors and the decision deviation of this sensor. Sometimes, the summed decision deviation may dominate individual decision deviations, and thus diminish the variance among the sensor reliabilities. In this case, we can replace the the summed decision deviation by the maximum decision deviation among all the sensors.

4.4 Algorithm

The detailed steps of the generalized decision aggregation algorithm are shown in Algorithm 1. The algorithm takes as input the observation matrix $\mathbf{A}$ as well as the individual decision of each sensor s_j for each event e_i (i.e., $\mathbf{d}_i^j$). It starts by initializing the aggregated decisions randomly[2] (line 1). The iterative process then begins in line 3. First, we collect the aggregated decision of each event observed by a sensor s_i, and update its reliability via Eqn. (7) (line 5). Then, the sensors' reliability information are used to consolidate the aggregated decision of each event e_i through Eqn. (4) (line 7). Finally, each event is assigned to the class corresponding to the highest probability in the aggregated decision (line 10).

Algorithm 1. Generalized Decision Aggregation

Input: Observation matrix $\mathbf{A}$, individual decisions $\mathbf{d}_i^j$, and error threshold ϵ;
Output: The class label for each event L_i;

1: Initialize $\mathbf{x}_i^{(0)}$, $\mathbf{x}_i^{(1)}$ randomly.
2: $\tau \leftarrow 1$
3: **while** $\sqrt{\sum_{i=1}^{t} ||\mathbf{x}_i^{(\tau)} - \mathbf{x}_i^{(\tau-1)}||^2} > \epsilon$ **do**
4: **for** $j \leftarrow 1$ to n **do**
5: $w_j^{(\tau+1)} \leftarrow \log \left(\frac{\sum_{j=1}^{n} \sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i^{(\tau)} - \mathbf{d}_i^j||^2}{\sum_{i=1}^{t} a_{ij} ||\mathbf{x}_i^{(\tau)} - \mathbf{d}_i^j||^2} \right)$
6: **for** $i \leftarrow 1$ to t **do**
7: $\mathbf{x}_i^{(\tau+1)} \leftarrow \frac{\sum_{j=1}^{n} a_{ij} w_j^{(\tau+1)} \mathbf{d}_i^j}{\sum_{j=1}^{n} a_{ij} w_j^{(\tau+1)}}$
8: $\tau \leftarrow \tau + 1$
9: **for** $i \leftarrow 1$ to t **do**
10: **return** $L_i \leftarrow \text{argmax}_k \, x_{ik}^{(\tau)}$

2. Since $\mathbf{P}$ is not a convex program, the block coordinate descent based algorithm would probably converge to local optimum. A common way to address this problem is to run the algorithm multiple times with a different set of randomly chosen initial aggregated decisions and select the best solution.

TABLE 1
Events

Event	Sensor Node	Decision Vector	Ground Truth
e_1	s_2	(0.1, 0.8, 0.1)	1
	s_3	(0.6, 0.2, 0.2)	
e_2	s_1	(0.3, 0.5, 0.2)	3
	s_3	(0.1, 0.3, 0.6)	
e_3	s_3	(0.8, 0.1, 0.1)	1
	s_4	(0.8, 0.1, 0.1)	
e_4	s_1	(0.4, 0.3, 0.3)	2
	s_4	(0.2, 0.6, 0.2)	
e_5	s_1	(0.5, 0.5, 0)	2
	s_2	(0.7, 0.3, 0)	
	s_5	(0.3, 0.7, 0)	

4.5 Performance Analysis

In each iteration, the GDA algorithm takes $O(mnt)$ time, where n and t represent the number of sensors and events, while $m = \max_{e_i \in \mathcal{E}} m_i$ is the maximum number of classes among all the events. Also, the convergence rate of coordinate descent method is usually linear [39] (we actually fix the number of iterations in the experiments). In practice, the number of candidate classes of the observed events and the number of sensor nodes that observe the same events are usually small. Thus, the computational complexity of the algorithm can be considered linear with respect to the number of events. Consequently, the proposed algorithm is not more expensive than the classification algorithms, and thus can be applied to any platform running classification tasks. Furthermore, since wireless/wired communication is the dominating factor of the energy consumption in distributed sensing systems, our algorithm actually saves much more energy than it consumes since it significant reduces the amount of information delivered by each sensor by transforming its raw data into decisions.

4.6 Example

We now walkthrough a simple example to illustrate the iterative process of the GDA algorithm. Table 1 provides the information of the first 5 events shown in Fig. 1. We omit other events' information due to space limitation. In this table, we list the sensor nodes that observe each of the 5 events, and the corresponding decision probability vectors generated by the sensors. In addition, the ground truth label of each event is given in the last column.

We apply the GDA algorithm to the sensing system in Fig. 1. Initially, the aggregated decision of each event is set as the average of individual decisions made by the sensors that observe this event. The predicted label corresponds to the class with the highest probability. In rare cases where ties occur, we break them randomly.

Then, the algorithm starts to iterate, and update the values of sensor reliability and aggregated decisions repeatedly. After the algorithm converges, as shown in Table 2, the predicted label for each event exactly matches the ground truth. From the results, we have several observations. For example, sensor s_2 and s_3 have conflicting decisions about event e_1, and s_2 is more confident with its decision. Thus, the simple averaged decision gives a predicted label of 2, which contradicts against the ground truth. In this case, the GDA algorithm outputs the correct label, because it takes into account the reliability of individual sensors. As can be seen in Table 2, the reliability score of s_3 is much higher than that of s_2, so the aggregated result should favor s_3's decision. A decision is considered to be correct (or incorrect) if the class with the highest probability in the decision vector matches (or differs from) the ground truth label. As shown in Table 2, the sensors that can make more correct decisions are assigned higher reliability scores.

TABLE 2
Final Results

Event	Aggregated Decision	Label	Sensor	Reliability
e_1	(0.5701, 0.2358, 0.1941)	1	s_1	1.6093
e_2	(0.1517, 0.3517, 0.4966)	3	s_2	0.2731
e_3	(0.8, 0.1, 0.1)	1	s_3	4.5801
e_4	(0.2505, 0.5243, 0.2252)	2	s_4	4.7438
e_5	(0.3712, 0.6288, 0)	2	s_5	3.9157

4.7 Incremental GDA

In many real applications, events (e.g., detection of animals) may take place sequentially, and as a result, the decisions may arrive in a "streaming" manner. Clearly, it is not economic to invoke the decision aggregation procedure once again on the whole data set whenever some new decisions are reported, since tremendous unnecessary computations would be incurred to update the decisions on outdated events. Therefore, our goal is to develop an incremental decision aggregation mechanism that can, upon the receipt of new decisions, calculate the true decision as well as update the reliability of each sensor without revisiting the old data.

Formally, for each sensor s_j, we represent its "old information" regarding the outdated events by $d_j = \sum_{i=1}^{t-1} a_{ij} \|\mathbf{x}_i - \mathbf{d}_i^j\|^2$, which summarizes the difference between its decisions and the aggregated decisions on all the events observed in the past. With d_j, on the occurrence of the tth event e_t, we are able to carry out decision aggregation and reliability update as follows:

- *Calculating the aggregated decision for e_t.*

$$\mathbf{x}_t \leftarrow \frac{\sum_{j=1}^{n} a_{tj}\omega_j \mathbf{d}_t^j}{\sum_{j=1}^{n} a_{tj}\omega_j}$$

- *Updating the reliability of each sensor.*

$$\omega_j \leftarrow \log\left(\frac{\sum_{j=1}^{n}\left(d_j + a_{tj}\|\mathbf{x}_t - \mathbf{d}_t^j\|^2\right)}{d_j + a_{tj}\|\mathbf{x}_t - \mathbf{d}_t^j\|^2}\right)$$

This idea is summarized in the following incremental generalized decision aggregation algorithm.

As can be seen in Algorithm 2, the "old information" d_j is used as the input to reduce unnecessary computations. After the algorithm calculates the aggregated decision for e_t (line 1), it will update d_j by combining "old information" with "new information", which is the difference between each sensor's decision on e_t and the aggregated decision (line 3). With updated d_j for each sensor node (line 5), the algorithm now can update the reliability for each sensor without revisiting the aggregated decisions on the past events. Although in I-GDA we update each aggregated decision $\mathbf{x}_t$ and each sensor reliability w_j only once instead of repeating the updating until convergence. The sensor reliability w_j will still gradually stabilize as the number of events increases.

Algorithm 2. Incremental GDA

Input: The aggregated decisions for the previous $t-1$ events $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1}\}$, each sensor's reliability at time $t-1$ (i.e., ω_j), each sensor's observation indicator for event e_t (i.e., a_{tj}), each sensor's decision on e_t (i.e., $\mathbf{d}_t^j$), and each sensor's information regarding the previous $t-1$ events $d_j = \sum_{i=1}^{t-1} a_{ij} \|\mathbf{x}_i - \mathbf{d}_i^j\|^2$;
Output: The class label of event e_t (i.e., L_t);

1: $\mathbf{x}_t \leftarrow \frac{\sum_{j=1}^{n} a_{tj}\omega_j \mathbf{d}_t^j}{\sum_{j=1}^{n} a_{tj}\omega_j}$
2: **for** each sensor node s_j **do**
3: $\quad d_j \leftarrow d_j + a_{tj}\|\mathbf{x}_t - \mathbf{d}_t^j\|^2$
4: **for** each sensor node s_j **do**
5: $\quad \omega_j \leftarrow \log\left(\frac{\sum_{j=1}^{n} d_j}{d_j}\right)$
6: **return** $L_t \leftarrow argmax_k\, x_{tk}$

Moreover, in some cases, when the real-time response is not that important, then we can further improve the efficiency of the system. For instance, we can group the events in sequential chunks: $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_l, \ldots\}$, where each chunk ε_l contains the events taking place within the lth time interval. We then employ the above two-step computation on the event chunks instead of individual events. This way, we update reliability of each sensor once for each chunk instead of each event, so significant computational resources can be saved.

4.8 Parallel GDA

With the proliferation of all kinds of sensing devices, an explosive increase of sensory data are expected in the near future. In order to deal with such kind of massive data, we propose to adopt cloud computing techniques to process data in parallel. Many parallel programming models have potential to be applied to the proposed GDA framework. Among them, one of the most widely used models is the MapReduce framework [17], which is adopted in this paper.

By implementing the GDA algorithm using the MapReduce framework, we can make our system scalable to large amount of data. When using the MapReduce framework, we are allowed to customize two major functions: (1) the map function that scan the input entries to generate intermediate key/value pairs, and (2) the reduce function that merges all intermediate pairs associated with the same key. In the GDA algorithm, both the aggregated decision for each event and the reliability for each sensor can be calculated independently, making this algorithm well adaptable to the MapReduce framework. Next, we will elaborate on design of our parallel GDA algorithm.

The inputs of our algorithm are decisions for each event from several sensors, so each decision can be expressed as the form of three tuple: $(i, j, \mathbf{d}_i^j)$, which represents respectively event ID i, sensor ID j and the decision vector $\mathbf{d}_i^j$ made by sensor s_j for event e_i. We initialize sensor reliabilities as uniform weights. Then we iteratively update aggregated decisions and sensor reliabilities until convergence. Each round of iteration is implemented by two MapReduce jobs:

- *Updating Aggregated Decision*. The aggregated decision is updated as follows: $\mathbf{x}_i \leftarrow \frac{\sum_{j=1}^{n} a_{ij}\omega_j \mathbf{d}_i^j}{\sum_{j=1}^{n} a_{ij}\omega_j}$. Clearly, this computation step can be executed independently for each event, making it easy to parallelize. Specifically, each mapper processes the input tuple $(i, j, \mathbf{d}_i^j)$ and forward it to reducers in the form of key/value pairs, in which the key is the event ID i and the value is the rest. After shuffled by Hadoop, all decisions associated with the same event e_i are forwarded to the same reducer, where they are used to derive $\mathbf{x}_i$, as the pseudo code in Algorithm 3 summarizes. After the MapReduce job, the aggregated decision can be directly collected from the output of the Reduce function.
- *Updating reliability*. For each sensor s_j, we update its reliability as $w_j \leftarrow \log\left(\frac{\sum_{j=1}^{n}\sum_{i=1}^{t} a_{ij}\|\mathbf{x}_i - \mathbf{d}_i^j\|^2}{\sum_{i=1}^{t} a_{ij}\|\mathbf{x}_i - \mathbf{d}_i^j\|^2}\right)$, in which the total error of each sensor s_j, i.e., $\sum_{i=1}^{t} a_{ij}\|\mathbf{x}_i - \mathbf{d}_i^j\|^2$, can be calculated independently. Under MapReduce framework, each mapper processes the input tuple $(i, j, \mathbf{d}_i^j)$ and calculates partial decision error of $\mathbf{d}_i^j$, i.e., $\|\mathbf{x}_i - \mathbf{d}_i^j\|^2$. The result is then associated with the sensor ID j and sent to reducers. The reducers then sum up these partial errors and get total error for each sensor. The pseudo code is summarized in Algorithm 4. After the MapReduce job, we derive sensor reliabilities according to the formula shown in line 8.

5 Performance Evaluation

In this section, we report experimental results on both synthetic data and a set of realistic audio recordings. We discuss the experimental setup in Section 5.1. Then, we compare GDA with the naïve majority voting scheme as well as state-of-the-art truth discovery and data fusion approaches in Section 5.2. Finally, the performance of incremental GDA and parallel GDA are discussed in Sections 5.3 and 5.4 respectively.

Algorithm 3. MapReduce for Updating Aggregated Decision

Input: Individual decisions $\mathbf{d}_i^j$ of sensor j for event i, and reliability ω_j for each sensor j;
Output: Aggregated decision $\mathbf{x}_i$ for each event i;

1: **function** Map (Key id, Value v)
2: $\quad (i, j, \mathbf{d}_i^j) \leftarrow v$
3: $\quad$ EMIT$(i, (j, \mathbf{d}_i^j))$
4: **function** Reduce (Key i, Value[] v)
5: $\quad [j, \mathbf{d}_i^j] \leftarrow v$
6: $\quad sum\_decision \leftarrow \sum_{j=1}^{n} \omega_j \cdot \mathbf{d}_i^j$
7: $\quad sum\_weight \leftarrow \sum_{j=1}^{n} \omega_j$
8: $\quad \mathbf{x}_i \leftarrow sum\_decision / sum\_weight$
9: $\quad$ EMIT$(i, \mathbf{x}_i)$

5.1 Experiment Settings

For comparison purposes, we include four baseline methods in the experiment. The first two baselines take the discrete decision information as input:

- *Majority Voting*. It counts the votes for each class, and picks the one with the highest vote count.
- *EM TruthFinder*. It is a state-of-the-art truth discovery approach [5], which uses Expectation Maximization

(EM) algorithm to jointly optimize the correctness of the claims made by a group of sources and the reliability of these sources.

For each individual sensor decision, we feed these two baselines with the class label that has the highest confidence in the decision vector.

The other two baselines are representative data fusion schemes [11], [12], [13], [14] that also take advantage of confidence levels of individual sensors when integrating their decisions:

- *Product-Rule Fusion*. Under our problem setting, it is equivalent to multiplying the decision probabilities over the sensors and labeling the event with the class corresponding to the largest probability product

$$L_i \leftarrow \arg\max_k \prod_{j=1}^{n} d_{ik}^j. \tag{8}$$

- *Sum-Rule Fusion*. It behaves exactly the same as the product rule fusion, except that summation is used instead of multiplication

$$L_i \leftarrow \arg\max_k \sum_{j=1}^{n} d_{ik}^j. \tag{9}$$

As can be seen, if normalized by dividing each sum by n, the sum rule fusion is equivalent to simply averaging the decision probability vectors on each event and labeling this event corresponding to the class with the highest probability in the averaged decision vector.

Algorithm 4. MapReduce for Updating Reliability

Input: Individual decisions $\mathbf{d}_i^j$ of sensor s_j for event e_i, and aggregated decision $\mathbf{x}_i$ for each event e_i;
Output: Reliability ω_j for each sensor s_j;

```
function MAP (Key id, Value v)
    (i, j, d_i^j) ← v
    decision_error ← ||x_i − d_i^j||^2
    EMIT(j, decision_error)
function REDUCE (Key j, Value[] v)
    total_error_j ← Σ_{i=1}^{t} v_i
    EMIT(j, total_error_j)
ω_j ← log( (Σ_{j=1}^{n} total_error_j) / total_error_j )
```

Both of the above fusion schemes, despite of being able to deal with continuous confidence probabilities, fail to take into account the varying sensor reliability.

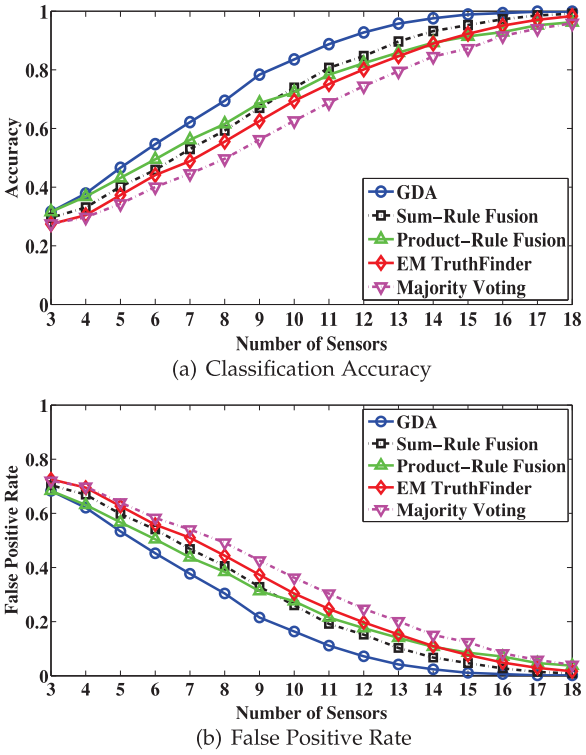
All the experiments in this section except for the MapReduce experiments are conducted on a Windows machine with 16 GB of memory and Intel Core i7 processor.

5.2 Experimental Results of GDA

In this section, we first evaluate the performance of GDA on synthetic data, and then use real audio data to see how the proposed GDA algorithm works in real-world applications.

5.2.1 Synthetic Data

In this experiment, we simulate a sensing system where a set of events are observed and monitored by multiple sensor nodes. In particular, we randomly generate events of different class labels. For simplicity, we assume that each class contains the same number of events. Then, we generate sensor nodes with uniformly distributed reliability. After that, we generate decision vector $\mathbf{d}_i^j$ through Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}_i)$ [40], where $\boldsymbol{\alpha}_i$ is a positive real m_i-dimensional vector with $\mathrm{E}[d_{ik}^j] = \frac{\alpha_{ik}}{\sum_{k=1}^{m_i} \alpha_{ik}}$. In our experiment, all the dimensions of $\boldsymbol{\alpha}_i$ are set to a constant c_i except for the one corresponding to the true class label, which is set to another constant r_{ij}. We let the difference $r_{ij} - c_i$ be non-negative and proportional to the reliability of source j because if a sensor j has a larger r_{ij}, it is more likely to make correct decision on event i.

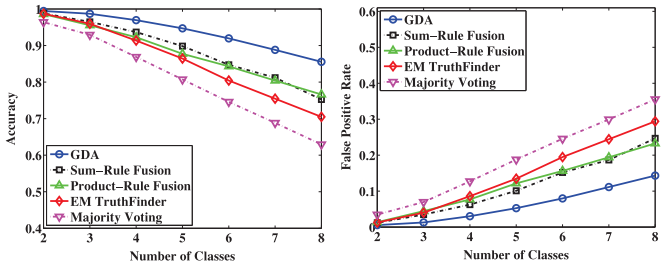


(a) Classification Accuracy
(b) False Positive Rate

Fig. 3. Classification performance under different sensor numbers on synthetic data.

1) Classification Performance under Varying Number of Sensors. We first demonstrate the classification performance under varying number of sensor nodes that observe the same events. We generate 6 classes with 100 events each, where the number of observing sensors varies from 3 to 18. The experiment is repeated 10 times. We report the average results.

Figs. 3a and 3b show, for all approaches, their classification accuracies (the percentage of correctly classified events, equivalent to true positive rate in this case), and false positive rates (the percentage of misclassifications of all the events that are classified to be of a particular class, then averaged among all classes), respectively. As clearly seen, our GDA framework outperforms the other approaches under any number of observing sensors in terms of both classification accuracy and false positive rate, as the classification benefits from accounting for both sensor reliability and decision confidences. On the other end of the spectrum, the majority voting yields the worst performance as it disregards useful information (sensor reliability and decision confidences) that otherwise would be useful for reaching



(a) Classification Accuracy (b) False Positive Rate

Fig. 4. Classification performance under different number of classes on synthetic data.

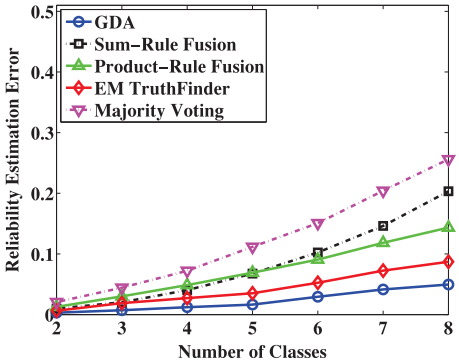


Fig. 5. Estimation errors of sensor reliability under different number of classes.

more accurate final decisions. The EM TruthFinder and data fusion approaches take only one factor (sensor reliability for EM TruthFinder, or decision confidences for data fusion) into consideration when aggregating individual decisions, therefore they, though outperforming majority voting, still fall short compared to our GDA approach, which utilizes *all* useful information. One other interesting observation is that all methods show similar performance when the number of sensors is either very small or quite large (e.g., 3 and 18, respectively in our experiments). This makes sense because, on one hand, when the number of sensors that observe the same events is small, it is hard to improve upon their individual poor decisions; On the other hand, as the number of sensors increases, each event is being observed by more and more diversified sensor nodes, which are more and more likely to cancel out each other's biases and errors, thus reaching better classification results. When there are a sufficiently large number of observing sensors, even the most naïve approach (e.g., majority voting) can achieve near perfect classification accuracy.

2) Classification Performance under Varying Number of Classes. Next, we look at how GDA's classification performance compares to the other approaches with varying number of classes. The results are shown in Fig. 4. In this experiment, we assume that each event is observed by 10 different sensors, and each class contains 100 events. The number of classes ranges from 2 to 8.

Figs. 4a and 4b show the classification accuracies and false positive rates of all approaches. As seen, our GDA approach consistently outperforms the other methods regardless of the number of classes, where the relative effectiveness of all studied approaches remains the same as that of the previous experiment. This is not surprising, as, still, the scheme that can take advantage of more information performs better. Also seen from the figures, it is clear that all approaches' classification performance degrades as the number of classes increases. This is generally expected for any classification task as the more candidate classes there are, the more confusion the classification algorithms need to come through. We do, however, notice that as the number of classes increases, our GDA's performance degradation is slightly slower than the other approaches in general.

3) Sensor Reliability. Using the same setting as the previous experiment, we now examine how the different schemes perform in terms of estimating sensor reliability by comparing the reliability estimation errors of our proposed GDA framework to that of the other four approaches, under varying number of classes. In particular, the reliability estimation error is computed as follows. For each individual sensor node, its ground truth reliability is defined as its standalone classification accuracy derived from comparing its individual decisions to the ground truth event labels, and its estimated reliability under a particular scheme is the classification accuracy derived from comparing its individual decisions to the aggregated decision reached by that scheme. A sensor's reliability estimation error is thus computed as the relative error between its estimated and ground truth reliability.

The results are shown in Fig. 5. Similar to previous experiments' results, our proposed GDA still consistently outperforms the others. In particular, we see that the approaches that take sensor reliability into account when performing decision aggregations (i.e., GDA and EM TruthFinder) achieve better performance than those who do not (i.e., Data fusion and Majority voting). Also, as the number of classes becomes larger, the estimation performance of all approaches gets poorer. Similar to the previous experiment, a higher number of classes would lengthen the distance between the aggregated decisions and ground truth event labels, thus causing more inaccurate sensor reliability estimates. That said, we do, however, still observe that our GDA scheme shows higher robustness than the other four methods as the number of classes increases.

Fig. 6 shows the reliability of 10 sensor nodes that observe the same set of events. For ease of illustration, we sort the sensor nodes in the increasing order of ground truth reliability. As can be seen, the ground truth reliability of these 10 sensor nodes roughly follow a uniform distribution. In Fig. 6, we also show the reliability estimated by our GDA as well as the truth discovery schemes. It is clearly seen that the estimations from our GDA framework follow more closely to the ground truth.

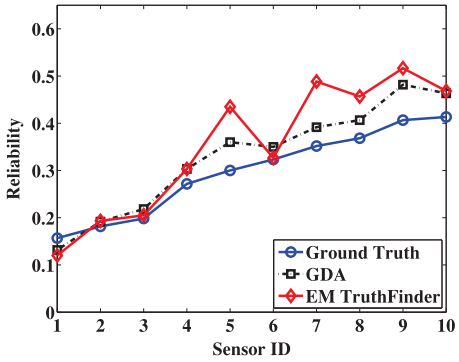
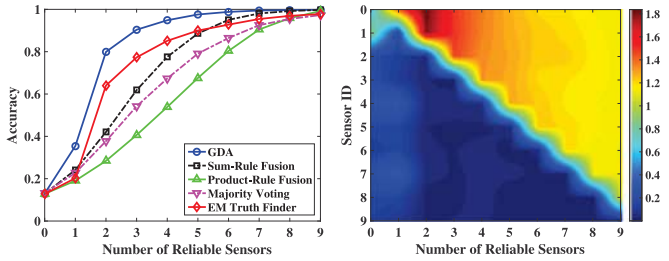


Fig. 6. Reliability measures of 10 sensors observing the same set of events.



(a) Classification Accuracy (b) Learned reliability

Fig. 7. Classification performance and learned reliability of each sensor under different number of reliable sensors on synthetic data.

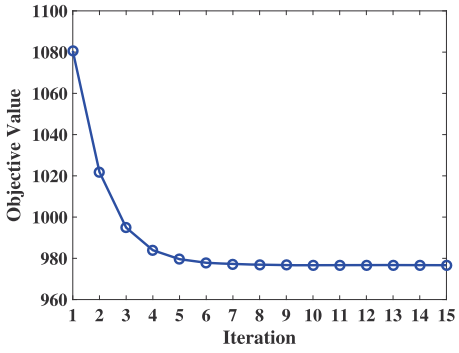


Fig. 8. Convergence.

4) Classification Performance under Varying Number of Reliable Sensors. Furthermore, we investigate how the classification performance of GDA varies when the number of reliable sensors increases. In particular, we fix the number of classes to be 8 and simulate two different types of sensors: reliable sensors (i.e., $r_{ij} = 8c_i$) and unreliable sensors (i.e., $r_{ij} = c_i$). Without loss of generality, we sort the sensors in the descending order of their reliabilities. Then we fix the total number of sensors as 10 and gradually increase the number of reliable sensors.

Fig. 7(a) demonstrates the performance of GDA and the baseline methods. When there is no reliable sensor, none of the approaches can achieve satisfactory accuracy. When the number of reliable sensors increases, the accuracy of GDA improves faster than the baseline approaches because it takes not only sensor reliability but also decision confidences into consideration. Fig. 7(b) further shows the reliability of each sensor learned by GDA. When the number of reliable sensors is extremely low, the learned reliabilities of both reliable sensors and unreliable sensors are low. However, as long as there are a few reliable sensors, GDA is able to distinguish reliable sensors (with warmer color) from unreliable ones (with cooler color), and thus achieve satisfactory accuracy. This is because unreliable sensors, in practice, tend to make mistakes randomly and independently. Therefore, even if the majority of sensors are unreliable, the mistakes will not significantly bias the aggregated decision towards a specific wrong class. On the contrary, all the reliable sensors consistently make the correct decision with high possibility. Therefore, the aggregated decision can be biased towards the correct class with just a few reliable sensors in the system. Since in GDA, sensors that agree with the aggregated decision will get higher reliabilities, GDA will further enhance the weights of reliable sensors and make the reliable sensors lead the aggregated decision. For this reason, although the truly reliable sensors may not be assigned high weights when the number of them is extremely low, the truly unreliable sensors will unlikely be assigned high weights even if there are many of them.

5) Convergence. Next up, we demonstrate the convergence of GDA. Specifically in the experiment we have 600 events equally distributed under 6 classes, where each event is observed by 10 different sensor nodes.

Fig. 8 shows the evolution of the objective value (Eqn. (1)) of $\mathbf{P}$, which denotes the weighted summation of the distances between individual decisions and the aggregated decision. According to the GDA algorithm, the objective value is initialized based on randomly selected aggregated decisions and the resultant sensor reliability. In the subsequent iterations of the algorithm, the objective value is progressively reduced by optimizing the aggregated decisions and the sensor reliability alternatively. As shown in Fig. 8, the objective value converges quickly within just a few iterations.

6) Complexity. Lastly, we look at GDA's computational complexity. We demonstrate that GDA's running time is linear with respect to the number of events under practical settings where, in particular, events are equally distributed under 6 classes with each event being observed by 10 different sensor nodes.

Fig. 9 shows the running time of GDA under different input sizes (i.e., the number of events in each class). As seen, GDA displays linear complexity with respect to the number of events. To further demonstrate this, we compute Pearson's correlation coefficient, a commonly used metric for testing linear relationship between variables. The coefficient ranges between -1 and 1, and the closer it is to 1 (or -1), the stronger the variables are positively (or negatively) linearly correlated. In our experiment, the Pearson's correlation coefficient for running time and the number of events is 0.985, indicating strong positive linear correlation.

5.2.2 Audio Data

We next shift our attention from synthetic data to realistic audio data, using which we examine the classification performance of our GDA framework as well as the aforementioned baseline approaches except the product rule fusion scheme. The product rule fusion scheme suffers when sources give confident but conflicting decision vectors. For example, if two sensors come up with the decision vectors like $(0, 1, 0)$ and $(0.8, 0, 0.2)$ for a particular event, then the resulted vector product would be $(0, 0, 0)$ no matter what decision vectors other sensors may provide, resulting in an undecidable L_i. We found that such cases occur frequently with the real audio data, and thus exclude this baseline from this experiment.

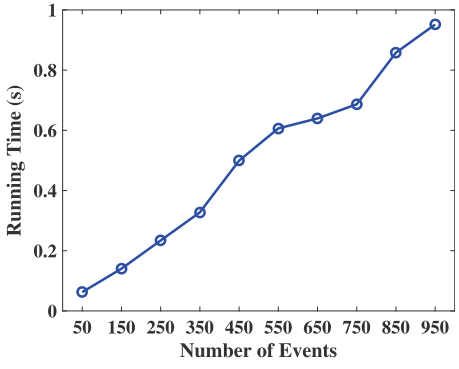
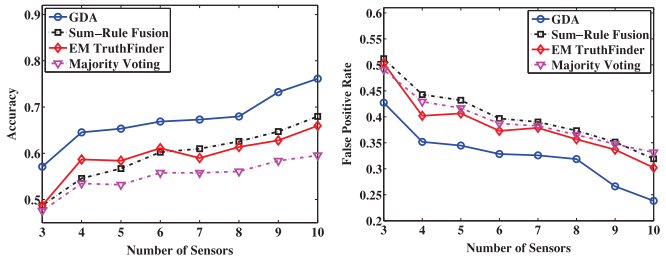


Fig. 9. Running time.



(a) Classification Accuracy (b) False Positive Rate

Fig. 10. Classification performance under different number of sensors on realistic audio data.

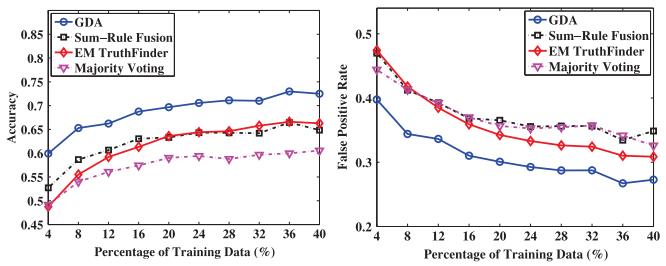
The audio clips we use in this experiment include the sounds of a tank moving, a helicopter flying, and a machine gun firing, corresponding to 3 different classes. We cut the audio clips into pieces with equal time duration, and make a copy for each sensor node. We then add random noise to the sounds received by sensor nodes with various SNRs (Signal-to-Noise Ratios). Next, we extract the MFCC (Mel-Frequency Cepstral Coefficients) features from each audio clip, and feed them as the input to the classification algorithms. In this experiment, we choose random forest as individual sensor node's local classifier. Random forest is a decision tree based classification algorithm that trains multiple decision trees simultaneously and has them vote for the final classification decision. Random forest can output both decision probability vectors and discrete labels (derived from decision probability vectors) that are fed to different approaches under evaluation.

The classification result with varying number of sensor nodes is shown in Fig. 10, which, as seen, is quite similar to that of the experiment on synthetic data shown in Fig. 3. The curves, however, are not as smooth, due to the randomness in the audio sounds themselves. Nevertheless, we can still observe the same general performance trends as displayed in the previous experiment.

Fig. 11 shows the classification performance of studied approaches under varying training data availability levels. We see that the general relative classification effectiveness of all approaches remains the same as all previous experiments, with our GDA framework consistently yielding the best performance. Also, the figure shows that, for all approaches, higher training data availability lead to better classification performance, as expected.

5.3 Experimental Results of Incremental GDA

In this section, we use the aforementioned audio data to compare the performance of incremental GDA (I-GDA) and GDA algorithms. Table 3 summarizes the performance of



(a) Classification Accuracy (b) False Positive Rate

Fig. 11. Classification performance under varying training data availability levels on realistic audio data.

TABLE 3
GDA versus I-GDA

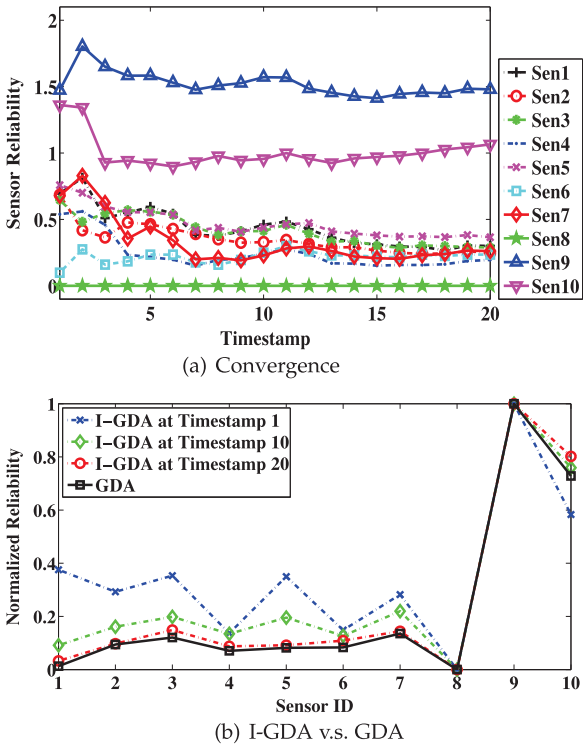
Method	Accuracy	Time (s)
GDA	0.7291	0.7520
I-GDA	0.7123	0.1740

GDA and I-GDA in terms of classification accuracy and running time. We can observe that although the accuracy of I-GDA is slightly lower than that of GDA, it has much better running time than GDA.

In addition, it is also necessary to study the convergence of I-GDA algorithm. Assuming that events come sequentially at constant speed (i.e., one event for each timestamp), we first show the convergence of sensor reliabilities in Fig. 12a. It can be seen that all sensors reach a stable weight value when sufficient events have been accumulated. In order to further compare sensor reliabilities derived by GDA and I-GDA (at timestamp 1, 10, 20), we normalize the reliability of each sensor to the interval $[0, 1]$ through dividing it by the largest weight. The result in Fig. 12b shows that although I-GDA deviates from GDA on sensor reliability estimation at the beginning (i.e., at timestamp 1), their estimates are getting closer over time and finally (i.e., at timestamp 20) become almost the same.

5.4 Experimental Results of Parallel GDA

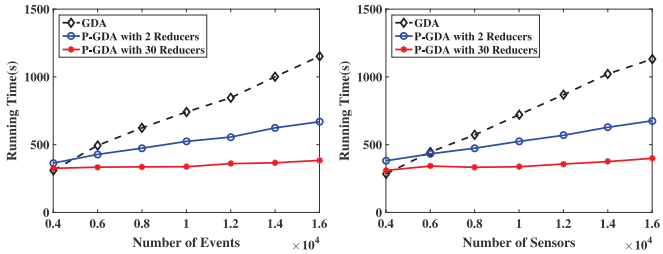
In this section, we evaluate the performance of our proposed parallel GDA (i.e., P-GDA) with regard to different sizes of synthetic data set and different scales of MapReduce system. In order to get a large-scale data set, we generate synthetic datasets with large sensor number and event number. All of our experiments are conducted on a 15-node Amazon EMR cluster. Each node is an m1. large node with 2 vCPU and 7.5 GB of memory. Java



(a) Convergence

(b) I-GDA v.s. GDA

Fig. 12. Convergence of sensor reliability.



(a) w.r.t. Number of Events (b) w.r.t. Number of Sensors

Fig. 13. Running time w.r.t number of events and sensors.

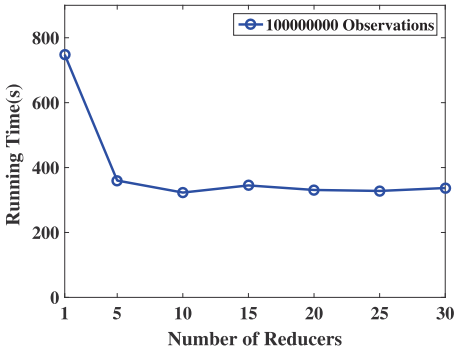


Fig. 14. Running time w.r.t number of reducers.

version 1.7.0_71 and Hadoop version 2.5.1 are used for the MapReduce system.

1) Performance under Varying Size of Data Set. To measure how much longer P-GDA takes on a given system when the data set becomes m times larger, we vary the size of data set by varying either the number of events or the number of sensors. The results are demonstrated in Fig. 13a and Fig. 13b.

In Fig. 13a, we vary event number from 4,000 to 16,000 while fixing the sensor number at 10,000. Correspondingly, in Fig. 13b, we vary sensor number from 4,000 to 16,000 while fixing the event number at 10,000. Thus, in these experiments the input data size ranges from 0.9 GB to 3.5 GB. For both experiments, we plot the running time of P-GDA with 2 and 30 Reducers and compare them with the GDA algorithm. It is not hard to find out that the running time of P-GDA grows much slower than that of the GDA as the number of events or sensors grows. Meanwhile, through calculating Pearson's correlation coefficient for each experiment in Table 4, we find that P-GDA remains GDA's property that running time increases nearly in linear with the size of data. The growth rate of running time, represented as slope of the line, decreases as the number of Reducers increases.

2) Performance under Varying Scale of MapReduce System. Next, we further evaluate how much the proposed parallel GDA can accelerate the GDA algorithm on MapReduce system with different scales. In the experiment, we keep the number of observations unchanged (10^8), and then increase the scale of MapReduce system. In Hadoop, Mapper number is automatically driven by the number of input splits [41] and only Reducer number can be customized. So we increase the scale of MapReduce system by increasing Reducer number from 1 to 30. The results are reported in Fig. 14, from which we can see that the decrease of running time is quick at the beginning, but slows down gradually.

6 Conclusion

In this paper we take a closer look at the decision aggregation problem in distributed sensing systems. Though some efforts have been made towards this problem, the resulting approaches suffer from the limitation of only examining discrete decisions from individual sensor nodes. As a way to avoid high energy cost and network overhead potentially caused by excessively transmitting raw data from sensor nodes, we receive from sensors only decision vectors. Our proposed generalized decision aggregation framework can handle this form of input by thoroughly accounting for and intelligently taking advantage of the decision confidence and reliability of each sensor. In this paper, two extensions of the basic GDA framework, Incremental GDA (I-GDA) and parallel GDA (P-GDA) are developed to deal with streaming data and large-scale data respectively. Through extensive experiments using both synthetic and realistic data, we validate that 1) the GDA framework consistently achieve higher final decision accuracy over the state of the art approaches, 2) the I-GDA algorithm has higher efficiency than GDA with slight accuracy loss, and 3) the P-GDA algorithm is well scalable to large-scale data. We believe the proposed GDA framework and its extensions bear superior generalizability and flexibility, and thus are suitable for a broad spectrum of sensing scenarios.

TABLE 4
Pearson Correlation Coefficient for Events and Sensors

Experiment	for Events	for Sensors
GDA	0.9976	0.9993
P-GDA with 2 Reducers	0.9975	0.9993
P-GDA with 30 Reducers	0.9552	0.9360

Acknowledgments

This work was sponsored in part by US National Science Foundation under grant CNS-1566374, CNS-1652503, IIS-1319973, IIS-1553411.

References

[1] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Mateo, CA, USA: Morgan Kaufmann, 2011.
[2] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.
[3] L. Su, Y. Yang, B. Ding, J. Gao, T. Abdelzaher, and J. Han, "Hierarchical aggregate classification with limited supervision for data reduction in wireless sensor networks," in *Proc. 9th ACM Conf. Embedded Netw. Sensor Syst.*, 2011, pp. 40–53.
[4] L. Su, et al., "Quality of information based data selection and transmission in wireless sensor networks," in *Proc. Real-Time Syst. Symp.*, 2012, pp. 327–338.
[5] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher, "On truth discovery in social sensing: A maximum likelihood estimation approach," in *Proc. ACM/IEEE 11th Int. Conf. Inform. Process. Sensor Netw.*, 2012, pp. 233–244.
[6] S. Wang, D. Wang, L. Su, L. Kaplan, and T. Abdelzaher, "Towards cyber-physical systems in social spaces: The data reliability challenge," in *Proc. IEEE Real-Time Syst. Symp.*, 2014, pp. 74–85.
[7] S. Wang, et al., "Scalable social sensing of interdependent phenomena," in *Proc. 14th Int. Conf. Inform. Process. Sensor Netw.*, 2015, pp. 202–213.
[8] F. Ma, et al., "Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation," in *Proc. 21th ACM SIGKDD Int. Conf. Know. Discovery Data Mining*, 2015, pp. 745–754.

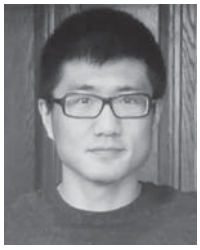
[9] S. Yao, et al., "Recursive ground truth estimator for social data streams," in *Proc. 15th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, 2016, pp. 1–12.
[10] S. Yao, et al., "On source dependency models for reliable social sensing: Algorithms and fundamental error bounds," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst.*, 2016, pp. 467–476.
[11] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.
[12] R. R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.
[13] L. M. Kaplan, S. Chakraborty, and C. Bisdikian, "Fusion of classifiers: A subjective logic perspective," in *Proc. Aerospace Conf.*, 2012, pp. 1–13.
[14] H. Wen, Z. Xiao, A. C. Symington, A. Markham, and N. Trigoni, "Comparison of accuracy estimation approaches for sensor networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Sensor Syst.*, 2013, pp. 28–35.
[15] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C.-W. Yi, "Data fusion improves the coverage of wireless sensor networks," in *Proc. 15th Annu. Int. Conf. Mobile Comput. Netw.*, 2009, pp. 157–168.
[16] R. Tan, G. Xing, Z. Yuan, X. Liu, and J. Yao, "System-level calibration for fusion-based wireless sensor networks," in *Proc. 31st IEEE Real-Time Syst. Symp.*, 2010, pp. 215–224.
[17] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[18] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. 1st ACM Int. Workshop Wireless Sensor Netw. Appl.*, 2002, pp. 88–97.
[19] Y. Guo, P. Corke, G. Poulton, T. Wark, G. Bishop-Hurley, and D. Swain, "Animal behaviour understanding using wireless sensor networks," in *Proc. 31st IEEE Conf. Local Comput. Netw.*, 2006, pp. 607–614.
[20] J. Cai, D. Ee, B. Pham, P. Roe, and J. Zhang, "Sensor network for the monitoring of ecosystem: Bird species recognition," in *Proc. 3rd Int. Conf. Intell. Sensors, Sensor Netw. Inf.*, 2007, pp. 293–298.
[21] W. Hu, V. N. Tran, N. Bulusu, C. T. Chou, S. Jha, and A. Taylor, "The design and evaluation of a hybrid sensor network for cane-toad monitoring," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, 2005, pp. 503–508.
[22] L. Gu, et al., "Lightweight detection and classification for wireless sensor networks in realistic environments," in *Proc. 3rd Int. Conf. Embedded Netw. Sensor Syst.*, 2005, pp. 205–217.
[23] R. R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," *Proce. IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.
[24] X. Cheng, J. Xu, J. Pei, and J. Liu, "Hierarchical distributed data classification in wireless sensor networks," in *Proc. IEEE 6th Int. Conf. Mobile Adhoc Sensor Syst.*, 2009, pp. 10–19.
[25] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *Proc. Int. Conf. Pervasive Comput.*, 2004, pp. 158–175.
[26] K. Lorincz, et al., "Mercury: A wearable sensor network platform for high-fidelity motion analysis," in *Proc. 7th ACM Conf. Embedded Netw. Sensor Syst.*, 2009, pp. 183–196.
[27] Z. Zeng, S. Yu, W. Shin, and J. C. Hou, "PAS: A Wireless-Enabled, Cell-Phone-Incorporated Personal Assistant System for Independent and Assisted Living," in *Proc. 28th Int. Conf. Distrib. Comput. Syst.*, 2008, pp. 233–242.
[28] S. Hu, L. Su, H. Liu, H. Wang, and T. Abdelzaher, "Smartroad: A crowd-sourced traffic regulator detection and identification system," in *Proc. ACM/IEEE Int. Conf. Inf. Proc. Sensor Netw.*, 2013, pp. 331–332.
[29] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the web," in *Proc. 13th ACM SIGKDD Int. Conference Knowl. Discovery Data Mining*, 2007, pp. 1048–1052.
[30] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the web," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 6, pp. 796–808, Jun. 2008.
[31] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2014, pp. 1187–1198.
[32] Y. Li, et al., "Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 1986–1999, Aug. 2016.
[33] S. M. Michael, M. J. Franklin, J. Hellerstein, and W. Hong, "Tag: A tiny aggregation service for ad-hoc sensor networks," in *Proc. 5th Symp. Operating Syst. Des. Implementation*, 2002, pp. 131–146.
[34] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proc. 22nd Int. Conf. Distrib. Comput. Syst.*, 2002, pp. 575–578.
[35] P. K. Varshney, *Distributed Detection and Data Fusion*. New York, NY, USA: Springer, 1997.
[36] T. Clouqueur, K. K. Saluja, and P. Ramanathan, "Fault tolerance in collaborative sensor networks for target detection," *IEEE Trans. Comput.*, vol. 53, no. 3, pp. 320–333, Mar. 2004.
[37] M. Ding, D. Chen, K. Xing, and X. Cheng, "Localized fault-tolerant event boundary detection in sensor networks," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Soc.* 2005, pp. 902–913.
[38] L. Su, et al., "Generalized decision aggregation in distributed sensing systems," in *Proc. IEEE Real-Time Syst. Symp.*, 2014, pp. 1–10.
[39] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Scientific, 1995.
[40] Wikipedia, "Dirichlet distribution," (2017). [Online]. Available: https://en.wikipedia.org/wiki/Dirichlet_distribution
[41] Apache, "Partitioning your job into maps and reduces," (2014). [Online]. Available: http://wiki.apache.org/hadoop/HowManyMapsAndReduces

Wenjun Jiang received the BS and MS degrees from the Department of Computer Science and Engineering, Tsinghua University, China. He is working toward the PhD degree in the Department of Computer Science and Engineering from SUNY Buffalo. His research interests include big data, distributed sensing system, and crowdsourcing.

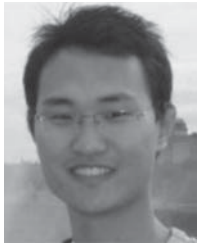
Chenglin Miao received the BS degree from the School of Computer and Information, Hefei University of Technology and the MS degree from the School of Computer Science and Technology, University of Science and Technology of China. He is working toward the PhD degree in the Department of Computer Science and Engineering from SUNY Buffalo. His research interests include crowd and social sensing systems, data mining, security, and privacy.

Lu Su reveived the PhD degree in computer science and the MS degree in statistics from the University of Illinois at Urbana-Champaign, in 2013 and 2012, respectively. He is an assistant professor in the Department of Computer Science and Engineering with SUNY Buffalo. His research focuses on the general areas of cyber-physical systems, wireless and sensor networks, and mobile computing. He has also worked with IBM T. J. Watson Research Center and National Center for Supercomputing Applications. He is a member of the ACM and the IEEE.

Qi Li received the BS degree in mathematics from Xidian University and the MS degree in statistics from the University of Illinois, Urbana-Champaign, in 2010 and 2012, respectively. She is currently working toward the PhD degree in the Department of Computer Science and Engineering, SUNY Buffalo. Her research interests include truth discovery, data aggregation, and crowdsourcing.



Shaohan Hu reveived the MS degree from the Department of Computer Science, Dartmouth College. He is working toward the PhD degree and research assistant at the Department of Computer Science, University of Illinois, Urbana-Champaign. His main research interests lie in designing and building mobile sensing and computing systems.



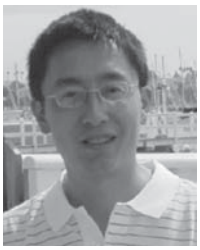
Shiguang Wang is working toward the PhD degree in Computer Science Department, University of Illinois, Urbana-Champaign. His research mainly focuses on the algorithm design and system study in social sensing.

Jing Gao received the PhD degree in Computer Science Department, University of Illinois, Urbana-Champaign, in 2011, and subsequently joined SUNY Buffalo, in 2012. She is an assistant professor in the Department of Computer Science and Engineering, SUNY Buffalo. She is broadly interested in data and information analysis with a focus on truth discovery, information integration, ensemble methods, mining data streams, transfer learning, and anomaly detection. She is a member of the IEEE.

Hengchang Liu received the PhD degree from the University of Virginia, in 2011, under Professor John Stankovic. He is an assistant professor with USTC. His research interests mainly include cyber-physical systems, mobile systems, named data networking, and wireless (sensor) networks.

Tarek F. Abdelzaher received the PhD degree from the University of Michigan, Ann Arbor, in 1999, under professor Kang Shin. He is a professor and Willett faculty scholar in the Department of Computer Science, University of Illinois, Urbana-Champaign. He was an assistant professor with the University of Virginia from August 1999 to August 2005. He then joined the University of Illinois, Urbana-Champaign as an associate professor with tenure, where he became full professor in 2011. His interests lie primarily in systems, including operating systems, networking, sensor networks, distributed systems, and embedded realtime systems. He is especially interested in developing theory, architectural support, and computing abstractions for predictability in software systems, motivated by the increasing software complexity and the growing sources of non-determinism. Applications range from sensor networks to large-scale server farms, and from transportation systems to medicine.

Jiawei Han is Abel Bliss professor in the Department of Computer Science, the University of Illinois. He has been researching into data mining, information network analysis, and database systems, with more than 500 publications. He received the ACM SIGKDD Innovation Award (2004), the IEEE Computer Society Technical Achievement Award (2005), and the IEEE W. Wallace McDowell Award (2009). His book *Data Mining: Concepts and Techniques* (Morgan Kaufmann) has been used worldwide as a textbook. He is a fellow of the ACM and the IEEE.



Xue Liu received the BS degree in mathematics and MS degree in automatic control both from Tsinghua University, China, and the PhD degree in computer science from the University of Illinois, Urbana-Champaign, in 2006. He is an associate professor and William Dawson scholar in the School of Computer Science, McGill University, Montreal, QC, Canada. He was also the Samuel R. Thompson associate professor in the University of Nebraska-Lincoln and HP Labs in Palo Alto, California. His research interests include computer networks and communications, smart grid, real-time and embedded systems, cyber-physical systems, data centers, and software reliability. He has been granted one US patent and filed four other US patents, and published more than 150 research papers in major peer-reviewed international journals and conference proceedings, including the Year 2008 Best Paper Award from the *IEEE Transactions on Industrial Informatics*, and the First Place Best Paper Award of the ACM Conference on Wireless Network Security (WiSec 2011). He is a member of the ACM.

Yan Gao received the PhD degree in computer science from the University of Illinois, Urbana-Champaign. Now he is senior software engineer with Linkedin, where he works on back-end services for big Linkedin profile data. Before joined Linkedin, he has worked with Accenture Tech Labs from 2011-2015, where he focused on big data platform, intelligent transportation systems, and temporal-spatial data monetization.

Lance M. Kaplan received the BS(distinction) degree from Duke University, Durham, North Carolina, in 1989 and the MS and PhD degrees from the University of Southern California, Los Angeles, in 1991 and 1994, respectively, all in electrical engineering. Currently, he is a researcher in the Networked Sensing and Fusion branch of the U.S Army Research Laboratory (ARL). He serves as editor-In-chief for the *IEEE Transactions on Aerospace and Electronic Systems* (AES) and as VP of Conferences for the International Society of Information Fusion (ISIF). Previously, he served on the Board of Governors of the IEEE AES Society (2008-2013) and on the board of directors of ISIF (2012-2014). He is a three time recipient of the Clark Atlanta University Electrical Engineering Instructional Excellence Award from 1999-2001. He is a fellow of the ARL. His current research interests include signal and image processing, information/data fusion, network science, and resource management.