Truth Discovery in Data Streams: A Single-Pass Probabilistic Approach

Zhou Zhao, James Cheng and Wilfred Ng
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
Department of Computer Science and Engineering, The Chinese University of Hong Kong
{zhaozhou, wilfred}@cse.ust.hk, jcheng@cse.cuhk.edu.hk

ABSTRACT

Truth discovery is a long-standing problem for assessing the validity of information from various data sources that may provide different and conflicting information. With the increasing prominence of data streams arising in a wide range of applications such as weather forecast and stock price prediction, effective techniques for truth discovery in data streams are demanded. However, existing work mainly focuses on truth discovery in the context of static databases, which is not applicable in applications involving streaming data. This motivates us to develop new techniques to tackle the problem of truth discovery in data streams.

In this paper, we propose a probabilistic model that transforms the problem of truth discovery over data streams into a probabilistic inference problem. We first design a streaming algorithm that infers the truth as well as source quality in real time. Then, we develop a one-pass algorithm, in which the inference of source quality is proved to be convergent and the accuracy is further improved. We conducted extensive experiments on real datasets which verify both the efficiency and accuracy of our methods for truth discovery in data streams.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications

General Terms

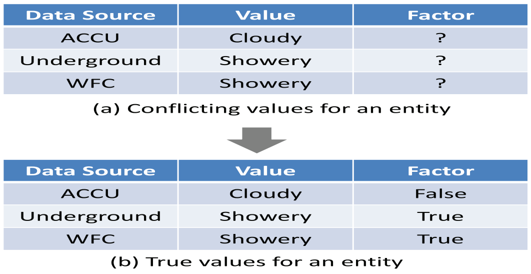
Algorithms, Design, Experiments

Keywords

Data Stream, Truth Discovery

1. INTRODUCTION

Truth discovery is an extensively studied topic in databases and its importance has been widely recognized by the research community [19, 6, 7, 14, 29, 25, 11]. In this paper, we study the problem of Truth Discovery in data streams. Many data stream management applications require integrating data from multiple sources in real time. For each entity, each source provides a value for it. However, the values of the entity from different sources may be conflicting, as some being true while others being false. To provide the true value for the entity, it is vital that the data stream management systems are capable of resolving such conflicts and discovering the true values.



Data Source	Value	Factor
ACCU	Cloudy	?
Underground	Showery	?
WFC	Showery	?

(a) Conflicting values for an entity

Data Source	Value	Factor
ACCU	Cloudy	False
Underground	Showery	True
WFC	Showery	True

(b) True values for an entity

Figure 1: Truth Discovery in Weather Forecast for New York

Consider a set of conflicting weather forecast values for New York at one timestamp as shown in Figure 1(a). The truth discovery for Figure 1(a) is to resolve the conflicts and find the true weather forecast values for New York in Figure 1(b). For example, the value “Cloudy” provided by the source ACCU is false information.

Previous works [3, 8, 6, 7, 10, 17, 18, 26, 27, 29, 31, 5] on truth discovery mainly focus on static databases. They defined the problem of truth discovery in the context of static databases. The definition is based on the quality of data sources and conflicting values for the entity. A data source that often provides true values is given a high score for its accuracy. A value for an entity that is provided by accurate sources is considered to be more likely true. Iterative methods were proposed to alternatively discover the true values and estimate the accuracy of the sources.

In recent years, advances in mobile technologies have led to the proliferation of many online data intensive applications, in which data in streaming format are being collected continuously in large volume and high speed. Effective and efficient truth discovery methods for such high speed data streams are essential to a wide range of applications, such as weather forecast, stock price prediction, flight schedule checking, etc. Moreover, there is also a rising need to share huge amounts of data in various commercial and scientific applications. Whether or not the data is naturally in streaming format, its sheer volume makes it impractical to make multiple passes of the data for truth discovery, while it is also unrealistic to assume that the data can be loaded into main memory for truth discovery. More compelling reasons of studying data stream integration can be found in [23]. However, as discussed in [14, 31], existing methods focus on static databases, where the prob-

CIKM'14, November 3–7, 2014, Shanghai, China.
Copyright 2014 ACM 978-1-4503-2598-1/14/11 ...$15.00.
http://dx.doi.org/10.1145/2661829.2661892.

lem of truth discovery is solved using iterative methods. Thus, it is difficult for these methods to discover truth in data streams, since their techniques are based on iterative updates of the score of source quality and values for the data, which requires the entirety of the data for the processing (i.e., the data needs to reside in main memory, or otherwise the cost of random disk access incurred will be too high).

To develop efficient and effective techniques for truth discovery in data streams, we should address the following three major challenging computational issues:

- **One-Pass Nature:** The streaming data from sources arrive in large quantities and at high speed. Thus, it is impractical to perform truth discovery on high-speed data streams by offline multi-passing algorithms. Instead, an efficient algorithm should read the data only once.

- **Limited Memory Usage:** Data streams are too voluminous to be kept in main memory, even though memory has become cheaper today. Moreover, many emerging applications require truth discovery in data streams in memory limited environments such as in mobile devices, which can only hold a limited amount of data.

- **Short Response Time:** Data streams such as weather forecast and stock price prediction are arriving continuously, while the applications often require real-time response. Thus, truth discovery should be performed with limited processing time, i.e., the algorithm should be able to process high-speed data streams.

None of the existing methods for truth discovery have effectively addressed the above computational issues. In fact, the best existing approach that addresses these issues is probably the method *majority vote*, which simply considers the value returned by the majority of the sources. However, this method is known to be error-prone [31, 14], since the method values the quality of all sources equally. In general, an effective truth discovery method should take into consideration the difference in the quality of various sources.

In this work, we formulate the problem of truth discovery in data streams, and address all the three computational issues stated above by proposing a generative model for truth discovery. We assume that there exists a true value for the entity among the conflicting values provided by different sources. Note that, our focus is not on fuzzy value integration and hence if a true value does not exist, there is nothing or no truth to find. The proposed generative model for the collected conflicting values from various sources is based on two fundamental factors, which are *data uncertainty* and *source quality*. Within our proposed model, we transform the truth discovery problem into a probabilistic inference problem. We derive the optimal solution for the inference problem and propose an iterative algorithm to converge it. Then, we improve the iterative algorithm and design a streaming algorithm that infers the truth as well as source quality in real time. Then, we develop a one-pass algorithm, in which the inference of source quality is proved to be convergent and the accuracy is further improved. Specifically, we compute the posterior distribution of all possible values, and find the most probable one with the maximum probability. Intuitively, this model best explains all the possible values reported by sources and the conflicting values in the data streams.

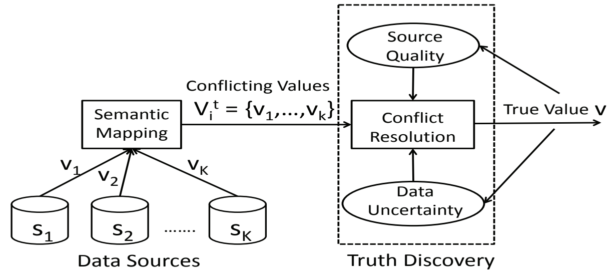


Figure 2: A Conceptual View of Truth Discovery in Data Streams

Figure 2 illustrates the important concepts and main ideas in the architecture of our truth discovery in data streams. As the sources can be heterogenous, we first employ a semantic mapping for the values provided by various sources, such that the values for truth discovery are in a consistent manner. For example, we consider the meaning of the weather conditions “rainy” and “wet” to be the same in weather forecast truth discovery. We also group “Partly Sunny” and “Mostly Cloudy” , and consider them to be the same as “Clear”. At each time t, the system collects a set of conflicting values for entity i as $V_i^t = \{v_1, v_2, \ldots, v_k\}$ from multiple data sources. Next, the system resolves the conflicts and discovers the true value v in V_i^t based on the current data uncertainty and source quality. Then, the system updates the data uncertainty and source quality based on the inferred value v and conflicting values V_i^t.

We summarize the main contributions of our work as follows.

- Most of the existing work focus on the problem of truth discovery in static databases. In this paper, we formulate the problem of truth discovery in data streams, and propose a new probabilistic model that resolves conflicting values arising from multiple data streams.

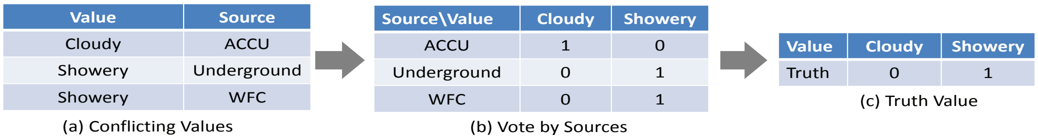
- We propose a novel source quality model for capturing various errors embedded in the information sources. Compared with existing source quality models that are based on a single accuracy value, our matrix model is more general and able to represent the quality of the sources better, since the model is able to represent true positive rate, true negative rate, false positive rate and false negative rate.

- We adopt a new approach that converts the truth discovery problem into a probabilistic inference problem. We then transform the complex inference problem into a high-dimensional optimization problem.

- We develop a one-pass algorithm that solves the problem of truth discovery in data streams, which assumes limited main memory and short response time. We prove the convergence of source quality inference for one-pass algorithm. For truth discovery from the sources with time-evolving quality, we devise a streaming algorithm to adaptively infer the source quality over time.

- We evaluate the performance of our algorithms with data from real applications of truth discovery in data streams. Our results verify both the accuracy and efficiency of our one-pass algorithm and streaming algorithm for truth discovery in data streams

Organization. The rest of the paper is organized as follows. Section 2 surveys the related work. Section 3 presents our probabilistic model and formulates the problem of truth discovery in data streams. Section 4 introduces the optimization algorithms for the proposed problem. We report the experimental results in Section 5 and conclude the paper in Section 6.



Value	Source
Cloudy	ACCU
Showery	Underground
Showery	WFC

(a) Conflicting Values

Source\Value	Cloudy	Showery
ACCU	1	0
Underground	0	1
WFC	0	1

(b) Vote by Sources

Value	Cloudy	Showery
Truth	0	1

(c) Truth Value

Figure 3: An Illustration of Probabilistic Inference in Truth Discovery

2. RELATED WORK

Truth discovery for conflicting values is a fundamental problem in databases. The main challenges of such a problem are to resolve data inconsistency [1]. The problem of truth discovery for static databases was first formalized by Yin et al. [29] and an iterative algorithm was proposed to jointly infer the truth values and source quality. Pasternack and Roth developed several web-link based algorithms and proposed a linear-programming based algorithm [17]. They also introduced a generalized framework that incorporates background knowledge into the truth finding process [18]. Galland et al. introduced several fix-point algorithms to predict the truth values of the facts [10]. Wang et al. proposed an EM algorithm for discovering the truth in sensor networks [24]. Yin and Tan explored semi-supervised truth discovery by utilizing the similarity between data records [30]. Dong et al. studied the source selection problem for truth discovery [9]. Kasneci et al. developed a probabilistic model for truth discovery from several knowledge bases [12]. Pal et al. tackled the problem of evolving data integration [16]. Li et al. conducted an experimental study on existing algorithms [14]. A comprehensive survey of truth discovery techniques can be found in [8].

There are works that focus on other interesting aspects of data integration which are related to the problem of truth discovery. The Q system [22, 21, 28] develops an information need driven paradigm for data integration. The copying relationship detection in data integration was studied in [3, 7, 6, 19]. Liu et al. proposed an early-return data integration method when enough confidence is gained for the data from the unprocessed sources that are unlikely to change the answer [15]. Mehmet et al. addressed the privacy-aware data integration [13]. Several works on data fusion in wireless sensor networks and RFID systems are also related to our problem, which discover the true location or reading from a set of observed readings from the sensors [20, 32, 33]. However, these techniques first train the sensor model based on training datasets and then infer the true reading based on the trained models, which are not suitable for truth discovery in data streams.

Nonetheless, none of the above-mentioned algorithms are applicable for handling data streams. This is because the existing methods mainly require the entire dataset for processing. More recently, Zhao et al. studied the truth discovery problem using *Gibbs* sampling and they showed that their algorithm outperforms prior methods [31]. An incremental algorithm was also proposed, which is based on the trained model from the batch databases, to discover the truth of the new data. However, their incremental algorithm assumes a training phrase on a given batch dataset, which is not efficient enough, or even not possible, for processing data stream integration in many applications.

Contrary to all the above-mentioned work, we develop efficient algorithms for truth finding in data streams, which satisfy the constraints of one-pass nature, short response time, and limited memory usage.

3. A PROBABILISTIC MODEL FOR TRUTH DISCOVERY IN DATA STREAMS

In this section, we present a probabilistic approach that transforms the problem of truth discovery over data streams into a probabilistic inference problem. We derive the optimal solution for the inference problem and devise an iterative method. Specifically, we calculate the posterior distribution of all possible values, and find the most probable one with the maximum probability. Intuitively, the proposed model explains all the possible votes by the sources on the conflicting values. We propose a truth discovery method based on the generative process of the votes in the data streams.

We start by illustrating the general process of truth discovery in data streams. After that, we introduce some basic notions and notations in Section 3.1, and define the problem in Section 3.2. Then, we present a generative process for the vote on conflicting values in Section 3.3 and define the probabilistic model in Section 3.4.

Now, we illustrate the general process of truth discovery for the vote on conflicting values by sources using the following example.

EXAMPLE 1. *Consider a set of conflicting weather forecast values for New York City at time t as shown in Figure 3(a). We aim to report the correct weather conditions in real time. We first extract the weather forecast values "Cloudy" and "Showery"provided by the sources. We then record the votes towards the extracted values in Figure 3(b). For example, the source ACCU votes only for "Cloudy" , the source Underground votes only for "Showery", and the source WFC also votes for "Showery".*

The process of truth discovery is to validate the correctness of each value provided by the sources. An example of true weather condition is given in Figure 3(c). □

3.1 Notions and Notations

3.1.1 Conflicting Values

The conflicting values for an entity at time t are a set of values provided by sources, which are exclusive. We denote the conflicting values for entity i at time t by $V_i^t = \{v_{1,i}^t, v_{2,i}^t, \dots, v_{K,i}^t\}$ where $v_{k,i}^t$ is the value by source k for entity i at time t. For example, the weather forecast values for New York City at time t are given in Figure 3(a), i.e., $V^t_{\text{New York}} = \{v^t_{\text{ACCU,New York}}, v^t_{\text{Underground,New York}}, v^t_{\text{WFC,New York}}\}$ = {Cloudy, Showery, Showery}. The value v can be either literal or numeric. We consider the conflicting values for entities at time t as V^t and the sequential conflicting values at different timestamps as $V = \{V^1, V^2, \dots, V^T\}$ where T can be infinite.

3.1.2 Vote

We now consider the vote by different data sources for an entity i at time t as $O_i^t = \{\{o^t_{i,1,v_1}, \dots, o^t_{i,K,v_1}\}, \dots, \{o^t_{i,1,v_{n_i^t}}, \dots, o^t_{i,K,v_{n_i^t}}\}\}$ where $\{o^t_{i,1,v_1}, \dots, o^t_{i,K,v_1}\}$ is the vote of sources on value v_1 for entity i and n_i^t is the number of possible values for entity i at time t. For example, the vote by three sources ACCU, Underground and WFC for the weather forecast values of New York at time t in Figure 3(b) by vote set $O^t_{\text{New York}} = \{\{o^t_{\text{New York,ACCU,Cloudy}}, o^t_{\text{New York,Underground,Cloudy}}, o^t_{\text{New York,WFC,Cloudy}}\}, \{o^t_{\text{New York,ACCU,Showery}}, o^t_{\text{New York,Underground,Showery}}, o^t_{\text{New York,WFC,Showery}}\}\} = \{\{1,0,0\}, \{0,1,1\}\}$. The vote value o can either be 0 or 1. We consider the vote for entities at time t as O^t and the vote at different timestamps as $O = \{O^1, O^2, \ldots, O^T\}$ where T can be infinite.

3.1.3 Truth Value

We denote the truth value for an entity i at time t as $Z_i^t = \{z^t_{i,v_1}, z^t_{i,v_2}, \ldots, z^t_{i,v_{n_i^t}}\}$ where z^t_{i,v_j} is a validator for value $v^t_j \in V^t_i$ ($z^t_{i,v_j} \in \{0,1\}$). The validator $z^t_{i,v_j} = 1$ if the value v_j is the truth in the values for entity i at time t, V_i^t, otherwise $z^t_{i,v_j} = 0$. An example of the truth value for entity New York is given in Figure 3(c), i.e., $Z^t_{\text{New York}} = \{z^t_{\text{New York,Cloudy}}, z^t_{\text{New York,Showery}}\} = \{0, 1\}$.

3.1.4 Source Quality

Existing work [8, 7] usually models the quality of data sources using single accuracy value. However, using single accuracy value may not explain the possible mistakes made by the source such as false positive and false negative mistakes. Thus, we propose a more general quality model using confusion matrix for each source s, denoted by π^s. The proposed quality model aims to explain the vote by the source. The quality model of source s is given by

$$\pi^s = \begin{pmatrix} \pi^s_{00} & \pi^s_{01} \\ \pi^s_{10} & \pi^s_{11} \end{pmatrix}. \tag{1}$$

Consider a conflicting value set for an entity i, V_i^t, with its true value Z_i^t, we explain the vote $o^t_{i,s,v}$ of source s on the value $v \in V_i^t$. Based on the confusion matrix of the quality model π^s, there are four cases of the vote, given by

$$\pi^s_{mn} = p^s(o^t_{*,s,v} = m | z^t_{*,v} = n), m, n \in \{0, 1\} \tag{2}$$

where $p^s(o^t_{*,s,v} = m | z^t_{*,v} = n)$ is the probability of source s to give a vote m given the value truth n.

We define π^s_{11} as true positive rate, π^s_{10} as false negative rate, π^s_{01} as false positive rate, and π^s_{00} as true negative rate, where π^s_{11}, π^s_{10}, π^s_{01}, $\pi^s_{00} \in [0, 1]$ and $\pi^s_{11} + \pi^s_{10} = 1$ and $\pi^s_{01} + \pi^s_{00} = 1$.

We now explain the vote by referring to the source given by the example in Figure 3. Suppose the true weather forecast value is "Showery" as given in Figure 3(c), we then explain the vote by sources ACCU and Underground. The probability of voting by source ACCU on "Cloudy" is based on its false positive rate π^{ACCU}_{01} and the probability of voting on "Showery" is based on its false negative rate π^{ACCU}_{10}. We simulate the vote by the sources based on the confusion matrix of the quality model. We can see that the source Underground casts a vote on the value "Showery" while the source ACCU does not cast a vote. We conclude that the true positive rate $\pi^{Underground}_{11}$ is higher. On the other hand, the source ACCU makes the mistake on voting the true value, since its false negative rate π^{ACCU}_{10} is high. Similarly, the source Underground does not cast a vote on a false value "Cloudy" , which illustrates its high true negative rate $\pi^{Underground}_{00}$. In our proposed source quality model, the accuracy of the source depends on both true positive rate and true negative rate.

We model the quality of a set of sources S by a collection of confusion matrices $\Pi = \{\pi^1, \pi^2, \ldots, \pi^{|S|}\}$.

3.2 Problem Definition

We now formulate the problem of truth discovery in data streams as follows.

Given sets of conflicting values $V^1, V^2, \ldots, V^\infty$ provided by a set of sources S, we aim to validate each value of the entities in the set V^t such that the following three computational issues are effectively addressed: (1) *one-pass nature*: the streaming collections of values can be only read once; (2) *limited memory usage*: only the current collection of values can be kept in main memory; and (3) *short response time*: the total running time should be linear to the size of streaming collections of values, and the validation of the values in each set V^t should be performed online.

3.3 A Generative Process

Given a set of sources, we illustrate the generative process for the observed vote O. We first denote a set of parameters $\varphi = \{\overrightarrow{\alpha}, \overrightarrow{\beta}\}$, where $\overrightarrow{\alpha}$ is the hyper-parameters for confusion matrices and $\overrightarrow{\beta}$ is the hyper-parameters for value uncertainty.

3.3.1 Generating Truth Value Z

For each entity i at time t, its truth Z_i^t consists of a set of validators z^t_{i,v_j} for values $v_j \in V_i^t$. Since the value of each validator z^t_{i,v_j} is binary (i.e. $z^t_{i,v_j} \in \{0,1\}$), we assume that it is generated from the Bernoulli distribution [2]. The Bernoulli distribution is the most widely used distribution for binary random variables, which generates value 1 with success probability θ and value 0 with failure probability $1 - \theta$. Thus, the probabilistic generation for validators z^t_{i,v_j} is given by

$$\begin{aligned} z^t_{i,v_j} &\sim Bernoulli(\theta^t_{v_j}) \\ &\sim (\theta^t_{i,v_j})^{z^t_{i,v_j}} (1 - \theta^t_{i,v_j})^{1 - z^t_{i,v_j}} \end{aligned} \tag{3}$$

where θ^t_{i,v_j} is the prior probability of value v_j to be the truth in the value set V_i^t.

The probability $\theta^t_{*,v}$ models the the uncertainty of value v to be the truth or not. We assume that $\theta^t_{*,v}$ is generated by a Beta distribution [2]. The Beta distribution generates a continuous value θ within an interval $[0, 1]$ with two parameters β_1 and β_0. We choose the Beta distribution to generate $\theta^t_{*,v}$ because the Beta distribution is the conjugate prior [2] of the Bernoulli distribution. The probabilistic generation of value uncertainty $\theta^t_{*,v}$ with hyperparameter $\overrightarrow{\beta} = (\beta_1, \beta_0)$ is given by

$$\begin{aligned} \theta^t_{*,v} &\sim Beta(\overrightarrow{\beta}) \\ &\sim \frac{\Gamma(\beta_1 + \beta_0)}{\Gamma(\beta_1)\Gamma(\beta_0)} (\theta^t_{*,v})^{\beta_1 - 1} (1 - \theta^t_{*,v})^{\beta_0 - 1} \end{aligned} \tag{4}$$

where Γ is a gamma function [2], β_1 is the prior truth count, and β_0 is the prior false count for the values to be the truth in the data streams.

3.3.2 Generating Confusion Matrix Π

We now show the generative process for each confusion matrix π^s. As stated above, the entries of the confusion matrix have the property that $\pi^s_{11} + \pi^s_{10} = 1$ and $\pi^s_{00} + \pi^s_{01} = 1$. For brevity, we give the generative process for true positive rate π^s_{11} and true negative rate π^s_{00} of source s.

We first assume that the true positive rate π^s_{11} is generated from a Beta distribution with hyperparameters α_{11} and α_{10} in $\overrightarrow{\alpha}$, given by

$$\begin{aligned} \pi^s_{11} &\sim Beta(\overrightarrow{\alpha}) \\ &\sim \frac{\Gamma(\alpha_{11} + \alpha_{10})}{\Gamma(\alpha_{11})\Gamma(\alpha_{10})} (\pi^s_{11})^{\alpha_{11} - 1} (1 - \pi^s_{11})^{\alpha_{10} - 1} \end{aligned} \tag{5}$$

where α_{11} is the prior true positive count and α_{10} is the prior false negative count for the confusion matrix π^s.

We then assume that the true negative rate π^s_{00} is also generated from a Beta distribution with hyperparameters α_{00} and α_{01} in $\overrightarrow{\alpha}$, given by

$$\begin{aligned} \pi^s_{00} &\sim Beta(\overrightarrow{\alpha}) \\ &\sim \frac{\Gamma(\alpha_{01}+\alpha_{00})}{\Gamma(\alpha_{00})\Gamma(\alpha_{01})}(\pi^s_{00})^{\alpha_{00}-1}(1-\pi^s_{00})^{\alpha_{01}-1} \qquad (6) \end{aligned}$$

where α_{00} is the prior true negative count and α_{01} is the prior false positive count for the confusion matrix π^s.

3.3.3 *Generating Vote O*

We show the generative process of the votes made by each source. For the conflicting values of each entity i, we assume that the votes by source s is generated from the Bernoulli distribution based on the confusion matrix π^s and the truth value for entity Z^t_i. The value of each vote by source s for entity i, $o^t_{i,s,v}$ is also binary, and thus the Bernoulli distribution is a suitable choice for its generation. The probabilistic generation for the vote $o^t_{i,s,v}$ is given by

$$\begin{aligned} o^t_{i,s,v} &\sim Bernoulli(\pi^s_{z^t_{i,v}}) \\ &\sim (\pi^s_{z^t_{i,v}})^{o^t_{i,s,v}}(1-\pi^s_{z^t_{i,v}})^{1-o^t_{i,s,v}} \qquad (7) \end{aligned}$$

For example, if the value v is the truth in V^t_i (i.e. $z^t_{i,v}=1$), then the vote $o^t_{i,s,v}$ is generated by the true positive rate π^s_{11} or false negative rate π^s_{10} of source s.

3.4 Model Definition

In the previous discussion, we described a generative process for the votes O. We now formally define a probabilistic model that represents the underlying joint distribution over the generation of prior distribution for the truth Θ, truth value Z, source quality Π and the votes O.

Given hyper-parameters $\varphi = \{\overrightarrow{\alpha}, \overrightarrow{\beta}\}$, and a set of sources S, we factorize the joint distribution over Z, Θ, Π and O, given by

$$p(\Theta, Z, \Pi, O|S,\varphi) = p(\Theta|\overrightarrow{\beta})p(Z|\Theta)p(\Pi|\overrightarrow{\alpha})p(O|\Pi, Z)$$

where

$$\begin{aligned} p(\Theta|\overrightarrow{\beta}) &= \prod_{t=1}^{T}\prod_{i=1}^{N}\prod_{v\in V^t_i} p(\theta^t_{i,v}|\beta_1,\beta_0), \\ p(Z|\Theta) &= \prod_{t=1}^{T}\prod_{i=1}^{N}\prod_{v\in V^t_i} p(z^t_{i,v}|\theta^t_{i,v}), \\ p(\Pi|\overrightarrow{\alpha}) &= \prod_{s\in S} p(\pi^s_{11}|\alpha_{11},\alpha_{10})p(\pi^s_{00}|\alpha_{01},\alpha_{00}), \\ p(O|\Pi, Z) &= \prod_{t=1}^{T}\prod_{i=1}^{N}\prod_{v\in V^t_i}\prod_{s\in S} p(o^t_{i,s,v}|\pi^s, z^t_{i,v}), \end{aligned}$$

and the probability distributions $p(\theta^t_{*,v}|\overrightarrow{\beta})$, $p(z^t_{*,v}|\theta^t_{*,v})$, $p(\pi^s_{11}|\alpha_{11},\alpha_{10})$, $p(\pi^s_{00}|\alpha_{01},\alpha_{00})$, $p(o^t_{i,s,v}|\pi^s, z^t_{i,v})$ are defined in Equations 3-7, respectively. For brevity, we omit the conditional part of the joint distribution $p(\theta, Z, \Pi, O|S, \varphi)$ and abbreviate it to $p(\theta, Z, \Pi, O)$ in the rest of this paper.

Based on the model, the problem of truth finding for observed vote can be transformed into a standard probabilistic inference problem, namely, finding the maximum a posterior (MAP) configuration of the truth Z conditioning on O. That is to find

$$Z^* = \arg\max_{Z} p(Z|O) \qquad (8)$$

where $p(Z|O)$ is the posterior distribution of Z given the votes O (and φ). However, it it difficult to compute the posterior distribution of Z,

$$p(Z|O) = \int\int p(\Theta, Z, \Pi|O)d\Theta d\Pi, \qquad (9)$$

where

$$p(\Theta, Z, \Pi, |O) = \frac{p(\Theta, Z, \Pi, O)}{\sum_Z \int\int p(\Theta, Z, \Pi, O)d\Theta d\Pi}. \qquad (10)$$

This distribution is intractable to compute due to the coupling between Π and Θ. To tackle this problem, we develop an efficient and effective approximation algorithm in the next section.

4. THE OPTIMIZATION ALGORITHM

In this section, we propose the algorithms to approximate the distribution $p(\Theta, Z, \Pi|O)$ defined in Equation 9. We first introduce a batch optimization algorithm for the proposed problem by assuming that T is a fixed value. Then, we introduce two online algorithms for solving the problem of truth discovery over data streams.

4.1 Batch Optimizing Algorithm

We present a variational algorithm for discovering the truth with fixed T. We first restrict the variational distribution to a family of distributions that factorize as follows:

$$q(\Theta, Z, \Pi) = (\prod_{t=1}^{T}\prod_{i=1}^{N}\prod_{v\in V^t_i} q(\theta^t_v)q(z^t_v))\prod_{s\in S} q(\pi^s_{11})q(\pi^s_{00}).$$

Thus the calculation of the joint probability distribution can be reduced to the product of multiple distributions and thus the computation cost can be greatly reduced.

The choice of variational distributions is not arbitrary and we require the distribution in the same family of the model probability distribution and take the following parametric form:

$$\begin{aligned} & q(\Theta, Z, \Pi|\gamma, \eta, \overrightarrow{\lambda}) \\ = \; & (\prod_{t=1}^{T}\prod_{i=1}^{N}\prod_{v\in V^t_i} q(\theta^t_v|\gamma)q(z^t_v|\eta))\prod_{s\in S} q(\pi^s_{11}|\lambda_1)q(\pi^s_{00}|\lambda_0), \end{aligned}$$

where

$$\begin{aligned} q(\theta^t_v|\gamma) &= Beta(\gamma), \\ q(z^t_v|\eta) &= Bernoulli(\eta), \\ q(\pi^s_{11}|\lambda_1) &= Beta(\lambda_1), \\ q(\pi^s_{00}|\lambda_0) &= Beta(\lambda_0). \end{aligned}$$

Here, γ, η, λ_1 and λ_0 are the variational parameters.

Thus, the inference for the truth value in Equation 8 can be simplified as follows:

$$\begin{aligned} Z^* &= [\arg\max_{Z^1} q(Z^1), \arg\max_{Z^2} q(Z^2), \ldots, \arg\max_{Z^T}(Z^T)] \\ &= [\arg\max_{\eta^1} \eta^1, \ldots, \arg\max_{\eta^T} \eta^T] \qquad (11) \end{aligned}$$

The goal of the variational algorithm is to find the variational distribution that is close to the true posterior $p(\Theta, Z, \Pi|O)$. This is equivalent to optimizing the variational parameters γ, η, λ_1 and λ_0 with respect to some distance measure, given by

$$\begin{aligned}&(\gamma^*, \eta^*, \lambda_1^*, \lambda_0^*)\\ =\ & \arg\min_{\gamma,\eta,\lambda_1,\lambda_0} D(q(\gamma,\eta,\lambda_1,\lambda_0)||p(\Theta,Z,\Pi|O)).\end{aligned}$$

In this work, we adopt the Kullback-Leibler (KL) divergence which is commonly used to measure the difference between two distributions. It is defined as

$$\begin{aligned}&KL(q||p)\\ =\ & \sum_Z \int\int q(\gamma,\eta,\lambda_1,\lambda_0)\log\frac{q(\gamma,\eta,\lambda_1,\lambda_0)}{p(\Theta,Z,\Pi|O)} d\Theta d\Pi,\end{aligned}$$

where KL divergence is a function of the variational parameters γ, η, λ_1 and λ_0. However, directly optimizing the KL divergence is infeasible because the KL divergence involves the term $p(\Theta, Z, \Pi|O)$, which is intractable.

Instead, we solve an equivalent maximization problem, whose objective function is defined as

$$\mathcal{L}(q) = \sum_Z \int\int q(\gamma,\eta,\lambda_1,\lambda_0)\log\frac{p(\Theta,Z,\Pi,O)}{q(\gamma,\eta,\lambda_1,\lambda_0)} d\Theta d\Pi$$

The equivalence between these two optimization problems can easily be seen as their objective functions sum up to a constant

$$KL(q||p) + \mathcal{L}(q) = \log p(O).$$

In order to maximize the objective function $\mathcal{L}(q)$, we take the derivatives of it with respect to the variational parameters γ, η, λ_1 and λ_0, and set these derivatives to zeros.

$$\nabla\mathcal{L}(q) = (\frac{\partial\mathcal{L}}{\partial\eta}, \frac{\partial\mathcal{L}}{\partial\gamma}, \frac{\partial\mathcal{L}}{\partial\lambda_1}, \frac{\partial\mathcal{L}}{\partial\lambda_0}) = \overrightarrow{0}. \tag{12}$$

For clarity, we put all the derivations in the Appendix. We report the solutions to the optimization problem by

$$\begin{aligned}\eta^t_{i,v,j} \quad \propto \quad & \exp\{\psi(\gamma^t_{i,j}) - \psi(\sum_{m=0}^{1}\gamma^t_{i,m})\}\\ \times \quad & \exp\{\sum_{s\in S}\sum_{j=0}^{1} o^t_{i,s,v,j}(\psi(\lambda^s_{i,v,j}) - \psi(\sum_{m=0}^{1}\lambda^s_{i,v,m}))\} \end{aligned}\tag{13}$$

$$\gamma^t_{i,j} \quad = \quad \beta_j + \sum_{i=1}^{N}\sum_{v\in V_i^t}\eta^t_{i,v,j} \tag{14}$$

$$\lambda^s_{0,j} = \alpha_{0,j} + \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,0}o^t_{i,s,v,j}. \tag{15}$$

$$\lambda^s_{1,j} = \alpha_{1,j} + \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,1}o^t_{i,s,v,j}. \tag{16}$$

for all $s = 1, \ldots, |S|$; $t = 1, \ldots, T$; $i = 1, \ldots, N$; $j \in \{0, 1\}$. $\psi(\cdot)$ is the Digamma function which is the logarithmic derivative of the Gamma function $\Gamma(\cdot)$, given by

$$\psi(x) = \frac{\partial \log\Gamma(x)}{\partial x}.$$

4.2 Streaming Optimization Algorithm

In this section, we develop a streaming truth finding algorithm called *StreamTF*, in Algorithm 1. The *StreamTF* algorithm is able to heuristically find the truth with one-pass nature, short response time and limited memory usage. Furthermore, *StreamTF* algorithm is also capable for truth discovery in the case that the quality of data sources evolves.

The idea of the *StreamTF* algorithm is based on the sequential Bayesian estimation, given by

$$p(\varphi^t|O^1, O^2, \ldots, O^t) \propto p(O^t|\varphi^{t-1})p(\varphi^{t-1}|O^1, O^2, \ldots, O^{t-1})$$

where φ^{t-1} is the estimated variational parameters at time t-1. This indicates that we can use a posterior $p(\varphi^{t-1}|O^1, O^2, \ldots, O^{t-1})$ as the prior and infer the variational parameters φ^t based on the collection of votes O^t. Thus, we present the techniques of the *StreamTF* algorithm that finds the truth and estimates the source quality sequentially. We notice that Equations 15 and 16 for estimating the source quality can also be represented as

$$\begin{aligned}\lambda^s_{0,j} \quad = \quad & \{\alpha_{0,j} + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,0}o^t_{i,s,v,j}\}\\ + \quad & \sum_{i=1}^{N}\sum_{v\in V_i^T}\sum_{s\in S}\eta^T_{i,v,0}o^T_{i,s,v,j},\end{aligned}$$

$$\begin{aligned}\lambda^s_{1,j} \quad = \quad & \{\alpha_{1,j} + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,1}o^t_{i,s,v,j}\}\\ + \quad & \sum_{i=1}^{N}\sum_{v\in V_i^T}\sum_{s\in S}\eta^T_{i,v,1}o^T_{i,s,v,j},\end{aligned}$$

where we can interpret the terms $\{\alpha_{0,j} + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,0}o^t_{i,s,v,j}\}$, and $\{\alpha_{1,j} + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{v\in V_i^t}\sum_{s\in S}\eta^t_{i,v,1}\, o^t_{i,s,v,j}\}$ as the prior parameters of true negative rate and true positive rate of source s, denoted as $(\lambda^s_{0,j})^{T-1}$ and $(\lambda^s_{1,j})^{T-1}$, respectively. Next, we consider $(\lambda^s_{0,j})^{T-1}$ and $(\lambda^s_{1,j})^{T-1}$ as the prior parameters for estimating data truth η^T and data uncertainty γ^T. Then, we estimate source quality $(\lambda^s_{0,j})^T$ and $(\lambda^s_{1,j})^T$ based on the estimated η^T and γ^T, given by

$$(\lambda^s_{0,j})^T = (\lambda^s_{0,j})^{T-1} + \sum_{i=1}^{N}\sum_{v\in V_i^T}\sum_{s\in S}\eta^T_{i,v,0}o^T_{i,s,v,j}, \tag{17}$$

$$(\lambda^s_{1,j})^T = (\lambda^s_{1,j})^{T-1} + \sum_{i=1}^{N}\sum_{v\in V_i^T}\sum_{s\in S}\eta^T_{i,v,1}o^T_{i,s,v,j}. \tag{18}$$

The *StreamTF* is outlined in Algorithm 1. We now show how the *StreamTF* algorithm effectively addresses the three computation issues stated in Section 3.2. First, *StreamTF* achieves one-pass nature since it is obvious that the algorithm reads the data only once. Second, *StreamTF* achieves limited memory usage because it only uses memory of size of one collection of votes at any time in the stream. Third, *StreamTF* achieves short response time since the algorithm reports the truth online and our experiments also verify that our algorithm can process a lot of collections of votes in one second, which is in effect real time response.

Algorithm 1 Streaming Truth Finding (**StreamTF**)

Input: Observed votes O^t, variational parameters λ^{t-1}, a threshold ϵ
Output: Variational parameters η^t, λ^t, γ^t

```
for $t = 1 \rightarrow \infty$ do
    for each source $s \in S$ do
        Set true negative rate $\lambda_0^s \leftarrow (\lambda_0^s)^{t-1}$
        Set true positive rate $\lambda_1^s \leftarrow (\lambda_1^s)^{t-1}$
    end for
    repeat
        for entity $i : 1 \rightarrow N$ do
            for each value $v \in V_i^t$ do
                Update $\eta_{i,v}^t$ by Equation 13
            end for
        end for
        Update $\gamma_i^t$ by Equation 14
    until change in $\frac{1}{N}\sum_{i=1}^{N}\sum_{v \in V_i^t} \eta_{i,v}^t < \epsilon$
    for each source $s \in S$ do
        Update true negative rate $(\lambda_0^s)^t$ by Equation 17
        Update true positive rate $(\lambda_1^s)^t$ by Equation 18
    end for
end for
return $\eta^t$, $\lambda^t$, $\gamma^t$.
```

Algorithm 2 One-Pass Truth Finding (**1PassTF**)

Input: Observed votes O, input data size T, a threshold ϵ
Output: Variational parameters η, λ, γ

```
Define $\rho^t = (\tau + t)^{-\kappa}$
for $t = 1 \rightarrow T$ do
    repeat
        for entity $i : 1 \rightarrow N$ do
            for $v \in V_i^t$ do
                Update $\eta_{i,v}^t$ by Equation 13
            end for
        end for
        Update $\gamma_i^t$ by Equation 14
    until change in $\frac{1}{N}\sum_{i=1}^{N}\sum_{v \in V_i^t} \eta_v^t < \epsilon$
    for each source $s \in S$ do
        Update variational parameters $\lambda^s$ by Equation 20
    end for
end for
return $\eta$, $\lambda$, $\gamma$.
```

4.3 One-Pass Optimization Algorithm

We can further improve the *StreamTF* algorithm if the size of the dataset is known, by which we can design a one-pass algorithm that not only satisfies the three computational issues of data stream processing stated in Section 3.2, but also stochastically maximizes the objective function in Equation 12, i.e., $\mathcal{L}(q)$. Such a one-pass algorithm is particularly useful for processing massive static databases.

We observe that the objective function $\mathcal{L}(q)$ can be represented as T functions of the variational parameters, given by

$$\mathcal{L}(q) \quad = \quad \sum_{t=1}^{T} \ell(O^t, \eta^t, \theta^t, \lambda_0, \lambda_1).$$

where we consider η^t and θ^t as local parameters for the function $\ell(O^t, \eta^t, \theta^t, \lambda_0, \lambda_1)$ and λ_0, λ_1 as global parameters for source quality.

The challenge of this problem is the inference for parameters λ_0, λ_1. The reason is that we only have to keep one collection of votes O^t at each time. To tackle this problem, we develop our one-pass algorithm based on the stochastic natural gradient algorithm [4]. We model the streaming collections of votes $O^1, O^2, \ldots, O^T$ to be sampled from uniform distribution, that is, $h(O^t) = \frac{1}{T}$. The expectation of the objective function is given by

$$E_h[\mathcal{L}(q)] = T \times E_h[\ell(O^t, \eta^t, \theta^t, \lambda_0, \lambda_1)] \tag{19}$$

Then, we optimize Equation 19 by repeatedly sampling the collection of votes at different times, and applying the update

$$\begin{aligned} & \lambda_{i,j}^s \\ \leftarrow \quad & \lambda_{i,j}^s + \rho^t T \frac{\partial \ell(O^t, \eta^t, \theta^t, \lambda_0, \lambda_1)}{\partial \lambda_{i,j}^s} \\ = \quad & (1-\rho^t)\lambda_{i,j}^s + \rho^t(\alpha_{i,j} + T\sum_{n=1}^{N}\sum_{v \in V_n^t}\sum_{s \in S} \eta_{n,v,i}^t o_{n,s,v,j}^t) \end{aligned} \tag{20}$$

for all s, $i \in \{0,1\}$ and $j \in \{0,1\}$, where ρ^t is the decay factor. The derivation of Equation 20 can be found in the Appendix.

To guarantee the convergence of source quality, we set the decay factor as the function of $\rho^t = (\tau + t)^{-\kappa}$ where the parameters κ,τ control the learning rate of old $\lambda_{i,j}^s$ to be forgotten. We set $\kappa > 0.5$ and $\tau > 0$ such that $\sum_{t=1}^{\infty} \rho^t = \infty$ and $\sum_{t=1}^{\infty}(\rho^t)^2 < \infty$, where the estimation of $\lambda_{i,j}^s$ can converge to a stationary point.

THEOREM 1. *(Online Optimization [4]) The general greedy descent method converges if and only if its learning rates ρ fulfills*

$$\sum_{t=1}^{\infty}(\rho^t)^2 < \infty, \sum_{t=1}^{\infty}\rho^t = \infty.$$

The detailed proof to the above theorem can be found in [4], and we give the intuition of the convergence on $\lambda_{i,j}^s$ here. The ratio $\rho^t = (\tau + t)^{-\kappa}$ is a function of t and becomes smaller after the algorithm is run for more iterations. The function $(\alpha_{i,j}+ T\sum_{n=1}^{N}\sum_{v \in V_n^t}\sum_{s \in S} \eta_{n,v,i}^t\ o_{n,s,v,j}^t)$ is the inference for new $\lambda_{i,j}^s$. The update on the variational parameter $\lambda_{i,j}^s$ is the product of $(\alpha_{i,j}+T\sum_{n=1}^{N}\sum_{v \in V_n^t}\sum_{s \in S} \eta_{n,v,i}^t\ o_{n,s,v,j}^t)$ and ρ^t which becomes less after the algorithm iterates more. Thus, the inference of parameter $\lambda_{i,j}^s$ becomes convergent.

The *1PassTF* algorithm is outlined in Algorithm 2. It is easy to see that the *1PassTF* algorithm also addresses the three computation issues stated in Section 3.2 by following the same analysis given to the *StreamTF* algorithm at the end of Section 4.2.

5. EXPERIMENTAL RESULTS

In this section, we evaluate the effectiveness and efficiency of our algorithms. All the algorithms, including those we compared with in the experiments, were implemented in Java and tested on machines with Windows OS, Intel(R) Core(TM2) Quad CPU 2.66Hz, and 8GB of RAM.

5.1 Datasets

We use three real datasets to evaluate the performance of our algorithms. Some statistics of the datasets are reported in Table 1.
Weather. We collected the weather forecast data in May 2013 for 285 US cities that have a population of at least 100,000. The weath-

Table 1: Statistics of Real Datasets

Datasets	#Votes O_i^t	#Sources	#Values		Entropy	
			Avg	Dev	Avg	Dev
Flight	35k	35	2.809	1.159	0.710	0.147
Weather	154k	7	2.148	2.502	0.531	0.412
Stock	21k	51	7.581	4.027	1.227	0.073

er forecast data are modeled as streams reported hourly from different sources. The source is obtained as follows. We searched "weather forecast" on *Google* and collected the deep-web sources from the top 100 returned results. Among them, we chose the sources where the weather forecast data are encoded in the URL. Then, we selected the sources that forecast the weather hourly and removed the sources whose data were copied from other sources. Finally, we obtained a set of seven sources. We also collected the historic weather data for these cities from a normal weather forecast website[1] as the groundtruth for our evaluation. We use the historic weather data as the truth data, since they are recorded after the day. On the other hand, the weather forecast data may contain some mistakes as the data are based on some kind of prediction.

Flight. The flight data contains 1200 flights from three airlines (AA, UA and Continental) in one month. The sources include the official websites of the three airlines, and 32 third-party websites that provide flight information for the airlines. We consider the data provided by the official websites of the three airlines as the gold standard.

Stock. The stock data contains 1000 stocks from 55 sources over one month. We use the data provided from NASDAQ100 as the gold standard.

We obtain both Flight data and Stock data from the website[2]. These two datasets are used as benchmark datasets in the experimental study in [14]. The gold standards are the same as suggested in [14]. For some sources of both Flight data and Stock data, they may copy the values from other sources to provide the values. We use the well-known copy-detection method in [7, 19] to remove these sources.

5.2 Experimental Settings

For continuous values in the datasets such as the departure time of flight and stock price, we transform their values into discrete format by the notations of tolerance and bucketing (as suggested in [14]) as follows.

- **Tolerance.** For the departure time in the flight data, we tolerate a 10-minute difference. For a stock price value at time t, we consider all the values V^t provided by the sources and compute the mean value $\overline{V^t}$. The tolerance for V^t is computed based on $\overline{V^t}$ and a predefined threshold ε, which is given by

$$\tau(V^t) = \varepsilon \times \overline{V^t},$$

  where the threshold ε is set to 0.01 by default.

- **Bucketing.** For each collection of values V^t, we group values with very small differences into a bucket. We start to compute its mean value $\overline{V^t}$ and put all the values into the following buckets: $\ldots, (\overline{V^t} - \frac{3\tau(V^t)}{2}, \overline{V^t} - \frac{\tau(V^t)}{2}], (\overline{V^t} - \frac{\tau(V^t)}{2}, \overline{V^t} + \frac{\tau(V^t)}{2}], (\overline{V^t} + \frac{\tau(V^t)}{2}, \overline{V^t} + \frac{3\tau(V^t)}{2}], \ldots$

We now report the consistency of the datasets above. We consider the observed values and votes at time t as V^t and O^t, respectively. We use the average number of values to measure the uncertainty of the data streams. We employ the entropy to measure the confusion of the conflicting values for the sources. Both average number and entropy are popular measurements for data consistency, which are also used in the experimental study of the work in [14].

- **Average Number of Values.** We denote the number of values for entity i by n_i^t. For the collection of values $V = \{V^1, V^2, \ldots, V^T\}$, we consider the average number of values for data streams as $\overline{|V|} = \frac{1}{T\times N}\sum_{t=1}^{T}\sum_{i=1}^{N} n_i^t$. The standard deviation of $\overline{|V|}$ is computed by

$$\delta(\overline{|V|}) = \sqrt{\frac{1}{T \times N}\sum_{t=1}^{T}\sum_{i=1}^{N}(n_i^t - \overline{|V|})^2}.$$

- **Average Entropy.** We consider the vote for entity i from source $s \in S$ on the value v at time t as $o_{i,s,v}^t$, which is a binary value. For all the observed votes $O = \{O^1, O^2, \ldots, O^T\}$, we consider the average entropy of the votes from sources by

$$\begin{aligned}\overline{Ent(O)} &= \frac{1}{T \times N}\sum_{t=1}^{T}\sum_{i=1}^{N} Ent(O_i^t)\\ &= \frac{1}{T \times N}\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{1}{n_i^t}\sum_{i=1}^{n_i^t}\sum_{s\in S}\frac{o_{i,s,v}^t}{|S|}\log\frac{o_{i,s,v}^t}{|S|}.\end{aligned}$$

  And its standard deviation can be computed as

$$\delta(\overline{Ent(O)}) = \sqrt{\frac{1}{T \times N}\sum_{t=1}^{T}\sum_{i=1}^{N}(Ent(O_i^t) - \overline{Ent(O)})^2}.$$

The details of the statistics of the real datasets can be found in Table 1.

We evaluate the performance of our batch algorithm and the two streaming algorithms using the above datasets. To measure the effectiveness of our methods, we define the average accuracy (AVG), minimum accuracy (MIN) and standard deviation of accuracy (DEV). We first partition the collection of the data streams into buckets of the same size and compute the accuracy of the algorithm for each bucket. Then, we compute the average accuracy and minimum accuracy over all the buckets as AVG and MIN, respectively. Finally, we compute the standard deviation of accuracy based on the bucket accuracy and average accuracy as DEV. By default, we set the size of each bucket to be 300. For the streaming algorithms, we also evaluate their robustness by varying the decay factor κ, and the decay seed τ.

5.3 Streaming Truth Finding

We next evaluate the performance of our streaming algorithms, **StreamTF** and **1passTF**, for the following three measures: (1) accuracy, (2) running time, and (3) robustness. Since there is no existing algorithm tackling truth finding over data streams, we use the incremental algorithm **LTMinc** [31] as the baseline for comparison. We use 10%-40% percentage of the data for training the LTM model for LTMinc, denoted by LTMinc$_{0.1}$-LTMinc$_{0.4}$, respectively.

5.3.1 Accuracy of Streaming Algorithms

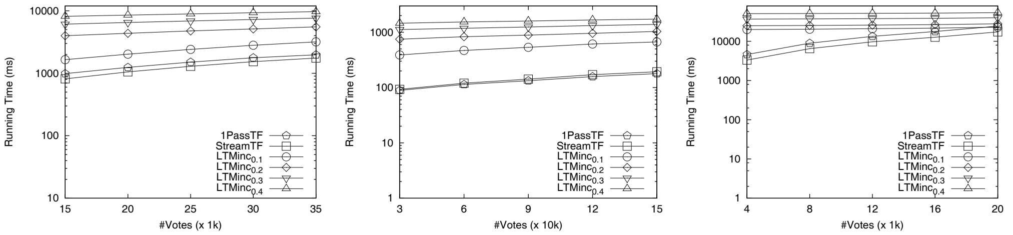
We now present the result of accuracy of the algorithms in Table 2. For all the three datasets, both StreamTF and 1passTF achieve

[1] `http://www.weatherforyou.com/`

[2] `http://cs.binghamton.edu/~xianli/truthfinding.htm`

Table 2: Accuracy of Streaming Algorithms (the best score in bold except Dev))

Method	Flight			Weather			Stock		
	Avg	Min	Dev	Avg	Min	Dev	Avg	Min	Dev
1PassTF	**0.9575**	**0.8732**	0.0165	**0.9426**	**0.8769**	0.0171	**0.9071**	**0.8320**	0.0506
StreamTF	0.9565	0.8683	0.0187	**0.9426**	0.8481	0.0226	0.8985	0.8320	0.0580
LTMinc$_{0.1}$	0.8426	0.8280	0.0105	0.8009	0.1699	0.3512	0.7733	0.6698	0.0699
LTMinc$_{0.2}$	0.8610	0.8471	0.0090	0.8092	0.2796	0.3736	0.7800	0.6650	0.0734
LTMinc$_{0.3}$	0.8572	0.8458	0.0076	0.8106	0.3133	0.3231	0.7711	0.6650	0.0758
LTMinc$_{0.4}$	0.8776	0.8664	0.0063	0.8177	0.2392	0.3553	0.7837	0.7279	0.0549



(a) Flight (b) Weather (c) Stock

Figure 4: Running Time of Streaming Algorithms

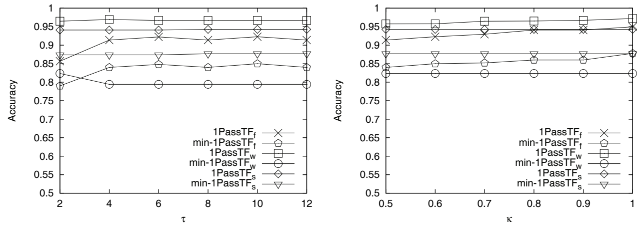
significantly higher accuracy than LTMinc. Both the average accuracy and minimum accuracy of StreamTF and 1passTF are higher than those of LTMinc.

Notably, the accuracy of 1passTF is higher than that of StreamTF, which can be explained as follows. Both StreamTF and 1passTF sequentially find the truth as well as estimate the source quality over the data streams. The difference between StreamTF and 1passTF is on the estimation of source quality. The StreamTF algorithm estimates online the source quality based on sequential bayesian estimation in Equation 17. However, the estimation of source quality based on sequential bayesian cannot optimize the likelihood objective function. On the contrary, the 1passTF algorithm stochastically optimizes online the objective function in Equation 19 in order to accurately estimate the source quality using gradient descent in Equation 20. As a result, the performance of 1passTF is better than that of StreamTF.

5.3.2 *Running Time of Streaming Algorithms*

We report the running time of the algorithms in Figures 4(a), 4(b) and 4(c), respectively. We sequentially pass the votes for the entity O_i^t to our algorithms. For all the datasets, StreamTF and 1passTF are faster than LTMinc, and able to process data streams at high speed.

It is also interesting to see that the running time of StreamTF and 1passTF decreases when more data have been processed (i.e., the running time increases sub-linear with the increase in the amount of data being processed), as shown in Figures 4(a), 4(b) and 4(c). The streaming algorithms keep estimating online the confusion matrix for source quality. They take more iterations to converge when estimating the source quality from the initial period of the data streams. As we know, the confusion matrix of the source quality is stationary. As the time passes by, our streaming algorithms take less and less iterations to converge. Finally, we find that the number of iterations for inferring the source quality becomes one when the estimation of source confusion matrix reaches the stationary point. Thus, the time cost of our streaming algorithms can be greatly reduced as more data is being processed.



(a) Decay Seed (b) Decay Ratio

Figure 5: Robustness of One-Pass Algorithms

5.3.3 *Robustness of One-Pass Algorithms*

We evaluate the robustness of the 1passTF algorithm by varying its parameters, decay ratio κ and decay seed τ, to validate its effectiveness. We measure the performance of the algorithms by the average accuracy and minimum accuracy (indicated by adding the prefix "**min-**" to the algorithm names in the figures), respectively. We denote the 1PassTF algorithm on different datasets such as flight, weather and stock by 1PassTF$_f$, 1PassTF$_w$ and 1PassTF$_s$, respectively. The results in Figure 5 show that our 1passTF algorithm is robust as it achieves quite consistent high accuracy for different values of decay ratio and decay seed. It is worth mentioning that the running time of 1passTF also remains stable for different values of decay ratio and decay seed.

6. CONCLUSIONS

In this paper, we studied the problem of truth discovery in data streams, which has a wide range of data stream applications such as weather forecast and flight scheduling. We proposed a probabilistic model that transforms the problem of truth discovery over data streams into a probabilistic inference problem. We first developed a streaming algorithm that discovers the truth under the constraints of one-pass nature, limited memory usage and short response time. Then, we also proposed a one-pass algorithm that is able to stochastically optimize the probabilistic inference of source quality, which is able to further improve the accuracy of the streaming algorithm. As for empirical study, we verified the effectiveness and efficiency of our algorithms using three real datasets from the data stream applications of weather forecast, flight scheduling and stock price prediction. The experimental results validate the effectiveness of our algorithms, in terms of both integration accuracy and running time.

Acknowledgments. We thank the reviewers for giving us many constructive comments, with which we have significantly improved our paper. This research is supported in part by SHIAE Grant No. 8115048, MSRA Grant No. 6903555, and HKUST Grant No. FS-GRF14EG31.

7. REFERENCES

[1] M. Arenas, L. Bertossi, and J. Chomicki. Consistent query answers in inconsistent databases. In *PODS*, pages 68–79, 1999.

[2] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

[3] L. Blanco, V. Crescenzi, P. Merialdo, and P. Papotti. Probabilistic models to reconcile complex data from inaccurate data sources. In *Advanced Information Systems Engineering*, pages 83–97, 2010.

[4] L. Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17:9, 1998.

[5] H. Chen, W.-S. Ku, H. Wang, and M.-T. Sun. Leveraging spatio-temporal redundancy for rfid data cleansing. In *SIGMOD*, pages 51–62, 2010.

[6] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: the role of source dependence. *PVLDB*, 2(1):550–561, 2009.

[7] X. L. Dong, L. Berti-Equille, and D. Srivastava. Truth discovery and copying detection in a dynamic world. *PVLDB*, 2(1):562–573, 2009.

[8] X. L. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. *The VLDB Journal*, 18(2):469–500, 2009.

[9] X. L. Dong, B. Saha, and D. Srivastava. Less is more: Selecting sources wisely for integration. *PVLDB*, pages 37–48, 2013.

[10] A. Galland, S. Abiteboul, A. Marian, and P. Senellart. Corroborating information from disagreeing views. In *Proceedings of the ACM international conference on Web search and data mining*, pages 131–140, 2010.

[11] L. Jia, H. Wang, J. Li, and H. Gao. Incremental truth discovery for information from multiple data sources. In *Web-Age Information Management*, pages 56–66. Springer, 2013.

[12] G. Kasneci, J. V. Gael, D. Stern, and T. Graepel. Cobayes: Bayesian knowledge corroboration with assessors of unknown areas of expertise. In *Proceedings of the ACM international conference on Web search and data mining*, pages 465–474, 2011.

[13] M. Kuzu, M. Kantarcioglu, A. Inan, E. Bertino, E. Durham, and B. Malin. Efficient privacy-aware record integration. In *EDBT*, pages 167–178. ACM, 2013.

[14] X. Li, X. L. Dong, K. Lyons, W. Meng, and D. Srivastava. Truth finding on the deep web: is the problem solved? *PVLDB*, pages 97–108, 2013.

[15] X. Liu, X. L. Dong, B. C. Ooi, and D. Srivastava. Online data fusion. *PVLDB*, 4(11), 2011.

[16] A. Pal, V. Rastogi, A. Machanavajjhala, and P. Bohannon. Information integration over time in unreliable and uncertain environments. In *WWW*, pages 789–798, 2012.

[17] J. Pasternack and D. Roth. Knowing what to believe (when you already know something). In *Proceedings of the International Conference on Computational Linguistics*, pages 877–885, 2010.

[18] J. Pasternack and D. Roth. Making better informed trust decisions with generalized fact-finding. In *Proceedings of the international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2324–2329, 2011.

[19] A. D. Sarma, X. L. Dong, and A. Halevy. Data integration with dependent sources. In *Proceedings of the 14th International Conference on Extending Database Technology*, pages 401–412. ACM, 2011.

[20] D. Smith and S. Singh. Approaches to multisensor data fusion in target tracking: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 18(12):1696–1710, 2006.

[21] P. P. Talukdar, Z. G. Ives, and F. Pereira. Automatically incorporating new sources in keyword search-based data integration. In *SIGMOD*, pages 387–398, 2010.

[22] P. P. Talukdar, M. Jacob, M. S. Mehmood, K. Crammer, Z. G. Ives, F. Pereira, and S. Guha. Learning to create data-integrating queries. *PVLDB*, 1(1):785–796, 2008.

[23] N. Tatbul. Streaming data integration: Challenges and opportunities. In *Data Engineering Workshops (ICDEW)*, pages 155–158, 2010.

[24] D. Wang, T. Abdelzaher, L. Kaplan, and C. C. Aggarwal. On quantifying the accuracy of maximum likelihood estimation of participant reliability in social sensing. *Urbana*, 51:61801, 2011.

[25] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. On truth discovery in social sensing: a maximum likelihood estimation approach. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 233–244. ACM, 2012.

[26] M. Wu and A. Marian. Corroborating answers from multiple web sources. In *WebDB*, 2007.

[27] M. Wu and A. Marian. A framework for corroborating answers from multiple web sources. *Information Systems*, 36(2):431–449, 2011.

[28] Z. Yan, N. Zheng, Z. G. Ives, P. P. Talukdar, and C. Yu. Actively soliciting feedback for query answers in keyword search-based data integration. *PVLDB*, pages 205–216, 2013.

[29] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *TKDE*, 20(6):796–808, 2008.

[30] X. Yin and W. Tan. Semi-supervised truth discovery. In *WWW*, pages 217–226, 2011.

[31] B. Zhao, B. I. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *PVLDB*, 5(6):550–561, 2012.

[32] Z. Zhao and W. Ng. A model-based approach for rfid data stream cleansing. In *CIKM*, pages 862–871, 2012.

[33] Z. Zhao, D. Yan, and W. Ng. A probabilistic convex hull query tool. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 570–573. ACM, 2012.