Recursive Ground Truth Estimator for Social Data Streams

Shuochao Yao*, Md Tanvir Amin*, Lu Su†, Shaohan Hu*, Shen Li*, Shiguang Wang*, Yiran Zhao*, Tarek Abdelzaher*, Lance Kaplan‡, Charu Aggarwal§, Aylin Yener¶

*Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801

†Department of Computer Science and Engineering, SUNY at Buffalo, Buffalo, NY 14260

‡Army Research Laboratories, 2800 Powder Mill Road, Adelphi, MD 20783

§IBM Research, 101 Route 134 Kitchawan Road, Yorktown Heights, NY 10598

¶Pennsylvania State University, 121 Electrical Engineering East, University Park, PA 16802

Abstract*—The paper develops a recursive state estimator for social network data streams that allows exploitation of social networks, such as Twitter, as sensor networks to *reliably observe* physical events. Recent literature suggested using social networks as sensor networks leveraging the fact that much of the information upload on the former constitutes acts of sensing. A significant challenge identified in that context was that source reliability is often unknown, leading to uncertainty regarding the veracity of reported observations. Multiple truth finding systems were developed to solve this problem, generally geared towards batch analysis of offline datasets. This work complements the present batch approaches by developing an *online recursive* state estimator that recovers ground truth from *streaming data*. In this paper, we model physical world state by a set of binary signals (propositions, called assertions, about world state) and the social network as a noisy medium, where distortion, fabrication, omissions, and duplication are introduced. Our recursive state estimator is designed to recover the original binary signal (the true propositions) from the received noisy signal, essentially decoding the unreliable social network output to obtain the best estimate of ground truth in the physical world. Results show that the estimator is both effective and efficient at recovering the original signal with a high degree of accuracy. The estimator gives rise to a novel situation awareness tool that can be used for reliably following unfolding events in real time, using dynamically arriving social network data.

I. Introduction

This paper presents the first *recursive* ground truth estimator for arbitrary social data *streams* (as opposed to offline datasets or periodic sensor measurements). It exploits social networks as sensor networks for *reliably observing* the physical world. The authors argue that networked sensing research has a lot to contribute to the analytical foundations of reliability in exploiting social networks for sensing purposes. Indeed, humans are among the most versatile and "widely deployed" sensors, and the emergence of social networks, such as Twitter and Instagram, offers unprecedented opportunities for them to meaningfully share their observations. In fact, they already do so voluntarily, sharing more than 500 million Twitter messages (called tweets) and uploading more than 70 million Instagram images *every day*. This social sensing paradigm [1] extends the model of participatory sensing [2] by viewing information on social networks as an output of virtual participatory sensing campaigns. The paradigm has seen interesting recent application examples, such as detecting Earthquakes with Twitter users [3], observing the pulse of a city with Instagram [4], and estimating characteristics of local businesses using human observers [5].

A key question posed from a sensing perspective is one of attaining reliability. Since sources who contribute the information are often unknown, is it possible to leverage the collective output of such unvetted observers of unknown reliability to correctly filter true observations about the physical world, while suppressing the false ones? This true/false ground truth estimation problem is often called *fact-finding*.

The offline fact-finding problem for batch data has been addressed extensively in multiple communities in recent years, including data mining [6]–[8], machine learning [9], sensor networks [5], [10]–[12], and social computing [13], [14]. While these algorithms work on offline datasets (i.e., where all data is present at once), the area of *online* truth recovery from unvetted social observers has not been covered by existing approaches. The online fact-finding problem is distinguished in that new observation data arrives continually, motivating an online (or recursive) solution that incrementally updates its beliefs about ground truth in view of newly arriving observations. This is in contrast with recomputing such beliefs from scratch, based on the entire augmented dataset, every time new data arrives. The online problem shares with its offline counterpart the fact that (i) the latent variables are categorical, and discrete, denoting assessments of reported observations as true or false, leading to combinatorial problems, (ii) the reliability of sources is not known ahead of time, since anyone can contribute observations without prior vetting, and (iii) sources emit their observations at random times, rather than at fixed sampling intervals as with physical sensors.

In recent sensor networks literature, a binary sensing model was proposed for the offline version of the problem [10], [11]. It views individual tweets as observations, called *claims*, with binary truth values, and abstracts away many natural language processing issues, formulating the core estimation-theoretic problem of recovering original world state from noisy social network observations. The work showed that

978-1-5090-0802-5/16/$31.00 ©2016 IEEE

using simple metrics such as the degree of corroboration (i.e., frequency or number of votes) to determine veracity of individual observations gives *inferior* results, since observers have different reliability, so their votes should have different weights [11]. Hence, to reconstruct what is true in the physical world from reported observations, the correctness of individual observations should be inferred jointly with the reliability of individual sources, as the two problems are intertwined. Most previous work [10], [11] focuses on offline batch algorithms that require all data to be available at once. Therefore these batch algorithms cannot scale to large volumes of streaming data with ease. The first recursive solution to that problem was reported in [15], whose work comes closest to ours. However, their online estimator assumes a periodic sampling model for all sources. This assumption is reasonable for physical sensors that have fixed sampling periods, but is not valid for sources in social networks, such as Twitter, that typically make observations at random times, without coordination. To the best of the authors' knowledge, we offer the first online (recursive) fact-finder that is *applicable to randomly arriving social network data streams* (as opposed to periodic sensors). The estimator combines Bayesian and maximum-a-posteriori estimation techniques to solve the reliability problem for streaming data.

After deriving the estimator, we first evaluate our algorithm in simulation to observe its performance over a broad set of conditions for the input data. We then present case studies, where both the recursive (online) algorithm and the offline algorithm are applied to actual Twitter data feeds to select a subset of tweets as "true". For evaluation purposes, both raw tweets and output tweets are manually inspected by individuals tasked with ascertaining their correctness. The percentage of tweets deemed "correct" (i.e., matching ground truth as discovered by these individuals) are then recorded for each algorithm. Results show that the recursive algorithm described in this paper matches the accuracy of offline algorithms, while reducing processing overhead by an order of magnitude.

The rest of this paper is organized as follows. Section II outlines the model and problem statement. Section III presents the solution overview. The two main components of our solution, namely the recursive and interpolative estimators are described in Section IV and Section V, respectively. Section VI describes evaluation results. Related work is discussed in Section VIII. The paper concludes in Section IX.

II. Model and Problem Statement

Our online estimator adopts a binary sensing model. Recent work on social sensing suggested that human sources are much better at making binary observations than they are at estimating values of continuous variables [11].[1] For example, it is easier to tell if a day is hot or not than it is to estimate the exact value of current temperature. Hence, when scavenging information from social networks, one should focus on binary claims. Table I shows examples of actual statements reported on Twitter, collected in October 2015. Note how these statements are amenable to a categorization into true or false, and hence fit the binary sensing model.

TABLE I: Example Tweets

4.5 magnitude earthquake shakes Cushing, Oklahoma http://fxn.ws/1ZpGzzo.
Russia takes out 55 ISIS sites with 64 bombing attacks in one day. That's the way to do it. http://t.co/7O3HDMYk7y
Russia Teams Up With Islamic State Against Syria's Rebels http://t.co/aPR58Fu0ft
Turkey orders media ban after deadly suicide bombing, internet access reportedly limited http://t.co/qnQ8mKnlCl

While our primary focus is to associate a binary truth-value with each claim, categorical data, such as the color of a bank-robber' escape vehicle, can be converted into a set of binary assertions, one regarding each of the reported colors. Extensions of the binary model to support general categorical variables are therefore straightforward, as demonstrated in prior work [16]. We hence restrict ourselves to binary variables in this paper, as the basic foundation for more general models.

With the above in mind, consider an event in the physical world that is being observed by individuals who voluntarily report aspects of what they see. These individuals collectively constitute the set of sources, S. The statements they make about the physical world collectively constitute the set of assertions, C. We refer to an individual source by $S_i \in S$, and to an individual assertion by $C_j \in C$, where i and j are the source and assertion identifiers, respectively. For example, the statement of each tweet in Table I is an assertion. The same assertion can be made by multiple sources. We call the act of making an assertion by a given source, a *claim*. We say $S_iC_j = 1$ if source S_i claims that C_j occurred, and is zero otherwise. The set of all claims received is denoted by SC. Note that, in this model, the only natural language processing need is to be able to recognize when the same assertion is made by multiple sources. In our implementation, we simply use cosine similarity as a distance metric between tweets, and lump together similar tweets into the same assertion. In practice, we find the approach sufficient for Twitter. Better approaches can be used, but are outside the scope of this paper.

A source might report something they observed first-hand, or might repeat information they heard on the social medium. Some sources may be influenced by other sources. Sources might not report correctly (or not at all) and may make false statements. Prior work [17] described algorithms for inferring the latent influence graph, G, among sources.[2] In this graph, nodes represent different sources and directional edges represent the direction of influence between source pairs, where influence (that may be attributed to persuasion, respect, fear, or desire to imitate) may cause one source to repeat claims espoused by another without verifying their correctness. We say that a claim S_iC_j made by source S_i is *original* if no

[1]Humans can also reliably classify measurements into a small number of distinct categories, but this is a relatively straightforward extension to the binary model.

[2]This graph is computed in an incremental fashion as data arrives.

ancestors of S_i in G made the same assertion, C_j, at an earlier time. In this case, S_i is the original source of assertion C_j. Otherwise, S_i may be acting under influence of another node. In this case, the claim made by S_i is viewed as *dependent*. We use the indicator, D_{ij}, to denote dependent claims. That is to say, $D_{ij} = 0$ indicates that S_i is the original source of C_j. $D_{ij} = 1$ indicates that the claim is dependent. In practice, we can set $D_{ij} = 1$ for retweets. Together these indicators form a dependency matrix, D.

The estimator must determine which assertions are true and which are false. Let $\tau(C_j)$ be the truth value of assertion C_j: $\tau(C_j) = 1$ means C_j is true, and $\tau(C_j) = 0$ means that it is false. Our goal is to determine the truth value $\tau(C_j)$ for each j given all claims, SC, thus far received from the sources and given the dependencies, D.

III. Solution Overview

To solve the problem formulated in the previous section in a recursive manner, we divide time into intervals of size T, and express the solution at the end of each interval, k, in terms of the solution computed at the end of the previous one, $k-1$, as well as the new data that arrived since. This section presents, by example, the intuition behind the recursive solution. The algorithms are described in detail in the following two sections.

If the probability that each source makes correct observations were known exactly, then we could easily determine the likelihood of correctness of each new assertion in the current window, based on the set of sources that made it. In reality, however, our experience with each source may be limited to only a finite number of samples. These samples constitute our prior *belief* in the reliability of the source. For example, we might estimate that a source, Sally, reported 8 observations correctly out of 10, so far. This is called the *prior* (belief). [3]

Assume that, in the next window, k, Sally contributes more observations, whose truth value is assessed by our algorithm. It may be that she made 5 observations of which 2 were believed to be correct by our estimator. Hence, our updated belief in Sally's reliability at the end of window, k, becomes 10 correct observations out of 15. This is called the *posterior* belief. Ideally, the estimate that 2 out 5 of Sally's observations were correct in the current window must be consistent with the *updated* belief in source reliability at the *end* of window, k, not the prior at the beginning of the window. This is because the posterior is more accurate. Yet, to compute that updated belief, one needs to know how many assertions Sally made correctly within the current window. This circular dependency suggests that an iterative approach is needed.

The solution of this problem is the *maximum a posteriori estimator*, combining prior belief with new samples to generate the maximum-likelihood posterior belief and assertion correctness. Our estimator uses *an expectation maximization algorithm* (EM) [18] that runs at the end of each window, combining data that arrived in that window with the prior belief (at the beginning of the window) to generate the posterior belief. The algorithm iteratively updates reliability estimates of sources and correctness estimates of assertions. Once the algorithm converges, the resulting posterior belief becomes prior for the next window. This is a *recursive estimator* because it uses the prior belief for one window to ultimately compute the posterior, which becomes the prior belief, for the subsequent window. Hence, beliefs are computed recursively at window boundaries (together with the probability of correctness of assertions).

Incorporating prior beliefs, the recursive *maximum a posteriori estimator* eliminates the burden of revisiting data in previous windows and significantly reduces the computation cost. However, it still involves an iterative optimization step, preventing us from arbitrarily shrinking window sizes due to restricted computation time when processing streaming data. In order to handle the incoming data that arrives inside the window, we therefore devise an interpolation mechanism that approximates the correctness of assertions at points inside the window, before the window is over. This mechanism simply treats the computed prior as an *exact probability estimate*. In other words, if Sally is estimated to have made 8 out of 10 observations correctly by the start of a window, her reliability is considered to be 80%. This is obviously an empirical probability based approximation, yet it allows us to use the Bayesian equation to estimate the probability of correctness of claims made by these sources before the end of the window. We call this the *interpolative estimator*. It is also computationally efficient with reasonable accuracy for estimating social data stream within each window.

Once the window is over, the recursive estimator computes a new correct belief. The computed belief becomes prior for the next window and the whole process repeats again as new observations arrive in that window. Compared with batch algorithms, the recursive estimator has super-linear complexity, making the estimating process efficient. Next, we derive the recursive and interpolative estimators, respectively.

IV. The Recursive Estimator

Let t_c^k denote the start time of chunk k. Let $\mathcal{SC}^k$ denote the set of claims made in the interval $[t_c^{k-1}, t_c^k)$. They can alternatively be represented by a matrix SC^k, whose dimensions are sources and assertions. Let $\mathcal{S}^k$ and $\mathcal{C}^k$ denote the corresponding set of sources and assertions (that appear in claims made in chunk k), respectively. In order to compute the estimated truth value, $\tau(C_j)$, of each assertion, C_j, we need to determine the reliability of sources. Towards that end, we introduce three parameters that characterize the behavior of source, S_i. These parameters will help determine the estimated truth values of assertions (e.g., the tweets). Specifically, for each source, S_i, we define the following:

- Let $a_i = P(S_iC_j = 1|\tau(C_j) = 1, D_{ij} = 0)$ denote the probability that S_i claims C_j, given that assertion C_j is true and the claim is an original claim.

[3]In the absence of any prior experience with the source, we can set the prior belief to 1 correct observation out of 2, which is the simplest way to say 50%.

- Let $b_i = P(S_iC_j = 1|\tau(C_j) = 0, D_{ij} = 0)$ denote the probability that S_i claims C_j, given that assertion C_j is false, and the claim is an original claim.
- Finally, let $p_{ij} = P(D_{ij} = 1)$ denote the probability that S_i makes a claim that is not original. In other words, the same claim was made earlier by an ancestor of S_i in the influence graph (e.g., using the algorithm in [17]).

In addition, we denote $z = P(\tau(C_j) = 1)$ as the probability that an arbitrary assertion, C_j, is true. It denotes the general prevalence of true statements in the overall assertion set. For notational simplicity, we shall henceforth use the notation, C_j, instead of $\tau(C_j)$, where no ambiguity arises. In other words, we overload the notation C_j to refer to the truth value of the assertion as well, if the meaning is clear from context.

We define vector θ as the vector of source parameters θ_i, to be estimated, where $\theta_i = \{a_i, b_i\}$, termed the source reliability vector. Note that, p_{ij} is not part of that vector because it can be empirically observed. In our implementation, we simply set p_{ij} to the probability that source S_i sends a retweet. The set G of ancestors of S_i in the influence graph is approximated by the set of sources that S_i retweeted. Hence, matrix D simply indicates which claims are retweets and which are not. The recursive estimation proceeds on three stages:

- *Computing mean reliability:* First, the estimator runs an expectation maximization algorithm to jointly estimate source reliability parameters, θ_i, for each source, $S_i \in \mathcal{S}^k$, and probability of correctness, $P(\tau(C_j) = 1|SC^k, D, \theta)$, for each assertion, $C_j \in \mathcal{C}^k$. The inputs to this step are matrix SC^k of observations in window k, the empirically measured matrix D, and the prior beliefs in reliability of each source, S_i. These beliefs are expressed in terms of the number of times a source made or did not make a claim, when the underlying assertion was true, denoted by $H_{a_i}^{k-1}$ and $F_{a_i}^{k-1}$, respectively, as well as the number of times a source made or did not make a claim, when the underlying assertion was false, denoted by $H_{b_i}^{k-1}$ and $F_{b_i}^{k-1}$, respectively. The output is a *mean* estimate of parameters of the reliability vector, θ_i for each source. To compute confidence, we also need an error variance around that mean. This leads to the next step.
- *Computing the error variance:* There are well known results for computing the error variance of a maximum likelihood estimator. With source reliability parameters $\theta_i = \{a_i, b_i\}$ estimated for each source, we use these results to compute the error variance of the estimates.
- *Computing the posterior belief:* With the reliability parameters of each source computed, we can express it in terms of the updated pair of estimates $H_{a_i}^k$, $F_{a_i}^k$, and the updated pair of estimates $H_{b_i}^k$ and $F_{b_i}^k$, representing the updated equivalent number of times that a source made or did not make a claim independently, when the underlying assertion was true and when it was false. They represent our updated belief in source reliability. Each of the aforementioned pairs of numbers is given by a beta distribution, whose mean and variance are a function of the corresponding parameter pair. We thus match the expression for the mean and variance of a beta distribution to the mean and variance computed for a_i and b_i in the preceding steps, generating four equations from which the four unknowns $H_{a_i}^k$, $F_{a_i}^k$, $H_{b_i}^k$, and $F_{b_i}^k$ can be computed (for each source, S_i). They become the new prior (source reliability parameters) for the next window, and the recursion repeats.

The above three steps are discussed in the following three subsections, respectively.

A. Computing Mean Reliability with Maximum a Posteriori Estimation

The maximum likelihood estimate of parameter vector θ can be computed based on the matrix SC^k, collected during time interval $[t_c^{k-1}, t_c^k)$ and the prior θ^{k-1} generated from the last time chunk, $k-1$ (we initialize prior with uniform beta distribution when $k = 1$), as follows:

$$\theta_{MAP} = \arg\max_{\theta} \left\{ \ln \left(P(SC^k|\theta) P(\theta^{k-1}) \right) \right\} \quad (1)$$

where, the probability $P(\theta_i^{k-1})$ is computed from the beta distribution, given $H_{a_i}^{k-1}$, $F_{a_i}^{k-1}$, $H_{b_i}^{k-1}$, and $F_{b_i}^{k-1}$ as follows:

$$P(\theta_i^{k-1}) = \frac{\Gamma(H_{a_i}^{k-1} + F_{a_i}^{k-1} + 2)}{\Gamma(H_{a_i}^{k-1} + 1)\Gamma(F_{a_i}^{k-1} + 1)} \times b_i^{H_{a_i}^{k-1}} (1 - b_i)^{F_{a_i}^{k-1}} \times \frac{\Gamma(H_{b_i}^{k-1} + F_{b_i}^{k-1} + 2)}{\Gamma(H_{b_i}^{k-1} + 1)\Gamma(F_{b_i}^{k-1} + 1)} \times b_i^{H_{b_i}^{k-1}} (1 - b_i)^{F_{b_i}^{k-1}} \quad (2)$$

where $\Gamma(x)$ is the gamma function. The likelihood $\mathcal{L} = P(SC^k|\theta)$ can be formulated as:

$$\begin{aligned} \mathcal{L} = P(SC^k|\theta) &= \prod_{C_j \in \mathcal{C}^k} \sum_{k=0,1} P(SC_j^k, C_j = k|\theta, D) \\ &= \prod_{C_j \in \mathcal{C}^k} \sum_{k=0,1} P(SC_j^k|C_j = k, \theta, D) P(C_j = k|\theta, D) \quad (3) \end{aligned}$$

where SC_j^k is the jth column of matrix SC^k, which lists claims making assertion C_j. In Equation (3), $P(SC_j^k|C_j = k, D, \theta)$ can be expressed as:

$$P(SC_j^k|C_j, D, \theta) = \prod_{S_i \in \mathcal{S}^k} P(S_iC_j|C_j, D_{ij}, \theta) \quad (4)$$

where we have

$$P(S_iC_j|C_j, D_{ij}, \theta) = \begin{cases} a_i & C_j = 1, D_{ij} = 0, \\ & S_iC_j = 1 \\ 1 - a_i & C_j = 1, D_{ij} = 0, \\ & S_iC_j = 0 \\ b_i & C_j = 0, D_{ij} = 0, \\ & S_iC_j = 1 \\ 1 - b_i & C_j = 0, D_{ij} = 0, \\ & S_iC_j = 0 \\ p_{ij} & D_{ij} = 1, S_iC_j = 1 \\ 1 - p_{ij} & D_{ij} = 1, S_iC_j = 0 \end{cases} \quad (5)$$

Combining Equations (1) - (4), and applying the E-step and M-step of standard EM algorithm [18], leads to the following two iteratively computed equations:

$$P(C_j = 1|SC^k\theta^{(r)k}) = \frac{P(SC^k_{[j]}|C_j = 1, D, \theta^{(r)k})P(C = 1)}{\sum_{k=0,1} P(SC^k_{[j]}|C_j = k, D, \theta^{(r)k})P(C = k)} \quad (6)$$

where $P(C = 1) = z^{(r)}_{MAP}$, and

$$a^{(r+1)}_{i,MAP} = \frac{H^{k-1}_{a_i} + \sum_{j\in S_iC^*} P(C_j|SC^k_{[j]})}{A^{k-1}_{a_i} + \sum_{j\in S_iC^* \bigcup \overline{S_iC^*}} P(C_j|SC^k_{[j]})}$$
$$b^{(r+1)}_{i,MAP} = \frac{H^{k-1}_{b_i} + \sum_{j\in S_iC^*} \left(1 - P(C_j|SC^k_{[j]})\right)}{A^{k-1}_{b_i} + \sum_{j\in S_iC^* \bigcup \overline{S_iC^*}} \left(1 - P(C_j|SC^k_{[j]})\right)}$$
$$z^{(r+1)}_{MAP} = \frac{\sum_{C_j\in\mathcal{C}^k} P(C_j|SC^k_{[j]})}{|\mathcal{C}^k|} \quad (7)$$

Here S_iC^* denotes a set of assertions made independently by source i in SC^k, and $\overline{S_iC^*}$ a set of assertions source i does not make in SC^k. $A^{k-1}_{\theta_i}$ denotes $H^{k-1}_{\theta_i} + F^{k-1}_{\theta_i}$. In addition, $P(C_j|SC^k_{[j]})$ denotes $P(C_j = 1|SC^k, \theta^{(r)k})$ in Equation (7). With easy initialization, e.g. $a_i = b_i$, EM algorithm works well without getting into degenerating local optima. Equation (7) solves the problem of computing the expectation of θ^k_i. It remains to determine the confidence in the above estimate, which depends on the error variance.

B. Computing the Error Variance

Using the Cramér Rao lower bound, the variance of any unbiased estimator can be bounded by the inverse of the Fisher information matrix [19]. Fisher information, $\mathcal{I}(\theta)$ measures the amount of information that an observable random variable X carries about an unknown parameter vector θ,

$$\mathcal{I}(\theta)_{1,2} = -E\Big[\frac{\partial^2}{\partial\theta_1\partial\theta_2}\log\mathcal{L}(X;\theta)\Big|\theta\Big] \quad (8)$$

where $\mathcal{L}(X;\theta)$ is the likelihood function. In this paper, $\mathcal{L}(X;\theta) = P(S_iC^k|a_i, b_i)$, which yields the Fisher information matrix for source i a 2×2 symmetric matrix:

$$\mathcal{I}(S_i) = \begin{bmatrix}\mathcal{I}(a_i,a_i) & \mathcal{I}(a_i,b_i)\\ \mathcal{I}(a_i,b_i) & \mathcal{I}(b_i,b_i)\end{bmatrix}$$

where:

$$\mathcal{I}(a_i,a_i) = \sum_{S_iC^*}\left(\frac{P(C_j=1)^2}{P(C_j=1)a_i + P(C_j=0)b_i} + \frac{P(C_j=1)^2}{P(C_j=1)(1-a_i) + P(C_j=0)(1-b_i)}\right)$$
$$\mathcal{I}(b_i,b_i) = \sum_{S_iC^*}\left(\frac{P(C_j=0)^2}{P(C_j=1)a_i + P(C_j=0)b_i} + \frac{P(C_j=0)^2}{P(C_j=1)(1-a_i) + P(C_j=0)(1-b_i)}\right)$$
$$\mathcal{I}(a_i,b_i) = \sum_{S_iC^*}\left(\frac{P(C_j=0)P(C_j=1)}{P(C_j=1)a_i + P(C_j=0)b_i} + \frac{P(C_j=0)P(C_j=1)}{P(C_j=1)(1-a_i) + P(C_j=0)(1-b_i)}\right) \quad (9)$$

Algorithm 1 Recursive Estimator

```
1:  Input: Current time stamp: $t = t^k_c$, beta prior $P(\theta^{k-1}_i)$, and parameters $\{a^{k-1}_i, b^{k-1}_i, z^{k-1}\}$
2:  Update $SC^k$ matrix containing from time interval $[t^{k-1}_c, t^k_c)$
3:  Initialize $\theta^{(0)k}_{MAP}$ and $z^{(0)}_{MAP}$ with the expectation of prior
4:  while $\theta^{(r)k}$ does not converge do
5:      for $C_j$ in $\mathcal{C}^k$ do
6:          Compute probability: $P(C_j|S_iC_j, \theta^{(r+1)k})$ according to (6)
7:      end for
8:      for $S_i$ in $\mathcal{S}^k$ do
9:          Estimate parameters: $a^{(r+1)k}_{i,MAP}$, $b^{(r+1)k}_{i,MAP}$, $z^{(r+1)k}_{MAP}$ according to (7)
10:     end for
11: end while
12: for $S_i$ in $\mathcal{S}^k$ do
13:     Compute elements in Fisher information matrix: $\mathcal{I}(a_i,a_i)$, $\mathcal{I}(b_i,b_i)$, $\mathcal{I}(a_i,b_i)$ according to (9)
14:     Compute variance: $\hat{v_{a_i}}$, $\hat{v_{b_i}}$ according to (10)
15:     Compute parameters of prior distribution: $H^k_{a_i}$, $H^k_{b_i}$, $F^k_{a_i}$, and $F^k_{b_i}$ according to (14) (15) (16) (17)
16: end for
17: Output: $P(C_j|S_iC_j, \theta^{(r)k})$, $a^{(r)k}_{i,MAP}$, $b^{(r)k}_{i,MAP}$, $z^{(r)k}_{,MAP}$, $H^k_{a_i}$, $H^k_{b_i}$, $F^k_{a_i}$, and $F^k_{b_i}$ for future estimation
18: END.
```

The inverse of Fisher information matrix, $\mathcal{I}(S_i)^{-1}$, yields the approximated covariance matrix $cov(a_i, b_i)$. The diagonal elements of the covariance matrix are the variance of parameter a_i and b_i we wish to obtain. Therefore $var(a_i)$ and $var(b_i)$ can be approximated as:

$$var(a_i) \approx \hat{v_{a_i}} = \frac{\mathcal{I}(b_i,b_i)}{\mathcal{I}(a_i,a_i)\mathcal{I}(b_i,b_i) - \mathcal{I}(a_i,b_i)^2}$$
$$var(b_i) \approx \hat{v_{b_i}} = \frac{\mathcal{I}(a_i,a_i)}{\mathcal{I}(a_i,a_i)\mathcal{I}(b_i,b_i) - \mathcal{I}(a_i,b_i)^2} \quad (10)$$

The above variance, together with expected values computed in Equation (7), give us what we need to compute the posterior belief.

C. Computing the Posterior Belief

To complete the recursion, it remains to show how to update our estimates of the equivalent number of times that a source independently makes or does not make a claim when the underlying assertion is true (i.e., $H^k_{a_i}$ and $F^k_{a_i}$), and when it is false (i.e., $H^k_{b_i}$ and $F^k_{b_i}$), respectively. Let the expectation and variance for parameter θ_i, computed by the recursive estimator at time t^k_c, be denoted by $\hat{\theta}^k_i$ and $\hat{v_\theta}^k_i$, respectively. First, the probability mass function for seeing $H^k_{a_i}$ claims and $F^k_{a_i}$ silences from source S_i, regarding true assertions, is the Beta distribution,

$$f(H^k_{a_i}, F^k_{a_i}, a_i) = \frac{\Gamma(H^k_{a_i} + F^k_{a_i} + 2)}{\Gamma(H^k_{a_i} + 1)\Gamma(F^k_{a_i} + 1)} {a_i}^{H^k_{a_i}} (1 - a_i)^{F^k_{a_i}} \quad (11)$$

whose expectation and variance are known to be:

$$\hat{a}_i^k = \frac{H_{a_i}^k + 1}{H_{a_i}^k + F_{a_i}^k + 2} \tag{12}$$

$$\hat{v_a}_i^k = \frac{(H_{a_i}^k + 1)(F_{a_i}^k + 1)}{(H_{a_i}^k + F_{a_i}^k + 2)^2(H_{a_i}^k + F_{a_i}^k + 3)} \tag{13}$$

From Equations (12) and (13) we can solve for $H_{a_i}^k$ and $F_{a_i}^k$, that match our prior confidence in source reliability:

$$H_{a_i}^k = \frac{(1 - \hat{a}_i^k)(\hat{a}_i^k)^2}{\hat{v_{a_i}}^k} - 1 - \hat{a}_i^k \tag{14}$$

$$F_{a_i}^k = \frac{(1 - \hat{a}_i^k)H_{a_i}^k + 1 - 2\hat{a}_i^k}{\hat{a}_i^k} \tag{15}$$

Similarly, we can derive:

$$H_{b_i}^k = \frac{(1 - \hat{b}_i^k)(\hat{b}_i^k)^2}{\hat{v_{b_i}}^k} - 1 - \hat{b}_i^k \tag{16}$$

$$F_{b_i}^k = \frac{(1 - \hat{b}_i^k)H_{b_i}^k + 1 - 2\hat{b}_i^k}{\hat{b}_i^k} \tag{17}$$

This completes the recursion and we are ready for the next window. The pseudocode for the maximum a posteriori estimator is shown in Algorithm 1.

V. Interpolative Estimator

The recursive estimator estimates assertion correctness only at window boundaries, t_c^k. We can activate the interpolative estimator to estimate these parameters at any time, *inside a window*, based on the prior provided by the latest available result of the recursive estimator, plus the partial information received so far in the current window. In this paper, we use a Bayesian Estimator as our interpolative estimator. Let t_c^l be the time that the latest-finished recursive estimator ran. Consider time $t_c^l + t$. Then, we have

$$P(C_j^{l+t}|SC^{l+t}, D, \theta^t) = \frac{P(SC_j^{l+t}|C_j = 1, D, \theta^l)P(C_j^l = 1)}{\sum_{k=0,1} P(SC_j^{l+t}|C_j = k, D, \theta^l)P(C_j^l = k)} \tag{18}$$

where $P(C_j^l = k)$, $k = \{0, 1\}$, is a prior information provided by Equation (6) according to the latest finished recursive estimator at time chunk l. $P(SC_j^{l+t}|C_j = k, D, \theta^l)$, $k = \{0, 1\}$, is calculated according to Equation (4).

If the latest finished recursive estimator is executed at time t_c^l, SC^{l+t} matrix contain all the data generated from the time interval $[t_c^l, t_c^l + t)$. Therefore this interpolative estimator will try to obtain the information in need from the SC^{l+t} matrix and the prior $P(C_j^l = k)$, $k = \{0, 1\}$, respectively.

VI. Evaluation

In this section, we evaluate our recursive estimation algorithm using synthetic data as well as an empirical Twitter-based study. We compare our algorithm to various baselines and demonstrate that it outperforms the state of the art in both speed and accuracy.

A. Testing with Synthetic Data

In order to evaluate the algorithm under a large variety of conditions, we first developed a synthetic data generator that produces a synthetic data trace on fictional events. The synthetic data generator is parameterized to produce claims from n_f sources, $\{S_1, \cdots, S_{n_f}\}$, collectively making m_f different assertions, $\{C_1, \cdots, C_{m_f}\}$, in the trace. Other parameters decide the ratio of true to false assertions in the trace, as well as total trace length.

The data generator operates as follows. First, it generates the requisite number of assertions and colors them into a true pool and false pool meeting the desired true-to-false ratio. For each assertion C_j, a *life-span* is chosen to indicate the time interval within which the assertion *can* be made by some sources. To generate the synthetic claims, the generator executes a loop for a fixed number of iterations. In each iteration, each source, S_i, participates with some controlled probability p_i^{on}. If participating, then another controlled probability, p_i^{depend}, decides if the source is going to make a retweet (i.e., a dependent claim) or not. If yes, a retweet of a previous claim is generated. Otherwise, the source makes an original claim, in which case another probability p_i^{true} is used to determine the true/false value of the assertion. Depending on the outcome, we pick one assertion from the true or false pool accordingly. The outcome of this process is a synthetic trace of claims made by a set of sources over a specified period. The trace reflects a different degree of participation (thanks to p_i^{on}), a different level of dependency on other sources (thanks to p_i^{depend}), and a different reliability (thanks to p_i^{true}). In our experiments, we generate a 10-hour trace. We then break it into 1 hour time-chunks and feed it to the algorithms being compared. We compare seven algorithms from different communities:

- *Recursive:* It refers to the new recursive estimator described in this paper.
- *Voting:* This algorithm ranks assertions according to the total number of times the assertion was made (e.g., total number of tweets making the same statement). The larger this number, the more credence is given to the assertion. This is a classic algorithm for processing streaming data, some variants can use additional window to shrink the data size or use approximation methods [20] [21].
- *EM (IPSN 2012):* This is the original offline algorithm for joint estimation of reliability of sources and correctness of claims, described in [10]. It improves upon voting by trying to infer and account for the reliability of sources, such that different sources are given different weights when evaluating the correctness of claims.
- *EM Social (IPSN 2014):* This is an offline algorithm [11], improved upon EM (IPSN 2012). It takes into account dependencies between sources. Hence, it is less vulnerable to false rumors that may give rise to a large number of claims that make the same assertion, all originating from the same source and are simply repeated by others.
- *Sums:* An iterative algorithm [22] that estimates the reliability of assertions and sources in turn by counting

the number of sources and assertions that support them.

- *Average.Log:* This is a variant of Sums algorithm [22]. It makes a tradeoff to trust more on sources who make more true assertions. During each iteration, source reliability is weighted by claims it has made.
- *Truth-Finder:* This is still an iterative algorithm [23]. It utilizes the interdependency between source trustworthiness and assertion confidence to find trustworthy sources and true assertions.
- *Bulk:* This is EM Social (IPSN 2014) running on all data at once. It represents a fully batched algorithm that considers all data collected. It is "optimal" in the sense of being a maximum-likelihood estimate computed based on *all* data.

Since the iterative algorithm, Sums, Average.Log, and Truth-Finder, as well as Voting has only the function of ranking, We "cheat", in their favor, by providing them with actual number of true assertions, T_k, for each time step. Then the $top - T_k$ ranking of these four algorithms are regarded as the output of true assertions at each time stamp.

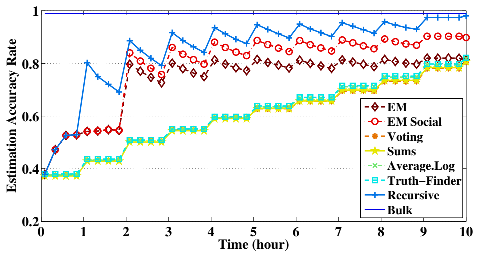

Fig. 1: Accuracy of Different Algorithms.

Our goal is to assess the advantages of recursive estimation. Specifically, while the recursive estimator uses a similar expectation maximization framework as the offline algorithm, it is designed to carry information from one time chunk to another via priors. The absence of a prior in previous offline algorithms requires them to work either on all data at once (which is eventually computationally prohibitive for streaming data) or work on smaller sliding windows, in which case they simply forget data that precedes the current window. As mentioned earlier in the description of our system, we use a window of 1 hour for all the recursive, iterative and batch algorithms. For purposes of this evaluation, we update the beliefs of batch algorithms by running them every 5 minutes on a sliding window of the preceding one-hour worth of data. The recursive MAP estimator, in contrast, is applied once an hour to *non-overlapping* one-hour windows, and we use interpolation in between to generate an output every 5 minutes. Priors are used to carry beliefs across MAP windows. The bulk algorithm was applied once to all 10-hours worth of data. The accuracy of different algorithms was then compared. Accuracy was defined as the percentage of correctly-classified assertions.

To evaluate accuracy, we need to address the issue that the same assertion might be classified differently in different windows, as different information is collected over time, resulting in updated beliefs. Hence, we need to decide which belief to use. Two methods are compared, labeled *Max* and *Final*. In the *Max* method, we regard an assertion, C_j, to be true according to an algorithm, if the algorithm considered it true in any chunk. The rationale lies in that old assertions eventually lose corroboration, as sources' attention moves to new events and may thus be later misclassified as false, especially by algorithms that forget the past. In contrast, *Final* shows the truth value of the assertion as determined by an algorithm in the last time window where the assertion appeared. For all the simulation results, *Max* method shows better performance. We therefore omit *Final* for the remainder of the evaluation for ease and clarity of presentation.

We conduct 50 independent experiments. Fig. 1 illustrates estimation accuracy (i.e., the percentage of correctly classified assertions) for all algorithms during the synthetically generated 10-hour trace. For data generation, we use, $n_f = 50$, $m_f = 250$, $p^{on} = 0.6$, $p^{true} = 0.8$, *minimum life-span* $= 1h$, *maximum life-span* $= 3h$. From Fig. 1, we can see that the recursive estimation algorithm ranks top for estimation accuracy. It does so, while reducing overhead thanks to the use of interpolation within a window.

Next, we evaluate the effect of changing different data set generation parameters. The results below are collected when the entire 10-hour data trace has been consumed by each of the algorithms compared. In the first experiment, we change the probability, p^{on}, that a source contributes a claim, while keeping other parameters fixed. p^{on} varies from 0.2 to 0.7 with 0.1 increments. From Fig. 2, we see that the recursive algorithm has the best performance of all algorithms except the bulk, under various settings. The bulk algorithm achieves higher accuracies, but it works on all data at once, which is eventually computationally intractable. Voting does the worst, demonstrating that relying on the number of sources who make the same assertion to assess veracity is not a good policy. This is true for two reasons. First, sources have different reliability. Hence, we cannot simply count votes. Second, some sources are not independent (e.g., they simply retweet others). Such sources do not add much to the level of data corroboration. In addition, voting and the three iterative algorithms illustrate quite large variance for all cases.

In the second experiment, we change the total number of sources in the data set, n_f, while keeping other parameters fixed. The number of sources, n_f, is changed from 50 to 550 with increments of 100. From Fig. 3, we see that adding more sources improves the performance of our recursive algorithm. This is good since the number of sources in real Twitter data sets is generally large. Again, we outperform all algorithms except the bulk. But an interesting observation is that the performance of two batch algorithms and three iterative algorithms degrades after the number of sources exceed a certain threshold, which is a bit counter-intuitive. After examining the result of simulation, we find that these algorithms reach the maximum iteration threshold used to make sure that iterative algorithms do terminate within reasonable time limits. Therefore the parameters in these algorithms are not well estimated. The reason why the recursive algorithm is not affected is that

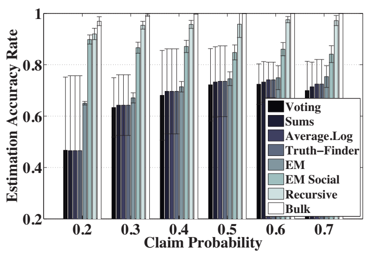

Fig. 2: Vary Claim Prob.

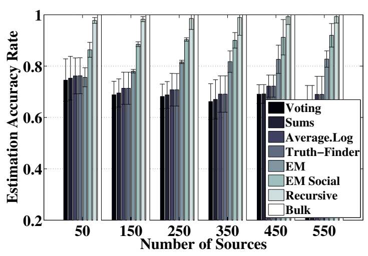

Fig. 3: Vary # of Source.

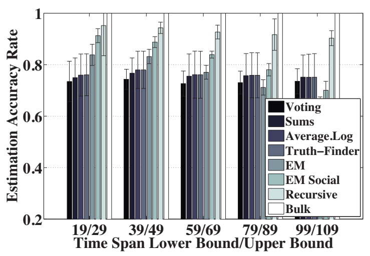

Fig. 4: Vary Life-Span.

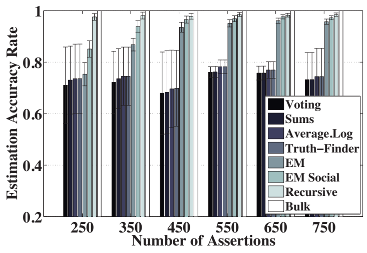

Fig. 5: Vary # of Assertion.

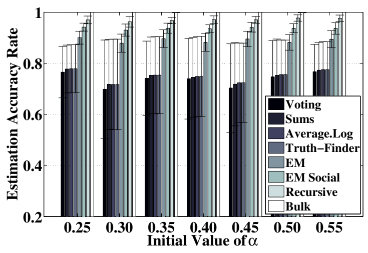

Fig. 6: Vary the initialization of a.

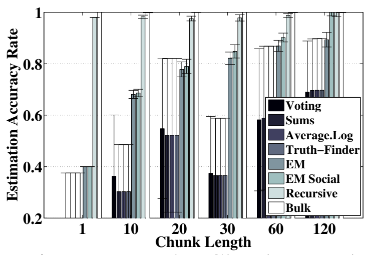

Fig. 7: Vary the Chunk Length.

it incorporates former estimations as prior knowledge, making the estimation converge much faster.

In the third experiment, we change the minimum and maximum length of life-span for the generated assertions, while keeping other parameters fixed. Results are shown in Fig. 4. The horizontal axis shows the min/max life-span selected in multiples of 5 minutes. We see that when span is large, the performance of different algorithms tend to converge. This is because claim patterns becomes more stationary. Hence, there is not much difference between past and current windows, thus reducing the penalty for forgetting past windows and improving accuracy of algorithms that operate on sliding windows without the benefit of a prior. A side-effect of making the span longer is that more assertions linger around at any given time. In our synthetic data trace generator, this translates into fewer claims per assertion (and hence a lower degree of corroboration). Hence, our recursive estimator suffers slightly.

In the forth experiment, we change the number of assertions, m_f, while keeping other parameters fixed. We can see from Fig. 5 that our algorithm does very well overall, exceeded only by the bulk.

In the fifth experiment, we change the initialization of the algorithm, a_i and b_i. We set the ground truth as $a_i = 0.6$ and $b = 0.25$. During this simulation, we set the initialisation value of $b_i = 0.25$ and change a_i from 0.25 to 0.55 with increments of 0.05. Results are shown in Fig 6. Actually it's the ratio of a_i/b_i that really matters. Even under the worst initialization $a_i/b_i = 1$, all the algorithm can achieve a satisfactory performance. Therefore $a_i = b_i$ can be chosen as straightforward initialization in practice.

Lastly, we experimentally test chunk lengths in the set [1, 10, 20, 30, 60, 120] (there are 120 slots in total). When chunk length is small, all batch algorithms perform badly. When chunk length becomes larger, batch algorithms perform better but with longer computation time.

B. Comparing Computation Times

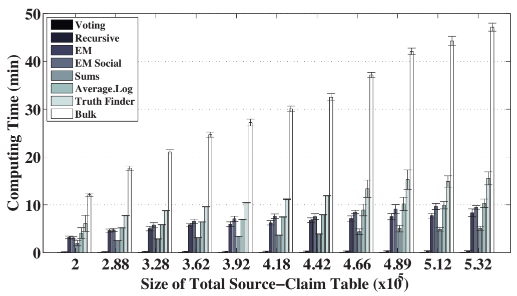

Fig. 8: Computation Time Comparison.

Here we compare the computation times of the recursive estimator to that of other algorithms. We then measure the time needed to finish processing the synthetic 10 hour data stream. We conduct several experiments, changing the total number of tweets made in the 10 hour interval. We make the source and claim both grow sub-linearly, i.e. $n_t = 200 + 40\sqrt{t}$ and $m_t = 1000 + 200\sqrt{t}$, which is equivalent to a middle-size twitter dataset. We average results of 20 independent runs for each data point. Results are shown in Fig. 8. The recursive algorithm is shown to run much faster than all other algorithms except voting. The bulk algorithm is the slowest, exceeding the run-time of the recursive algorithm by nearly two orders of magnitude. The running time of batch and iterative algorithm grows linearly as the size of source-claim table, and it grows quadratically when both sources and assertions grow linearly.

C. Empirical Evaluation

In this section, we report experiences with using our recursive algorithm on four data collection campaigns on Twitter, collected in March 2015. The campaigns were created using the tool we developed for data collection and cleaning. Each data set was collected by specifying three keywords and a geographic location. The corresponding data collection task would then collect tweets that contain the indicated keywords or originate from the indicated location. Table VI-C summarizes data collected by the four tasks we created, labeled (i) *Ukraine*, (ii) *Kirkuk*, (iii) *Superbug*, and (iv) *LA Marathon*. These tasks collected data on the (i) Russia/Ukraine crisis, (ii) a battle with ISIS in the Iraqi city of Kirkuk, (iii) the outbreak of a deadly virus in a UCLA hospital, and (iv) the recent marathon in LA, respectively. These data sets are available for download from (blinded URL for anonymity).

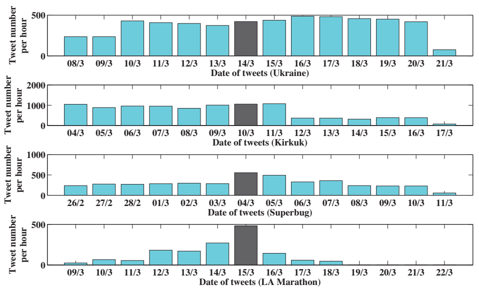

Fig. 9: #Tweets/hour Generated for Each Day

The tool collected and ranked the credibility of tweets over time. Both a real-time view and an archival view of top ranked tweets over time is made available to users. These can also be retrieved from the above URL. For evaluation purposes in this paper, we show results from a 10-hour interval only. The 10-hour duration, in each case, starts at noon local time. For the date, we chose dates when interesting events occurred, as follows:

- Ukraine: We show results from March 14th 2015. On that date, the one year anniversary of Russian annexation of Crimea was approaching. New physical structures were being erected on Moscow's Red Square. The Russian President Vladimir V. Putin had not been seen in public for more than a week. He had canceled a trip to Kazakhstan and postponed a treaty signing with representatives from South Ossetia. Speculations were popping up in the news and social media on possible reasons offering a good noisy environment to test the state estimators. Contrary to some of the rumors, of course, the Russian president was alive. The reason for the structures on Red Square also became clear later.
- Kirkuk: We show results from March 10th 2015. On that day, Kurdish forces in Northern Iraq attacked the self-proclaimed "Islamic State" (ISIS) outpost west of the city of Kirkuk. Battles raged and a lot of commentary followed on social media about state of affairs on the ground.
- Superbug: We show results from Mar 4th 2015. This date was in the middle of a crisis resulted from the spread of a deadly virus among patients of a UCLA hospital.
- LA Marathon: We show results from Mar 15th, the day of a the Los Angeles 2015 Marathon. The runners started from Dodger Stadium and ran a course to the coastline in Santa Monica, passing through downtown Los Angeles, Hollywood, the Sunset Strip, Santa Monica Blvd, and Ocean Avenue, among other landmarks. Much Twitter activity followed the event as it unfolded.

Fig.9 shows the tweets generated per hour on each of the above events over a two week window around the selected evaluation days (shown in figure with a darker color). In the evaluation, we compare our novel recursive algorithm to the four baselines mentioned earlier, each running on 1 hour data chunks for a total data span of 10 hours. For the bulk, we could not run the batch algorithm on all data at once because it is prohibitively large, which is precisely the reason we invested in developing a recursive one in the first place. Instead we ran it on a chunk of 24 hours worth of data ending at the same time as the aforementioned 10 hour interval.

To evaluate accuracy, the top-20 most trustworthy tweets (i.e., those with the highest probability of correctness) identified by each algorithm were collected from each of 10 non-overlapping 1 hour windows, generating 200 most credible tweets. We also obtained the top-100 tweets from the bulk algorithm. We then merged all these tweets from the different algorithms into a single file (where the name of the generating algorithm was anonymized), and manually graded the file by human graders. Since the algorithms were anonymized, the grader did not know which algorithm generated which output to prevent bias. The grader was responsible to do background research on each tweet as needed to be able to mark it as "True", "False", or "Opinion" according to the following rule:

- True: Tweets making a verifiable assertion that was confirmed to be true by the grader
- False: Tweets making a verifiable assertion that was confirmed to be false by the grader.
- Opinion: Tweets making a subjective assessment such as "President Arthur is good" or tweets that do not constitute an act of sensing (e.g., "Please support dolphins in Australia").

The algorithms were then de-anonymized, and we computed the percentage of assertions found True in the output of each algorithm. That is to say, we computed the ratio #True/(#True + #False + #Opinion).

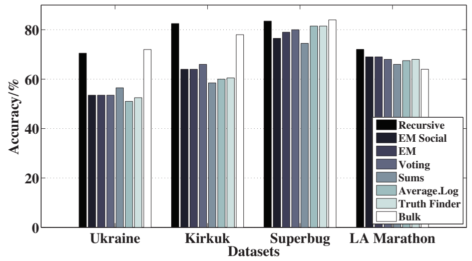

Fig. 10: Empirical Accuracy Results

The evaluation accuracy of each algorithm is shown in Fig. 10. In general, the recursive algorithm performs better than batch algorithms. It often does as well or even better than

TABLE II: Information Summary of Twitter Datasets.

	Total Start Time (UTC)	Total End Time (UTC)	Evaluation Day	#Assertions	#Sources	#Total Claims	#Original Claims	Locations
Ukraine	Feb 20 12:15:28 2015	Mar 31 23:10:12 2015	Mar 14 2015	7298	5403	7192	4242	Ukraine
Kirkuk	Jan 31 01:47:25 2015	Jan 31 01:47:25 2015	Mar 10 2015	6172	4816	6188	3079	Kirkuk
Superbug	Feb 19 17:42:39 2015	Apr 09 18:29:01 2015	Mar 4 2015	9891	7764	9426	5831	LA
LA Marathon	Mar 12 01:38:29 2015	Mar 18 02:14:42 2015	Mar 15 2015	7872	5174	7148	4332	LA

the bulk algorithm. Note that, the bulk in this case ran only on the last 24 hours worth of data, as opposed to the entire trace. Hence, technically optimality was lost. Ours on the other hand, passed beliefs from one window to the next since the beginning of data collection, which caused it sometimes to outperform the bulk. This is desirable in terms of scalability, considering that we were orders of magnitude faster.

Let us now take a deeper look into Fig. 10. The basic EM algorithm (IPSN 2012) performs roughly the same as the voting baseline. This is different from the evaluation shown in prior work [11]. The reason for this phenomenon is that we ran it on chunks that are only one hour long. In contrast, prior work used the algorithm on much larger chunks of data [10], [11], similar to the bulk method in our previous experiments. The batch approach becomes computationally infeasible quickly as the data size increases.

The reason why the EM Social algorithm (IPSN 2014) can hardly outperform the regular EM (IPSN 2012) is similar. The EM Social does not have enough data to correctly compute the source dependencies, which is needed for improving the estimation accuracy during a single time chunk.

Three iterative algorithms: Sums, Average.Log, and Truth Finder perform with high variance, which we have already observed in our simulation cases. In different datasets, they sometimes perform better than EM and Social EM algorithms, but sometimes not.

The bulk algorithm (EM Social running on 24 hours worth of data) is a strong competitor, although the recursive method beats it in two twitter datasets, “Kirkuk” and “LA Marathon”. We looked into the data to see why. For the “Kirkuk” dataset, we found that there is a single user who consistently tweeted rumors. The recursive algorithm ruled out this “bad” user successfully, whereas the bulk did not. This is because the latter ran on a 24 hour window only. In contrast, ours ran in a streaming fashion, and passed beliefs in source reliability from hour to hour. These beliefs therefore accumulated since the beginning of data collection, resulting in more accurate assessment of sources. A similar reason applied to the “LA Marathon” dataset, the other case where we beat the bulk.

Finally, we perform a comparison of execution time on the real datasets. We compare the computation time of each algorithm when ingesting data from a window of size that varies from 1 hour to 10 hours. Batch algorithms run on the entire window at once. The recursive algorithm runs once on each 1 hour of data, passing the results as a prior to the next. The evaluation result on the four real data sets is shown in Fig. 11 - Fig. 14. The differences in speed between the recursive algorithm and the baselines are not as stark as in the results presented for synthetic data. This is due to the existence of the added clustering phase to pull tweets that make the same assertion together, before proceeding with analysis. However, as the data grows larger, the difference between batch processing and recursive estimation becomes increasingly more prominent.

VII. Discussion

Several items regarding the approach in this paper deserve further discussion. First, while the technique does not interpret the content of tweets and hence minimizes the use of natural language processing, it does not eliminate the need for language understanding by the user. The user would still need to frame a meaningful query to collect the tweets and the user will need to interpret the tweets deemed correct by the system. Eliminating the human is not an objective of this system.

The approach treats inputs as binary in the sense that they are either true or false. In many applications data is categorical and the number of categories exceeds two. For example, the color of a car might be one of a large set of labels. The easiest way to extend this system to handle more general categorical data is to view each label as independent true/false proposition, although exact solutions have also been proposed [16].

Collusion is another challenge that needs to be accounted for. It is easy to detect groups of collaborators if their answers are similarly incorrect and the ground truth is already known, for example in an in-class exam. The challenge in collusion, however, is whether one can detect such groups even when ground truth is not known? The intuition why that is possible lies in one key difference between the classroom exam setting (where everyone gets to answer the same questions) and Twitter (where different individuals typically comment on different issues - more like choosing to “answer” a small subset of “questions” from a very large set). Groups who often choose to comment on the same issues and whose comments on those issues coincide are akin to those who coincidentally choose to answer largely the same questions (from the big set) and answer them the same way. One approach to systematically detect such correlations over time was presented in earlier literature [17] and is used by our system.

A separate issue is whether the system can be fooled by a source, or a set of sources, who gradually gain trust by offering true observations for a while, then use it to disseminate incorrect information. This is a common mode of failure in reputation systems. It can be controlled in our system by deciding how much accumulated confidence in the reliability of a source to pass from one window to another. This confidence value is inherited from one window to another measured in (virtual) samples, referred to in the paper as the *beta prior*. By upper-bounding the number of (virtual) samples that quantify confidence in the computed degree of reliability,

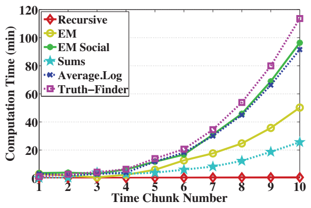
Fig. 11: Kirkuk.

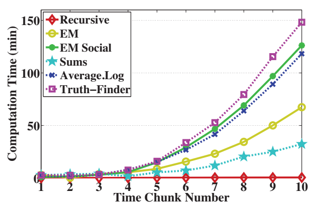
Fig. 12: Ukraine.

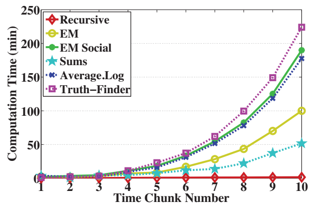
Fig. 13: SuperBug.

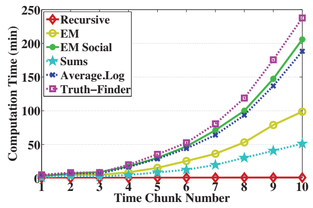
Fig. 14: LA Marathon.

it is possible to reduce the ability of individuals to accumulate significant trust.

The discussion in the current paper focused on text (tweets). In reality, the algorithms presented are quite agnostic to the nature of content. The foundation for truth estimation is the source-claim graph that presents who made which claim(s). It does not interpret the claims themselves. To generate this graph, we simply need to understand when two sources claim the same "thing" so that both of them get connected to the same claim node. Claims could be any data types as long as one can define a distance function between objects of that data type. For example, consider an application that aims to understand locations of damage after an earthquake by asking volunteers to take pictures of damaged areas and buildings. In this case, the "distance function" might simply be the physical distance between locations of taken images. Pictures taken roughly in the same place will thus be clustered together into the same claim, and the resulting source-claim graph subjected to exactly the same mathematical treatment as presented in this paper to determine the true/false values of claims (i.e., to determine which of the reported locations are likely to have real damage).

Finally, it is possible to combine output of humans acting as sensors with output of real sensors. The source-claim graph does not need to know who or what the source is. For example, in the post-disaster damage survey scenario mentioned above, it is equally acceptable for a sensor to report its location when it senses that damage has occurred (e.g., a structure was compromised). It also is possible to initialize trusted sensors with the appropriate value of reliability and confidence as well as exempt them, if so desired, from the limits (discussed above) applied to prevent accumulation of trust and vulnerability to subsequent deceit.

VIII. Related Work

Social sensing emerged as a key topic in sensor networks research that attracted much attention. It is a type of participatory sensing, initially introduced in Burke et al. [24]. Examples of early services include CenWits [25], CarTel [26], BikeNet [27]. Application-specific redundancy-eliminating sensing services, such as PhotoNet [28], and CARE [29].

Social sensing is enabled mainly by the proliferation of mobile sensors and smart devices held by people (such as smartphones) and the popularity of massive social media (such as Twitter, Facebook, and Instagram). These provide an environment that makes social sensing possible thanks to increased connectivity and capacity for sharing. Social sensing applications can thus be viewed as those, where people act as sensor carriers [30] (e.g., opportunistic sensing), sensor operators(e.g., participatory sensing) [24] or sensor themselves [10], [11]. In this paper, we focus on humans acting sensors.

Data quality and trustworthiness is a key problem for social sensing. Humans can easily introduce noise into sensing data. They may not report correctly (or not at all) and may make false claims. Recent work focused on estimating reliability of reported social sensing data. Early examples are bound in machine learning and data mining literature [22], [23], [31] and as well as recent sensor network literature [10], [11], [15], [32]

One of the earliest efforts in this domain, Hubs and Authorities [31] presented a basic fact-finder, where the belief in a claim and the truthfulness of a source are computed in a simple iterative fashion. Latter, Yin et al. introduced TruthFinder as an unsupervised fact-finder for trust analysis on a providers-facts network [23]. Pasternack et al., extended the fact-finder framework by incorporating prior knowledge into the analysis and proposed several extended algorithms: Average.Log, Investment, and Pooled Investment [22]. Towards a joint estimation on source reliability and claim correctness, Wang et al. [10] first proposed an expectation maximization method to jointly estimate assertion value and source reliability. They further extend the joint estimation model by exploring the dependency among sources and assertions [11], [15].

The above work employed batch processing. In batch processing, there exists a large time gap between starting and finishing data collection, making fast estimation impossible. Therefore, we focus on streaming data in our current framework. An exception is [15] that discussed stream processing for truth estimation, but the work falls short due to its constraint that all sources make claims at each time slot, which is inapplicable for real applications. In this paper, we delete this limitation and propose a new streaming system. The system we proposed is both efficient and effective at estimating source and assertion reliability with a high degree of accuracy.

IX. Conclusions

In this paper, we presented a novel recursive estimator for ascertaining the correctness of observations reported on social networks. The estimator can be used in the context of social sensing applications by collecting the output of a social network, such as Twitter, that matches a given topic selected by the user (using an appropriate Twitter API), and interpreting the collected tweets as output of a participatory sensing campaign on the topic. Since participants are not vetted in advance, participant reliability varies, leading to the problem of joint estimation of source reliability and observation correctness. The paper is the first to offer a general *recursive estimator* that solves this problem. The estimator adopts a binary sensing model borrowed from recent literature. We evaluate the accuracy of our estimator in simulation as well as using real data collected from Twitter in March 2015. Results show that our estimator is significantly faster than previous bulk (offline) solutions, while matching or exceeding their accuracy. The estimator is incorporated into a tool that (i) allows defining virtual data collection "campaigns", (ii) collects tweets on campaign topics, then (iii) ascertains their reliability as discussed above, hence filtering out noise. The tool is currently in use as a personalizable news delivery systems using humans as sensors.

Acknowledgements

We sincerely thank Neal Patwari for shepherding the final version of this paper, and the anonymous reviewers for their invaluable comments. Research reported in this paper was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement W911NF-09-2-0053, DTRA grant HDTRA1-10-10120, and NSF grants CNS 09- 05014 and CNS 10-35736. 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] D. Wang, T. Abdelzaher, and L. Kaplan, *Social Sensing, Building Reliable Systems on Unreliable Data*, 1st ed. Morgan Kaufmann, March 2015.

[2] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," in *World Sensor Web Workshop*, 2006.

[3] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: Real-time event detection by social sensors," in *Proc. WWW*, April 2010.

[4] T. Silva, P. de Melo, J. Almeida, J. Salles, and A. Loureiro, "A picture of instagram is worth more than a thousand words: Workload characterization and application," in *In Proc. DCOSS*, May 2013.

[5] W. R. Ouyang, L. M. Kaplan, P. Martin, A. Toniolo, M. B. Srivastava, and T. J. Norman, "Debiasing crowdsourced quantitative characteristics in local businesses and services," in *Proc. IPSN*, April 2015.

[6] H. Zhuang, A. Parameswaran, D. Roth, and J. Han, "Debiasing crowdsourced batches," in *Proc. SIGKDD*, 2015.

[7] S. Zhi, B. Zhao, W. Tong, J. Gao, D. Yu, H. Ji, and J. Han, "Modeling truth existence in truth discovery," in *Proc. SIGKDD*, 2015.

[8] J. Gao, Q. Li, B. Zhao, W. Fan, and J. Han, "Truth discovery and crowdsourcing aggregation: A unified perspective," in *Tutorial in International Conference on Very Large Data Bases (VLDB)*, September 2015.

[9] J. Pasternack and D. Roth, "Latent credibility analysis," in *Proc. WWW*, 2013.

[10] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher, "On truth discovery in social sensing: A maximum likelihood estimation approach," in *Proc. IPSN*, 2012.

[11] D. Wang, M. T. Amin, S. Li, T. Abdelzaher, L. Kaplan, S. Gu, C. Pan, H. Liu, C. C. Aggarwal, R. Ganti *et al.*, "Using humans as sensors: An estimation-theoretic perspective," in *Proc. IPSN*, 2014.

[12] C. Meng, W. Jiang, Y. Li, J. Gao, L. Su, H. Ding, and Y. Cheng, "Truth discovery on crowd sensing of correlated entities," in *Proc. SenSys*, November 2015.

[13] A. Zubiaga and H. Ji, "Tweet, but verify: Epistemic study of information verification on twitter," in *Social Network Analysis and Mining*, 2013.

[14] S. Sikdar, B. Kang, J. O'Donovan, T. Hollerer, and S. Adah, "Understanding information credibility on twitter," in *Social Computing (SocialCom), 2013 International Conference on*, Sept 2013, pp. 19–24.

[15] D. Wang, T. Abdelzaher, L. Kaplan, and C. C. Aggarwal, "Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications," in *Proc. ICDCS*, 2013.

[16] D. Wang, L. Kaplan, and T. F. Abdelzaher, "Maximum likelihood analysis of conflicting observations in social sensing," *ACM Trans. Sen. Netw.*, vol. 10, no. 2, pp. 30:1–30:27, Jan. 2014. [Online]. Available: http://doi.acm.org/10.1145/2530289

[17] P. Netrapalli and S. Sanghavi, "Learning the graph of epidemic cascades," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1. ACM, 2012, pp. 211–222.

[18] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[19] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[20] G. S. Manku and R. Motwani, "Approximate frequency counts over data streams," in *Proc. VLDB*, 2002.

[21] G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine, "Synopses for massive data: Samples, histograms, wavelets, sketches," *Foundations and Trends in Databases*, vol. 4, no. 1–3, pp. 1–294, 2012.

[22] J. Pasternack and D. Roth, "Knowing what to believe (when you already know something)," in *Proc. Coling*, 2010.

[23] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the web," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 20, no. 6, pp. 796–808, 2008.

[24] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," in *Workshop on WSW*, 2006.

[25] J. Huang, S. Amjad, and S. Mishra, "Cenwits: a sensor-based loosely coupled search and rescue system using witnesses," in *Proc. Sensys*, 2005, pp. 180–191.

[26] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: A distributed mobile sensor computing system," in *Proc. Sensys*, 2006.

[27] R. Ganti, N. Pham, H. Ahmadi, S. Nangia, and T. Abdelzaher, "Greengps: A participatory sensing fuel-efficient maps application," in *Proc. MobiSys*, 2010.

[28] M. Uddin, H. Wang, F. Saremi, G. Qi, T. Abdelzaher, and T. Huang, "Photonet: A similarity-aware picture delivery service for situation awareness," in *Proc. RTSS*, 2011.

[29] U. Weinsberg, A. Balachandran, N. Taft, G. Iannaccone, V. Sekar, and S. Seshan, "Care: Content aware redundancy elimination for disaster communications on damaged networks," *Arxiv preprint arXiv:1206.1815*, 2012.

[30] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell, "Urban sensing systems: opportunistic or participatory?" in *Proc. HotMobile*, 2008.

[31] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[32] M. T. Amin, T. Abdelzaher, D. Wang, and B. Szymanski, "Crowd-sensing with polarized sources," in *in Proc. DCOSS*, 2014.