2015

Pay-as-you-go Feedback in Data Quality Systems

Romila Pradhan
Purdue University, rpradhan@purdue.edu

Siarhei Bykau
Purdue University, sbykau@purdue.edu

Sunil Prabhakar
Purdue University, sunil@purdue.edu

Report Number:
15-003

Pay-as-you-go Feedback for Data Quality Systems

Romila Pradhan Siarhei Bykau Sunil Prabhakar
Department of Computer Sciences
Purdue University
West Lafayette, Indiana, USA
{rpradhan, sbykau, sunil}@purdue.edu

ABSTRACT

In many domains such as the web, sensor networks and social media, sources often provide conflicting information. It is of utmost importance to resolve conflicts and identify correct information. A number of approaches, referred to as *truth finders*, have been proposed recently. They address the problem of truth discovery using different principles such as link analysis, Bayesian modeling and reputation systems. None of the existing approaches, however, leverages user feedback to improve the performance of these truth finders. In the present work, we propose a novel framework based on the concept of the value of perfect information that orders existing conflicts by their ability to boost the collective performance of the truth finder on all objects. We devise a number of algorithms that take into account the voting network structure and the level of agreement/disagreement among sources, and produce effective orderings of objects for validation with interactive response rates. Finally, we present an extensive experimental evaluation where we show that our solution outperforms existing truth finders, and also study the trade-offs between the efficiency and effectiveness of the various ordering algorithms.

1. INTRODUCTION

With the advent of modern information systems and services, the amount and diversity of data have been growing at an unprecedented pace in recent years. Moreover, the number of sources that provide data has significantly increased, spanning well-known sources, such as top news agencies (e.g., CNN, BBC, AFP), to individual contributors of Wikipedia articles. Unsurprisingly, conflicts among such data sources arise often, e.g., travel agencies report different departure times for the same flight [16], financial firms publish different stock prices of the same company [16], sensors report conflicting measurements [29], online bookstores list different authors for identical books [7] and so on. Resolving such conflicts is important since inaccurate information may result in unfavorable consequences such as a missed flight or severe financial losses.

A number of approaches, known as *truth finders*, have been proposed to deal with conflicting data sources and to discriminate `true` and `false` claims. Truth finders employ various techniques, such as *majority voting* that consider a claim provided by most of the sources to be true, *link-based approaches* [14] that consider the correctness of a claim to be dependent on the trustworthiness of its sources and the trustworthiness of a source to be an average of the correctness of claims it provides, or the most recent ideas based on *Bayesian modeling* [32, 6, 22, 29] that regard the credibility of sources and correctness of claims as latent variables. The latter methods show high effectiveness along with good practical applicability.

In order to further increase the effectiveness of a truth finder, we propose to leverage feedback provided by users. In this work we assume that the users are capable of providing highly accurate feedback on most of the claims. Dealing with uncertainty of feedback, e.g. collecting feedback using crowdsourcing [8, 17, 18] is orthogonal to the scope of our work and considered as future work. The users confirm or reject some of the claims and using that feedback, the truth finder improves its accuracy on other claims. There are, however, a number of technical challenges that need to be addressed: (i) Typically, a truth finder deals with a large number of claims (hundreds of thousands) thus limiting the ability to collect feedback to very small fractions of all claims. (ii) Selecting claims for validation is a difficult problem both from the efficiency and effectiveness points of view. The set of claims have complex dependencies among them and the validation of one claim causes changes in correctness of many other claims through *change propagation* in the network, e.g., when a claim is validated, the accuracy of sources that voted for it are modified and in turn, alter the probabilities of correctness of the claims these sources voted for and so on. As a result, validating one claim may lead to changes in others that are several hops away from it. (iii) Furthermore, since the state-of-the-art truth finders are based on the Expectation-Maximization (EM) algorithm, we face the problem of quantitatively estimating the impact of the validation of one claim on other claims. Due to the iterative nature of the EM algorithm, we cannot predict the changes in the probabilities of other claims analytically and need to re-run the EM algorithm for each possible validation – a prohibitively expensive procedure.

To the best of our knowledge, we are the first to propose the use of the decision-theoretic concept of the *value of perfect information* (VPI) [25] for the problem of data fusion. VPI has been used widely in areas such as economics [20],

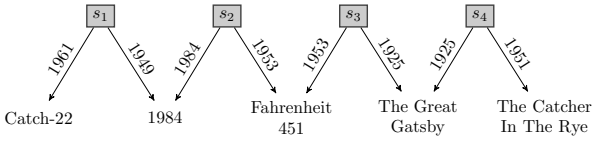


Figure 1: A motivating example: sources provide information on the publication date of famous novels.

healthcare [3] and in data cleaning [31, 12] and classification [13] within the database community. Data fusion, however, is fundamentally a different problem and adapting a solution from an unrelated domain is technically infeasible as these solutions are built on assumptions inherent to the specific problem, the data and the associated rules.

VPI is based on a *utility function* that measures the desirability of the current state of a truth finder for its users. We use VPI to select a claim whose validation maximizes the expected gain of the utility function. We show that this procedure leads to a prohibitively expensive computational cost since we have to compute the expected utility gain of every object using the iterative EM algorithm. To this end, we propose a set of approximation formulas that allow us to analytically estimate the impact of validation without re-running the EM. Furthermore, we take advantage of the voting network structure in order to identify claims that might have a greater impact upon validation.

Finally, we conduct an extensive experimental evaluation where we present the trade-offs between the efficiency and effectiveness of the proposed methods. Our findings indicate that the proposed techniques find sequences of validation that have high accuracy at low computational cost.

The summary of our contributions is as follows:

- We define the problem of feedback solicitation for truth finders.
- We design a framework based on VPI that provides an order in which objects should be validated in order to maximize the utility of the truth finder system.
- We propose algorithms that approximate the propagation of change in the the surrounding network.
- We conduct an extensive experimental evaluation where we show that the proposed algorithms have high efficiency and effectiveness.

The structure of the paper is as follows. In Section 2, we present a motivating example to illustrate the feedback solicitation problem in truth finders. Section 3 describes the technical details of a truth finder. Section 4 introduces the framework based on the concept of the value of perfect information. The solution is proposed in Section 5. In Section 6, we describe the experimental results. Section 7 discusses the related work of truth finders along with topics related to the value of perfect information and Bayesian networks. Finally, we conclude in Section 8.

2. MOTIVATING EXAMPLE

Consider an example of websites (sources) providing information on when certain English novels were first published (Figure 1). Each source is shown as a rectangle and novels are shown by their titles. The votes of sources are represented as arrows and labels denote the claims, e.g., source S_1 claims that `"Catch-22"` was first published in `1961`.

A truth finder takes the depicted voting graph as an input, outputs the accuracy of each source (i.e., the probability that a claim provided by the source is correct) and for each object, it provides the probability that a particular claim is correct. See Section 3 for details of a truth finder.

Assuming we can validate any object and know which of its claims is correct (by crowdsourcing or hiring an expert), which object should we select for validation? Obviously, selecting an object at random is not the best choice because some objects will lead to large changes and some will have no change in the probabilities of itself and its neighbors. Intuitively, our goal is to validate an object that would bring the modified probability estimates of all objects closer to ground truth (i.e., which values are `true` and which are `false`).

This is a difficult problem because we have to deal with a number of issues. First, we do not possess ground truth and therefore, need to find heuristics to select the *best* object. Second, since each object may potentially influence any other object in the voting network, the exhaustive search, i.e., checking each object and estimating its effect on all others, is prohibitively expensive (see Section 6 for details). We have to find better ways of selecting objects by taking advantage of the details of operations of truth finders as well as of the voting network structure.

Our first observation is that objects have different *levels of uncertainty* by virtue of the agreement/disagreement of sources on some claims. For example, one may expect that the impact of validating `"1984"` would be higher than that of `"The Great Gatsby"` since S_1 and S_2 disagree on `"1984"` and S_3 and S_4 agree on `"The Great Gatsby"` and we expect to learn more from the validation of objects with *disagreement*. Another observation is based on the voting network structure. Although an object may have disagreement over its values, its validation may not lead to high impact if it has few neighbors. For instance, validating `"Fahrenheit 451"` would potentially impact `"1984"` and `"The Great Gatsby"` since they are only one hop away whereas the validation of `"Catch-22"` influences directly only `"1984"`.

In this work, we focus on the problem of determining the best object to validate given the current state of the database and present an efficient solution that does not depend on ground truth.

3. TRUTH FINDERS

In this section, we describe a data model as well as a Bayesian truth finder. Note that there although there are various types of truth finders [2, 14], in this work our focus is on a Bayesian truth finder since it showed a superior performance in recent studies [32, 9, 6, 22, 29]. Many extensions and variations of a Bayesian truth finder have been proposed such as leveraging source dependencies [6, 23], using the hardness of facts [9], and many others. Our approach is based on the version that lies at the core of all extended methods and is presented below.

The input of a truth finder is viewed as a probabilistic graphical model [15] or, more specifically, as a Bayesian network. Let $S = \{s_1, \ldots, s_n\}$ be a set of sources that provide claims about objects from set $O = \{o_1, \ldots, o_m\}$. Let each object o_i have a number of possible claims, denoted by $V_i = \{v_i^1, \ldots, v_i^{k_i}\}$ where k_i is the total number of distinct claims about o_i. Only one of the claims is considered to be `true` and the rest are `false`. A set of claims about all objects is denoted by $V = \{V_1, \ldots, V_m\}$. Sources provide (or

vote for) specific claims of objects (at most one per object)– modeled as the set $\Psi = \{\psi_{j,i,k}\}$ where $\psi_{j,i,k} = 1$ if source s_j voted for claim v_i^k of o_i, and $\psi_{j,i,k} = 0$ otherwise.

EXAMPLE 3.1. *In the motivating example presented in Section 2, the set of all values of object "1984" is* $V_{\text{"1984"}} = \{1949, 1984\}$ *and the fact that source* s_1 *voted for value 1949 and not for value 1984 of object "1984" is represented by setting* $\psi_{1,\text{"1984"},1949} = 1$ *and* $\psi_{1,\text{"1984"},1984} = 0$.

Given all components defined above, we formally introduce a truth finder with its input and output structures.

DEFINITION 1. *A database,D, is a tuple*$\langle O, S, \Psi, V\rangle$ *where O is a set of all objects, S – a set of all sources,* $V = \{V_1, \ldots, V_{|O|}\}$ *– a set of sets of object claims and* Ψ *– a set of observations.*

A truth finder, denoted by $\mathcal{F}$, is a function that takes database D as input and outputs probability assignment P and a set of source accuracies A, i.e. $\mathcal{F} : D \rightarrow \langle P, A\rangle$ where for each claim $v_i^k \in V_i$ of object $o_i \in O$, $P(v_i^k) \in [0, 1]$ (note that as a shortcut, we use p_i^k as a substitute for $P(v_i^k)$) is the probability that claim v_i^k is `true` and for each $s_j \in S$ $A(s_j) = A_j$ is the overall accuracy of the jth source. Further, the probabilities of the distinct values of o_i sum up to 1.

In the above model, there are two kinds of variables: those we observe (the votes of sources on claims (Ψ)), and those we do not observe and have to infer (the accuracies of sources ($A(s_j)$) and the probabilities of claims p_i^k). Given the observable variables, our goal is to infer the unobservable variables. A solution for this problem is the iterative Expectation-Maximization (EM) algorithm [10]. The EM algorithm computes the accuracies of sources given probabilities of claims it provides and then computes the probabilities of claims given the accuracies of sources that supported a particular claim. This process is repeated until either the accuracies or the probabilities converge.

We now define the probability of the correctness of a claim and the accuracy of a source. The accuracy, $A(s_j)$, of source s_j is the probability that its claim about an object is `true` and is computed as follows:

$$A(s_j) = \frac{\sum_{i=1}^{m}{}^{\psi_{j,i,k}=1} p_i^k}{N(s_j)} \tag{1}$$

where $N(s_j)$ = number of objects for which source s_j votes.

In order to compute the probability of value v_i^r of object o_i being `true`, we use Bayesian analysis and first compute the probability of observation of object o_i conditioned on v_i^r being true as:

$$p(\psi_{.,i.,} \mid v_i^r = \texttt{true}) = \prod_{s\in S(v_i^r)} A(s) \cdot \prod_{s\in S_i \setminus S(v_i^r)} \frac{1 - A(s)}{|V_i| - 1}$$
$$= \prod_{s\in S(v_i^r)} \frac{(|V_i| - 1)A(s)}{1 - A(s)} \cdot \prod_{s\in S_i} \frac{1 - A(s)}{|V_i| - 1}$$

where S_i is the set of sources that provided information about object o_i and $S(v_i^o)$ is the set of sources that vote for some value v_i^o of o_i.

With the knowledge that only one of the claims is `true` and the rest are `false`, we apply Bayes rule to obtain the probability that claim v_i^r is `true` as:

$$p_i^r = p(v_i^r = \texttt{true} \mid \psi_{.,i,.}) = \frac{\prod_{s\in S(v_i^r)} \frac{(|V_i| - 1)A(s)}{1 - A(s)}}{\sum_{v_i^o \in V_i} \prod_{s\in S(v_i^o)} \frac{(|V_i| - 1)A(s)}{1 - A(s)}} \tag{2}$$

The EM algorithm initializes source accuracies with default values and computes the probabilities of claims of each object. It then recomputes the accuracies of sources using Formula 1. The process is repeated until either the accuracies of sources or the probabilities of claims converge.

4. USER FEEDBACK MODEL

In this section, we present the model of user feedback solicitation that allows us to improve the effectiveness of a truth finder. We discuss the basic concepts of our framework such as utility, action and the value of perfect information. We show that in the absence of domain knowledge and ground truth, we have to rely on an approximate utility function that is based on the idea of uncertainty reduction and referred to as the entropy utility function.

In this work, we follow a decision-theoretic framework for feedback collection that was introduced in the area of Artificial Intelligence and is found useful in diverse fields such as economics [20], healthcare [3] and data management [12, 21, 31, 11] (see Section 7 for more details). Data fusion, however, is fundamentally different from these works. Therefore, adapting solutions from an unrelated domain is not feasible and we need to introduce our specific framework.

We define the *utility function* as a function that measures the usefulness of a truth finder. Utility is higher if a truth finder is able to predict a greater number of `true` claims. Let $\mathcal{T} : V \rightarrow \{true, false\}$ be a truth function that assigns `true` to a correct claim and `false` to an incorrect claim.

DEFINITION 2. *Given truth function* $\mathcal{T}$*, database D and truth finder* $\mathcal{F} : D \rightarrow \langle P, A\rangle$*, the utility function* $U(D, \mathcal{F}, \mathcal{T})$ *is defined as:* $U(D, \mathcal{F}, \mathcal{T}) = \sum_{V_i\in V} \sum_{v_i^k \in V_i} \frac{p_i^k \delta(\mathcal{T}(v_i^k))}{|V_i|}$

where $p_i^k \in P$ *and* $\delta(v) = \begin{cases} 1, & \text{if } v = true \\ 0, & \text{otherwise} \end{cases}$

$U(D, \mathcal{F}, \mathcal{T})$ can be interpreted as measuring the average probability of `true` claims based on the probabilities output by truth finder $\mathcal{F}$. The closer the utility function to 1, the greater the number of claims correctly identified by $\mathcal{F}$.

We consider that user feedback is solicited in the form of validation of an entire object, e.g., we ask the user to provide the `true` affiliation of `"The Great Gatsby"`. In this work, we consider the user to be a domain expert who can provide highly accurate claims for most objects. There is a large body of work (e.g.,[28, 11]) that aims at collecting different types of feedback from a crowd of workers (e.g., Amazon Mechanical Turk, ClowdFlower). User feedback, in those cases, may contain errors. Our focus is the question of establishing the most important feedback and our approach is based on the mechanics of truth finder and available data. Dealing with a crowd, in our context, is left to future work.

The validation of an object is called an *action* and θ_i is a validation of object $o_i \in O$. The space of possible actions is determined by a set of objects that have not yet been validated. After the user performs an action, a truth finder may change its probabilities of claims of other objects since the

validated object augments our knowledge about accuracies of sources and subsequently, about the value probabilities.

Suppose the user performs action θ_i, validating object o_i. Our goal is to measure the usefulness of this action with respect to our utility function.

DEFINITION 3. *The value of perfect information (VPI) of action θ_i is defined as follows:*

$$VPI(\theta_i) = \sum_{v_i^k \in V_i} U(D, \mathcal{F}, \mathcal{T} \mid \mathcal{T}(v_i^k) = \texttt{true}) P(v_i^k) - U(D, \mathcal{F}, \mathcal{T})$$

where $\mathcal{F}(D) = \langle P, A \rangle$.

In other words, the VPI of action θ_i is an expected gain of the utility function based on the initial probabilities of claims v_i^k. Our goal is to identify an action (object) that would have the highest VPI.

In real-world applications, since we do not possess $\mathcal{T}$, we cannot use the utility function from Definition 2. We need an alternative that does not require the knowledge of ground truth. Generally, there are two strategies to solve this problem. On one hand, it is possible to approximate the utility function based on domain knowledge, e.g., query result quality in dataspaces [12], the importance of location (building) in geo-tagging [29], the market cap of a stock in stock data [6]. On the other hand, in the absence of domain knowledge, we can utilize the idea of *uncertainty reduction*, i.e., we identify actions that would reduce the uncertainty associated with the probabilities obtained with a truth finder. Note that this idea has been used extensively in the past [21, 11]. In this work, we assume that no domain knowledge is available and we focus on the latter type of utility function.

The utility function based on uncertainty reduction, referred to as the *entropy utility function*, is built on the concept of *entropy* [27] that is widely used in areas such as information theory, machine learning and statistics. It provides a measure of the level of uncertainty of probabilistic objects.

DEFINITION 4. *Given database D, truth finder $\mathcal{F} : D \rightarrow \langle P, A \rangle$, the entropy utility function $EU(D, \mathcal{F})$ is defined as:*

$$EU(D, \mathcal{F}) = -\sum_{p_i^k \in P} p_i^k log(p_i^k)$$

Intuitively, if the entropy of an object is low, then it has a low uncertainty, i.e., some claim has a high probability of being `true`, whereas if the entropy is high then all claims are almost equally likely. If the truth finder estimates claims to be equally likely, or in other words, produces a high-entropy output, we cannot really determine which claims should be considered as `true` and which – `false`. On the contrary, if the truth finder produces a low entropy output, we are more certain about `true`/`false` labels that should be attached to the claims and hence expect to obtain more benefits from the truth finder. However, a low entropy does not necessarily mean that the truth finder accurately predicts `true` claims since it may produce a high probability of a `false` claim that would lead to its *flip* upon validation. In Section 6, we present our detailed experimental study that shows that the entropy utility function is a good approximation of the utility function presented in Definition 2.

Having defined the entropy utility function, we apply the idea of VPI where instead of U we use EU in order to decide which action (validation) is the most useful. A set of all possible actions, denoted by Θ, consists of an action θ_i for each non-validated object $o_i \in O$. Among all available actions, Θ, we choose the one that maximizes the reduction of expected uncertainty:

$$\theta_i = \underset{\theta_i \in \Theta}{argmax} \ (EU(D, \mathcal{F}) - \sum_{p_i^k \in V_i} p_i^k EU(D, \mathcal{F} \mid v_i^k = true)) \quad (3)$$

Note that this kind of validation strategy is called *myopic* since we look only one step ahead each time we make a decision. It is possible that some object may not lead to the highest VPI at the current step but validating it could have resulted in the highest VPIs in subsequent validations. Typically, sequential validations are computationally expensive and in this work we focus only on myopic strategies.

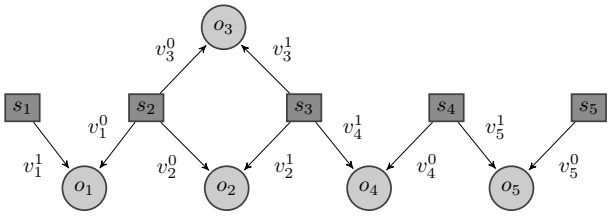


Figure 2: Example with five objects (circles) and five sources (boxes). Arrows and their labels represent claims provided by sources on objects.

5. SOLUTION

This section presents techniques to determine the action that would lead to maximum reduction in uncertainty of the database of objects. We introduce a brute force implementation (Section 5.1) of the VPI-based framework. We then present two heuristic solutions aimed at maximal uncertainty reduction in a single object (Section 5.2) or across all objects in the database (Section 5.3). Finally, we present a method that leverages the structure of interactions between objects and sources (Section 5.4.1).

5.1 Maximum Entropy Utility

We present a straightforward implementation of the framework described in Section 4 using the entropy utility function (`EU`). Maximum Entropy Utility, denoted by `MEU`, computes the expected entropy utility gain by considering the one-step lookahead state of the database after a *potential* action and aims at maximal uncertainty reduction across all objects.

`MEU` implements a *what-if* approach to determine the next action. It initially *assumes* a claim of an object to be `true` and computes the entropy utility of the object based on this claim. The entropy utility of the object is computed as the expected entropy utilities considering each of its claims to be `true`. The next action is selected as in Eq. (3).

EXAMPLE 5.1. *Consider object o_1 in Figure 2. The truth finder, $\mathcal{F}$, outputs $p_1^0 = p_1^1 = 0.5$. Before validation, $EU(D, \mathcal{F}) = 3.4657$. `MEU` runs $\mathcal{F}$ twice–once for each claim of o_1–to obtain $EU(D, \mathcal{F} \mid v_1^0 = \texttt{true}) = 0.008996$ and $EU(D, \mathcal{F} \mid v_1^1 = \texttt{true}) = 0.008974$. The expected utility gain of o_1 is computed as $\Delta EU(o_1) = 3.4657 - [(0.5)(0.008996) + (0.5)(0.008974)] = 3.4567$. `MEU` selects the object with the highest ΔEU.*

In the absence of ground truth or domain knowledge, `MEU` is considered to be the best alternative to the ground truth utility function. We see our experimental results with `MEU` in

Section 6.4.1. MEU shows an effectiveness close to the ground-truth-based utility function (Figure 3a) and also performs well with respect to the entropy utility function (Figure 3b).

The main drawback of MEU is its efficiency. In order to decide the next action, MEU re-runs truth finder $\mathcal{F}$ on the database of objects D for each claim of every object $o \in D$. The time complexity of MEU is $O(m\kappa t_{\mathcal{F}})$ where m is the number of unvalidated objects in D, κ is the average number of unique claims for each object and $t_{\mathcal{F}}$ is the time needed to run $\mathcal{F}$ for one instance of data. A typical run of the truth finder iterates over all objects and all sources until convergence. This contributes to an $O(m\kappa\mathcal{I}(m+n))$ complexity where $\mathcal{I}$ is the average number of iterations to convergence and n is the number of sources. With objects far outnumbering sources, the result is a complexity of $O(m^2\kappa\mathcal{I})$. Not surprisingly, in practice we observe a clear quadratic growth of time with increasing numbers of objects (see Figure 3c). Concluding, MEU can tackle datasets a few hundred objects in size at a reasonable time whereas our goal is to be able to order datasets with at least a few thousands of objects.

While MEU is a forthright implementation of the VPI-based framework, it is inherent with prohibitively expensive computation. To overcome this efficiency problem, we propose a number of heuristic-based methods that allow us to explore different trade-offs between efficiency and effectiveness and, as shown in the experiments, to scale our framework to thousands of objects without sacrificing the accuracy.

5.2 Local-MEU

This section presents a heuristic-based technique that aims at resolving conflict at the site of a single object. Local-MEU is built upon the principle of majority voting where a true claim of an object is the one that is supported by the largest number of sources. The intuition behind Local-MEU is that an object that a majority of the sources agree upon is less likely to be predicted incorrectly by the truth finder whereas the true claim of an object disputed by many sources might still be questionable. In such a case, it might be more beneficial to validate the latter object.

In order to determine the next action, for each $o_i \in O$, we use the votes of sources over its claims. Local-MEU computes the probability of the correctness of a claim v_i^k as the fraction of sources (voting for o_i) that supported v_i^k, i.e.,

$$p_i^k = \frac{\sum_{j=1}^{n} \psi_{j,i,k}}{\sum_{r=1}^{|V_i|} \sum_{j=1}^{n} \psi_{j,i,r}}$$

Given the probability distribution of claims of all objects in D, we determine the next action as the one that maximizes its entropy utility as:

$$a_i = \underset{\theta_i \in \Theta}{\operatorname{argmax}} \ (-\sum_{v_i^k \in V_i} p_i^k log(p_i^k)) \tag{4}$$

Local MEU selects an object with the highest local entropy utility. In other words, it selects the object that the truth finder is the least confident about.

Example 5.2. *In Figure 2, the local entropy utility of object* $o_4 = -(0.5\log(0.5)+0.5\log(0.5)) = 0.693$ *while the local entropy utility of object* $o_2 = -(1\log(1)) = 0$. *Clearly, validating* o_4 *will lower the database uncertainty to a greater extent than that achieved by confirming the true claim of* o_2.

Comparing Eqs. 3 and 4, we observe that MEU aims at lowering uncertainty across all objects in the database, whereas Local-MEU considers minimizing the uncertainty in a single object and does not take other objects into account.

Local-MEU involves computation at the site of an object and hence can be computed once at a very low cost. However, it has certain intrinsic limitations: (i) it does not take the accuracy of sources into account, and (ii) it does not consider possible interdependence among objects. While Local MEU reduces the uncertainty of the validated object, by ignoring dependence between objects there is no guarantee regarding the uncertainty reduction in other objects.

5.3 Approximate-MEU

MEU and Local-MEU are based on the implicit assumption that objects in the database are independent of each other. However, we expect that the validation of an object will alter its own probability distribution along with the probability distribution of its neighbors at the least. This intuition is based on principles inherent in Bayesian network inference methods such as belief propagation [15], variational message passing [30] and incremental expectation-maximization [19]. These methods decompose the computation into local object calculations that then pass to other objects via messages. Applied to our problem, a validation is considered as a local update of the probability distribution of an object that, in turn, is propagated to its neighbors.

This section proposes a method, denoted by Approx-MEU, that aims at leveraging the structure of interactions between objects and sources in order to determine the next action. In the bipartite object-source network, any change in one object is propagated to another through their common sources. Approx-MEU estimates the impact of a validation on the probability distributions (obtained using truth finder) of other objects either directly (through one or more sources) or indirectly (through other objects). We consider the impact of changes due to the validated object only and the updated probability distributions are obtained through linear approximation by differentials. The computation ignores higher order differentials and Approx-MEU selects the action that results in the maximum uncertainty reduction of the first-order approximate probabilities across all objects.

We start our analysis with the probability distributions of all objects in O obtained with truth finder $\mathcal{F}$.

Definition 5. *Given the probability distribution of claims of object* o_i, *the dominant claim* v_i^d *of* o_i *is defined to be the one that has the highest probability of being* true:

$$v_i^d = \underset{v_i^k \in V_i}{\operatorname{argmax}} \ p_i^k$$

All other claims $v_i^k \in V_i \setminus \{v_i^d\}$ *are non-dominant.*

The truth finder considers the dominant claim of o_i to be true. However, it may not always be correct and may incorrectly predict a false claim to be true. In such cases, we say that there is *flip* in judgement and incorporate the effect of this flip in our model. If $\mathcal{A}_{\mathcal{F}}$ is the accuracy of the truth finder, the probability that v_i^d is true is $\mathcal{A}_{\mathcal{F}}$ and there is only a $(1-\mathcal{A}_{\mathcal{F}})$ chance that a non-dominant claim is true.

Let us now consider two objects o_i and o_j. Our goal is to estimate the approximate probabilities of o_j after o_i has been validated. Approx-MEU operates in two steps: (i) measuring the change in probability distribution of o_i, and (ii)

estimating how the changes in o_i are propagated to o_j. In the following, we explain each of the steps of `Approx-MEU`:

Change in probabilities of o_i. We assume an arbitrary claim v_i^t of o_i to be `true`. After validating o_i, the change in probability of v_i^t is: $\Delta p_i^t = (1 - p_i^t)$. This validation ensures that all the other $(|V_i| - 1)$ claims of o_i are `false`. The change in probability of v_i^f, any claim other than v_i^t, i.e., $v_i^f \in V_i \setminus \{v_i^t\}$, is given by: $\Delta p_i^f = (0 - p_i^f) = -p_i^f$.

Propagation of changes in o_i to o_j. We want to estimate the change in probabilities of o_j as a function of the change in probabilities of o_i.

Before going into details, it is important to note that objects o_i and o_j could be either connected through a common source that votes for both of them or through a path consisting of alternating objects and sources. For example, in Figure 2, o_1 and o_2 are connected through source s_2 whereas o_1 and o_4 are connected through the $\langle o_1, s_2, o_2, s_3, o_4 \rangle$ path. We present an analysis of both the cases:

***Case I:* o_i and o_j have at least one common source.** We examine how the changes in the probability distribution of o_i affect the accuracies of the common sources (that voted for both o_i and o_j) as it is through these sources that the change is propagated to o_j.The influence in this case is straightforward: if a source supports the correct claim of o_i, we instill more trust in it and put greater belief in the information it provides about other objects. On the other hand, if a source provides false information about o_i, we believe it less for other objects as well.

Updates in source accuracies. With ΔP_i, the distribution of the change in probabilities of values of o_i, our model rewards sources that supported v_i^t and penalizes sources that voted for some other claim v_i^f. From Eq. (1), when only o_i has been updated, the accuracy $A(s)$ of a source s that supported claim v_i^k changes by $\Delta A(s) = \Delta p_i^k / N(s)$, where $N(s)$ is the number of objects for which source s voted. Thus,

$$\Delta A(s) = \begin{cases} \Delta p_i^t / N(s), & \text{if s voted for } v_i^t \\ \Delta p_i^f / N(s), & \text{if s voted for } v_i^f \in V_i \setminus \{v_i^t\} \end{cases} \quad (5)$$

Propagation of updates in sources to o_j. Our next task is to measure further propagation of changes from the sources to o_j. The analysis requires us to look deeper into the formulae described in Section 3. The probability of the correctness of claim v_j^r can be expressed as:

$$\frac{1}{p_j^r} = \sum_{v \in V_j} f(v_j^r, v) = \sum_{v \in V_j} \frac{\prod_{s \in S(v)} \frac{(|V_j|-1)A(s)}{1-A(s)}}{\prod_{s \in S(v_j^r)} \frac{(|V_j|-1)A(s)}{1-A(s)}} \quad (6)$$

In order to compute the change in p_j^r, we need to compute the approximate change in each $f(v_j^r, v)$ which is obtained by computing the first derivative of the term. This is done in a series of steps. We first take logarithm of $f(v_j^r, v)$ to obtain $\log f(v_j^r, v)$:

$$= \sum_{s \in S(v)} \log \frac{(|V_j|-1)A(s)}{1-A(s)} - \sum_{s \in S(v_j^r)} \log \frac{(|V_j|-1)A(s)}{1-A(s)} \quad (7)$$

Let us denote each log term in Eq. (7) by $A'(s)$, i.e.,

$$A'(s) = \log \frac{(|V_j|-1)A(s)}{1-A(s)}$$

The change in $A'(s)$ is obtained by differentiating it with respect to $A(s)$:

$$\Delta A'(s) = \frac{\Delta A(s)}{A(s)(1-A(s))}$$

Using this representation of $\Delta A'(s)$, we obtain $\Delta f(v_j^r, v)$ by differentiating Equation 7 as:

$$\frac{\Delta f(v_j^r, v)}{f(v_j^r, v)} = \sum_{s \in S(v)} \frac{\Delta A(s)}{A(s)(1-A(s))} - \sum_{s \in S(v_j^r)} \frac{\Delta A(s)}{A(s)(1-A(s))} \quad (8)$$

There is one last piece to the puzzle. Eq. (8) has the term $\Delta A(s)$, where s is a source that voted for object o_j. Observe that $\Delta A(s)$ can take a value as noted in Eq. (5) depending on whether: (i) s supported v_i^t, (ii) s voted for a claim other than v_i^t, or (iii) s did not provide any information about o_i. It is obvious that if s belongs to the third category, it will not be affected by validation of o_i. Knowing the claims of each of the sources, we can substitute $\Delta A(s)$ for each source appropriately in Eq. (8).

If we denote the set of sources that voted for the `true` claim v_i^t of o_i and for claim v_j^r of o_j by $S_{t,r}$, and similarly denote sources that supported a `false` claim v_i^f of o_i and for value v_j^r of o_j by $S_{f,r}$, $\frac{\Delta f(v_j^r, v)}{f(v_j^r, v)} =$

$$\sum_{s \in S_{t,v}} \frac{\Delta p_i^t}{N(s)A(s)(1-A(s))} + \sum_{s \in S_{f,v}} \frac{\Delta p_i^f}{N(s)A(s)(1-A(s))} - \sum_{s \in S_{t,r}} \frac{\Delta p_i^t}{N(s)A(s)(1-A(s))} - \sum_{s \in S_{f,r}} \frac{\Delta p_i^f}{N(s)A(s)(1-A(s))} \quad (9)$$

We are now ready to compute the change in probability of claim v_j^r of object o_j as a result of the change in probability distribution of o_i by the method of approximation by differentials of Eq. (6):

$$\Delta p_j^r = -(p_j^r)^2 \sum_{v \in V_j} \Delta f(v_j^r, v) \quad (10)$$

The approximate probability of claim v_j^r is obtained as:

$$(p_j^r)' = p_j^r + \Delta p_j^r \quad (11)$$

***Case II:* o_i and o_j have no source in common.** We know that any change in o_i reaches objects connected to it via at least one source, i.e., through objects that are one-hop away from o_i. The changes in these objects then reach objects one-hop away from them, and so on.

THEOREM 5.1. *The change in probabilities, Δp_j^r, of object o_j due to change in probabilities, Δp_i^k, of object o_i is inversely proportional to the minimum number of objects a source votes for, raised to the power of the number of hops o_j is away from o_i.*

$$\Delta p_j^r \propto \left(\frac{1}{N^2}\right) \Delta p_i^k$$

Proof. A detailed proof can be found in Appendix A.

Real-world datasets typically consist of a few sources that provide claims about a large number of objects. Therefore,

we observe an exponential decay in the change in probability distributions as we move away from the validated node. `Approx-MEU`, thus, ignores the changes in objects that are more than one hop away from the validated node.

Deciding the next action. Using Eq. (10), `Approx-MEU` estimates first-order approximations of the probabilities for all objects within one hop of o_i as a result of validating claim v_i^k of o_i. It then computes the overall utility of the resulting database as the Shannon entropy of probability distributions across all objects.

As discussed in the beginning of Section 5.3, there is a $\mathcal{A}_\mathcal{F}$ probability that the dominant claim, v_i^d, is `true`, and some other claim of o_i is `true` with a probability of $(1 - \mathcal{A}_\mathcal{F})$. We treat all `false` claims of an object equally and hence for each of the $(|V_i| - 1)$ `false` claims, the probability that it is `true` is $(1 - \mathcal{A}_\mathcal{F})/(|V_i| - 1)$.

The expected uncertainty of the database as a result of validating o_i is expressed as a weighted sum of the uncertainties if each of the $|V_i|$ distinct values of o_i is considered `true`. Among all available actions, `Approx-MEU` selects the one that results in the largest uncertainty reduction as:

$$a_i = \underset{\theta_i \in \Theta}{\operatorname{argmax}} \left(EU(D, \mathcal{F}) - \sum_{v_i^k \in V_i} \omega_k \sum_{v_j^r \in V_j} -(p_j^r)' \log(p_j^r)' \right) \tag{12}$$

where

$$\omega_k = \begin{cases} \mathcal{A}_\mathcal{F}, & \text{if } v_i^k \text{ is dominant value of } o_i \\ (1 - \mathcal{A}_\mathcal{F})/(|V_i| - 1), & \text{otherwise} \end{cases}$$

Example 5.3. *For the example presented in Figure 2, $EU(D, \mathcal{F}) = 3.4657$. `MEU` selects o_1 for validation since it has the highest entropy utility gain. `Local-MEU` selects one at random as all the objects have the same local entropy. `Approx-MEU` also selects o_1 as it has the maximum reduction in uncertainty of the approximate probabilities – validating o_1 reduces the uncertainty of the database by 0.8156 while validating o_5 results in a 0.7259 reduction in uncertainty.*

It is worthwhile to note that while `Local-MEU` may select o_5, `Approx-MEU` will never do so as it is aims at global reduction in uncertainty. This also aligns with our intuition that o_1 can directly influence more objects (o_2 and o_3) while o_5 can only affect o_4 and have diminished impact on the rest.

Complexity. `Approx-MEU` eliminates the bottleneck iterative computation (`MEU`). Instead, we have a first-order approximate estimation of changes due to each potential action that has κ distinct claims on an average. While deciding the next best action, it computes the approximate changes in probability distributions of all objects within one-hop neighborhood of the candidate object. As a result, the time complexity of `Approx-MEU` is $O(m\kappa d)$ where m is the number of unvalidated objects in D, κ is the average number of distinct claims for each object and d is the average number of objects one hop away from any object.

5.4 Shrinking the set of potential candidates

`Approx-MEU` considers the effect of validating one object on its neighboring objects. In datasets where each object is connected to every other object through at least one common source, the time complexity of `Approx-MEU` blows up to $O(\kappa m^2)$. There are, however, certain observations that aid us in improving this cost at a trade-off for effectiveness:

1. `Approx-MEU` is agnostic to the structure of the underlying object-source network.
2. `Approx-MEU` treats all objects equally while evaluating them for their impact on the entire dataset.

In the following sections, we describe methods built on these observations.

5.4.1 Network Approximate-MEU

Specific domain datasets may pertain to long-tail data that consist of few objects and a large number of sources such that the sources provide information about very few objects. The concept of a neighborhood is lost in such sparse datasets. Interestingly, the equations of the EM algorithm allow us to delve deeper in leveraging the network structure and the interaction of source votes. Since the change in probabilities of o_i travel to o_j through common sources, from Eq. (5), one of the primary observations in sparse networks is that: sources that vote for fewer objects allow better propagation of change in probabilities than sources that vote for more objects. This is because the change disappears in sources that vote for a large number of objects due to the $N(s)$ term in denominator.

This section proposes a heuristic, denoted by `Network-MEU`, built on the first observation mentioned above. It identifies a set, S_g, of n_g sources that act as *good conductors*, i.e., these sources allow better propagation of change in probabilities. All objects that sources in S_g provide information about are added to the set, O_p, of potential candidates for validation. For each object $o_i \in O_p$, it updates the probabilities of objects one hop away and selects the one that ensures the maximum reduction in uncertainty across all other objects.

Observe that n_g can be tuned such that O_p can have as few objects as the minimum number of objects one source votes for and go up to $|O|$ objects, i.e.

$$\min_{s \in S} \ |N(s_j)| \leq |O_p| \leq |O|$$

Complexity. By reducing the number of potential candidates for validation, `Network-MEU` has a time complexity of $O(|O_p|\kappa d)$ where O_p is the set of candidate objects, κ is the average number of claims per object and d is the average number of objects one hop away from any object.

5.4.2 Top-k Approximate-MEU

In the absence of ground truth and domain knowledge, the objective of our algorithm is to maximize the reduction in uncertainty of the database. We observe from the probabilities generated by the truth finder that some objects have much higher local uncertainty compared to others, i.e., the output from the truth finder is not polar. It is in our best interest to resolve these objects before considering others.

This section proposes a heuristic, denoted by `Approx-MEU`$_k$ that ranks objects according to their local entropies (computed as the Shannon entropy of the probabilities of an object as generated by the truth finder) and only considers the top k objects to compute the effect of validating one object on others. Mathematically, an object having higher local entropy contributes more to the uncertainty of the database and is more beneficial in terms of reducing uncertainty than objects having a lower local entropy. `Approx-MEU`$_k$ takes a parameter k as input, ranks objects based on their local entropies and adds the top k objects to the set, O_p, of potential candidates for validation. For each object in O_p, it

then calculates the effect of validating it on the reduction in uncertainty of the rest of the $(k-1)$ objects in O_p.

Complexity. For each of the k objects, `Approx-MEU`$_k$ computes first-order approximate estimation of changes in $(k-1)$ other objects. The resulting complexity is $O(k^2\kappa)$, where κ is the average number of claims per object and k is the parameter for the number of top objects to be considered.

6. EVALUATION

This section presents an empirical evaluation of the proposed algorithms using both real-world and synthetic datasets. Our datasets are described in Section 6.1. Section 6.2 outlines the various metrics reported in the experiments. while Section 6.3 enlists the algorithms proposed in this paper. The results of all the experiments are detailed in Section 6.4.

6.1 Datasets

6.1.1 Synthetic datasets

Dense data. According to our observations, truth finders are typically used in domains where the number of objects is significantly greater then the number of sources and voting networks are dense (e.g., see [16]). (Also, each source votes for many objects.) In the experiments with synthetic data, we assume that each object has two claims – one of which is `true` and the other is `false`. This allows us to focus on the feedback solicitation framework.

We introduce several parameters that are used for synthetic data generation. The number of objects (n) and the number of sources (m) are our primary parameters. In order for the data to exhibit characteristics similar to real-world data, we do not treat all sources equally – there are few very good sources (that provide correct information on more than 90% of the objects), few bad sources (that provide correct information on less than 65% of the objects) and the rest are average in quality. The accuracy of a source (i.e. the probability that it provides a true value for an object) is drawn uniformly from a range - $[a_{min}, a_{max}]$. Second, d specifies the density of a voting network, i.e. the probability that an arbitrary source voted for an arbitrary object. The default values for those parameters, $a_{min} = 0.6$, $a_{max} = 1$ and $d = 0.4$, correspond to the real dataset observations.

For each pair of objects and sources, source S_j provides a value for object o_i with probability d and the claim is correct with probability equaling the accuracy of S_j. We also maintain the constraint that each object has been voted by at least two sources.

Sparse data. To demonstrate the importance of network structure and the effectiveness of the approaches in Section 5.4, we perform experiments on sparse synthetic data. In real-world scenarios, this corresponds to long-tail data, i.e., sources providing information on very few objects. Such data, however, have certain sources that provide information about a lot of objects, thus rendering the object network dense. Instead, our requirements are: (i) objects have few votes, (ii) sources vote for few objects, (iii) the object network is connected, and (iv) there are certain important objects, e.g., famous geographical locations or trending tweets.

Keeping these in mind, a basic grid of objects and sources is laid down where each source votes for exactly two objects and each object is voted by at least two and at most, four sources. Hubs are added on top of the grid to replicate the last of our requirements. Again several parameters are considered for data generation. h is the fraction of objects that act as hubs (default = 0.02). The object-source votes matrix is filled up to density d as: for any pair of object and source, two Bernoulli trials were run with the probabilities p_N and p_A where a positive outcome of the first trial indicates that the source voted on the object and a positive outcome of the second trial indicates that the source supported the `true` claim (or on the `false` claim, otherwise).

6.1.2 Real datasets

Books dataset. We run our experiments on the books dataset that contains a listing of computer science books and their authors as provided by different online bookstores [32, 6]. The dataset has information about 1,265 books from 894 bookstores that were registered at *Abebooks.com*. In order to solicit feedback from the oracle, we need truth values for all objects in the dataset – we used the results when different fusion methods reach agreement; otherwise the silver standard provided in [6] was used.

Flights dataset. Our second real dataset is the flights data that contains flight status information (estimated/actual arrival/departure gates/times) for flights over a month's time as reported by 38 sources [16]. The result is a collection of more than $200,000$ *objects* where each object is a tuple identifying a flight and its corresponding value for an attribute. As a gold standard, we considered data provided by each of the carrier websites, *American Airlines, United Airlines* and *Continental*, to be correct information.

We permit slightly different reported values in flight arrival/departure times that might have arisen due to slight lag in updates, or simply due to pure error in estimating times. We tolerate a difference of a maximum of 10 minutes in two reported times and place these in the same bucket.

To simplify the experiments, for both the datasets, we consider objects that have up to two contesting values. In the case of the books dataset, whenever possible, we consider the top two author sets for each book.

6.2 Measurements

We measure the efficiency and effectiveness of the proposed methods using the following metrics:

Effectiveness. The utility of a truth finder is higher when it predicts a higher number of `true` claims. When we validate an object, we aim at approaching the ground truth claims (on all objects collectively) as much as is possible. Since we deal with a sequence of validations, we measure the effectiveness across many consecutive validations.

Distance to ground truth. An effectiveness experiment is a sequential validation of all available objects (in the order determined by a given method) where for each validated object, we obtain an assignment of `true` and `false` claims using truth function $\mathcal{T}$. We report the percentage reduction in distance to ground truth where the distance to ground truth itself is defined as:

$$distance\_to\_ground\_truth = \sum_{i=1}^{m} \sum_{v_i^k \in V_i} \frac{\delta(\mathcal{T}(v_i^k))(1 - p_i^k)}{m}$$

where $\delta(\mathcal{T}(v_i^k)) = 1$ if $v_i^k =$ `true` and p_i^k is the truth finder predicted probability of v_i^k, and m is the total number of objects. Intuitively, `distance_to_ground_truth` can be seen as an average error of a truth finder.

Reduction in Uncertainty. As discussed in Section 4, the ground truth utility function is not always feasible due to the absence of ground truth in real-life problems. To this

end, we report the reduction in uncertainty of all objects after each validation computed as:

$$reduction\_in\_uncertainty = U_0 - \sum_{i=1}^{m}\sum_{k=1}^{|V_i|} -p_i^k \log(p_i^k)$$

where U_0 is the uncertainty of the probabilities before validation and p_i^k is the probability that claim v_i^k is `true`.

Figures 3a and 3b respectively exhibit example curves for the reduction in `distance_to_ground_truth` and reduction in `uncertainty`. Both the curves start at 0 when no object is validated and gradually approach -100 (when all objects are validated). We say that a method has a higher effectiveness if the reduction in the said metric is faster (i.e., the slope of its effectiveness plot is steeper).

Efficiency Another important aspect of the various validation methods is the time taken to determine the object to be validated next. Our goal is to provide an interactive response time for users of a truth finder and thus, we report the time taken for one validation (measured in seconds) as the efficiency metric.

6.3 Competing Methods

Greedy Upper Bound (GUB). Assuming that ground truth is known, this method chooses to validate an object that results in the highest ground truth utility gain, i.e.,

$$\theta_i = \underset{\theta_i \in \Theta}{argmax}\ \ (1 - U(D, \mathcal{F}, \mathcal{T}))$$

No other myopic method can outperform the upper-bound method in reporting the `distance_to_ground_truth`.

Random. Among all available actions, this method selects an object at random.

MEU. This method selects the object that has the maximum expected reduction in uncertainty in the absence of domain knowledge and ground truth utility function. Our ultimate aim is to achieve `MEU` reduction in uncertainty.

Local-MEU. This method uses the simple strategy of validating an object with the highest entropy (uncertainty in values) by computing the local entropies of all the objects in a single pass of the database. `Local-MEU` deals with the claims of a single object at a time and, in the process, does not consider other objects.

Approx-MEU. This method selects the next object for validation using the approach described in Section 5.3. The intuition is to choose an object that is the most influential (in terms of reducing uncertainty) in the object-source network.

Network-Approx-MEU. This method reduces the size of the set of potential objects for validation using insights from the EM algorithm and couples it with `Approx-MEU` to select the next object for validation.

Top-k-Approx-MEU. This method considers the top-k objects for computing their impact on each other and then applies `Approx-MEU` on the smaller set of candidate objects.

6.4 Experiments

6.4.1 Basic evaluation on dense synthetic dataset

In this section, we primarily compare the efficiency and effectiveness of `MEU` and `GUB` on small synthetic datasets. Note that we cannot compare `MEU` and `GUB` on real data or at a large scale because `GUB` needs ground truth (i.e., which value is `true`) for all objects whereas our real datasets provide only the ground truth for a small subset of objects. Moreover, both `MEU` and `GUB` become prohibitively expensive to run on datasets with more than 1K objects.

The results of the effectiveness experiment are shown in Figure 3a where we report the distance to ground truth for increasing numbers of validated objects for five validation methods: `Random`, `GUB`, `MEU`, `Local-MEU` and `Approx-MEU`.

This experiment is run on synthetically generated data with 300 objects and 10 sources. Once an object is validated, we do not discard the validation result for the next one. Therefore, we observe a cumulative gain of all validations – the distance to ground truth and uncertainty are finally reduced by hundred percent (when all objects are validated).

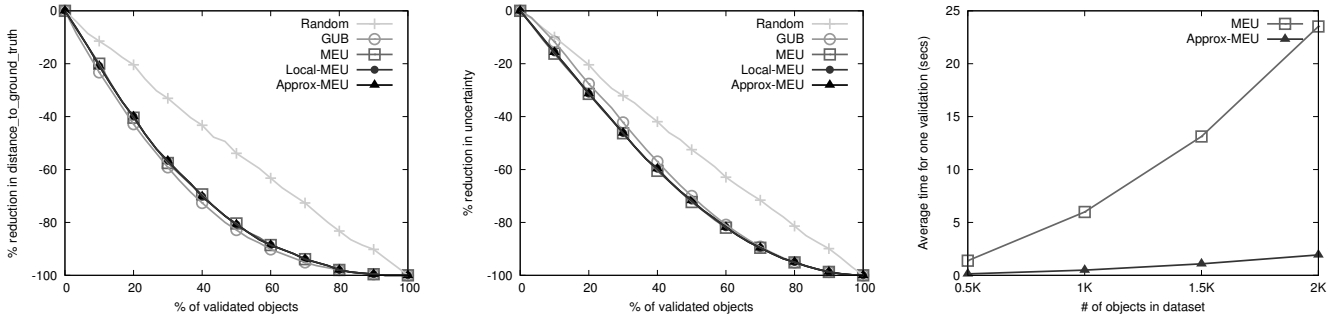
Distance to ground truth. From the five lines shown in Figure 3a we observe that the baseline method, `Random`, linearly decreases the distance by 100% indicating that only the number of validated objects determines its effectiveness. Not surprisingly, `GUB` has a steeper curve than `MEU` since `GUB` uses the ground truth information. `GUB` can be seen as our best myopic performance method (details in Section 6.3). Importantly, we observe that `MEU` performs better than `Random` and is close to `GUB` for all sizes of datasets. This confirms our assumption that `MEU` provides a sound way of selecting objects for validation when no domain information available. Furthermore, `Local-MEU` and `Approx-MEU` closely follow `MEU` indicating their suitability as an alternative to `MEU`.

Change in uncertainty. In Figure 3b, we report the percentage change in uncertainty as the set of validated objects grows. The downward sloping plots indicate that as we validate objects in succession, not only is the uncertainty in validated objects removed but also is the confidence in the probabilities of other objects enhanced. `Random`, by virtue of selecting objects in a truly random fashion, reduces uncertainty linearly. Of greater interest is the observation that `MEU` performs better than `GUB`. This follows from the design of `MEU` that aims at reducing the uncertainty across all objects. On the other hand, `GUB` guarantees the best reduction in the distance to ground truth but has no control over the reduction in uncertainty. `Approx-MEU` and `Local-MEU` have almost the same reduction in uncertainty as `MEU`.

Efficiency. In this experiment, we measure the time taken for one validation using `MEU` and `Approx-MEU`. All experimental results have been obtained on an Intel Core 2 Duo system (2.53GHz, 4GB RAM). For selecting one object, `Random` takes no time and `Local-MEU` takes time proportional to the dataset size. Note that `GUB` cannot be implemented in practice. We performed this experiment on a number of synthetically generated datasets. The results of the efficiency experiment are shown in Figure 3c.

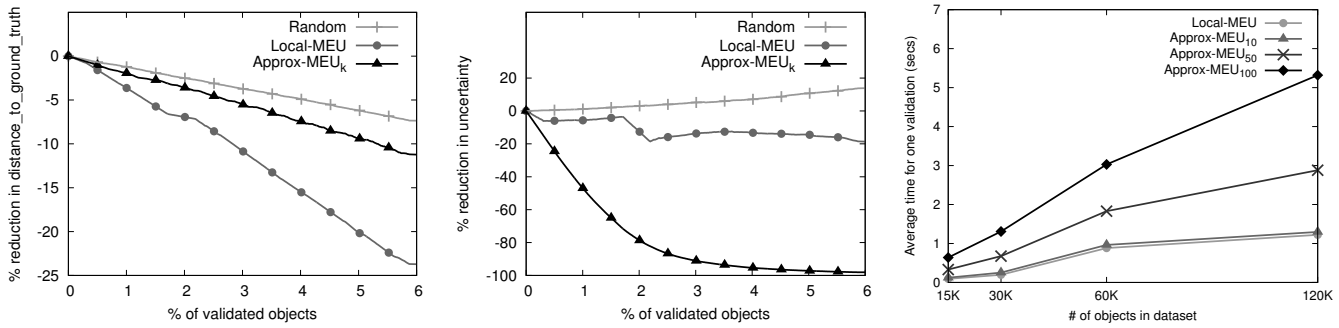
We observe that the time needed for one validation in `MEU` grows rapidly while `Approx-MEU` slashes the time by about one order of magnitude and the effect is more pronounced in larger datasets. Our goal for efficiency is to provide an online validation time such that the users of a truth finder could interact with it. As a result, we conclude that `MEU` cannot be used for typical datasets that truth finders deal with as they have around a hundred thousand objects. From a theoretical standpoint, the time complexity of `MEU` is based on the time complexity of the iterative EM algorithm (we run EM for each available object) – a step that is completely eliminated in `Approx-MEU`.

Experiment Takeaways. (1) `MEU` provides a sound way of selecting objects for validation, one that is close to the ground-truth-based method, `GUB`. We claim that `MEU` is the best strategy to select objects if no domain information is available. (2) `MEU` has an extreme computation cost and can-



(a) Reduction in distance to ground truth. (b) Reduction in uncertainty. (c) Efficiency of `MEU` and `Approx-MEU`.

Figure 3: Effectiveness and efficiency plots of all the approaches on dense synthetic data. $|O| = 300$, $|S| = 10$, density=35%.



(a) Reduction in distance to ground truth. (b) Reduction in uncertainty. (c) Effect of k on the time for validation.

Figure 4: Effectiveness and efficiency of the `Local-MEU` and `Approx-MEU`$_k$ on the flights dataset. $|O| = 121,000, |S| = 38$. In 4a-4b, $k = 10, 50, 100$ generate identical plots and are replaced by a single line.

not be used for online validation in large datasets (with more than 1K objects). (3) `Approx-MEU` is a close approximation of and computationally less expensive than `MEU`.

6.4.2 *Performance of* `Local-MEU` *and* `Approx-MEU`

We now present the effectiveness of `Approx-MEU`. We compare the performance of our method against `Random` and `Local-MEU`. We set the parameter $\mathcal{A}_{\mathcal{F}} = 0.8$ in line with the accuracy of the EM algorithm on real datasets. In case there is no object that reduces the uncertainty of the database, we select one that the truth finder is the least confident about.

In Figures 4 and 5, we report the results of the experiments on real datasets. Due to the density of both the datasets, the complexity of the `Approx-MEU` algorithm is $O(|O|^2)$. This limits our ability to apply the method on the flights dataset and we resort to shrinking the set of potential candidates (see Section 5.4). `Local-MEU` has a steeper curve in Figure 4a indicating better performance over both `Random` and `Approx-MEU`$_k$. `Approx-MEU` has higher reduction in uncertainty (Figure 4b) – with as few as 3% of validated objects, `Approx-MEU` reduces the uncertainty of the database by more than 90% whereas `Local-MEU` is able to achieve only 20% reduction. This behavior could be explained as: `Local-MEU` might pick objects that have high entropy based on their votes but low uncertainty based on the probabilities output by the truth finder. On the other hand, `Random` takes the database to a state of higher uncertainty – this is not surprising since it might validate an object can degrade the quality of the objects surrounding it.

Figure 4c presents the efficiency of `Approx-MEU`$_k$ with different values of k as the dataset size increases from $15,000$ to $120,000$. Comparing with Figures 4a and 4b, we can conclude that a smaller value of k is more efficient only in terms of time for validation.

The plots for the book dataset are fascinating because it confirms the power of user feedback in effectively improving the truth finder system. The crossed line representing `Approx-MEU`$_k$ further proves that the parameter k could be tuned to trade effectiveness for efficiency.

Experiment Takeaways. (1) `Approx-MEU` has higher entropy utility gain than `Local-MEU`. (2) For real datasets, `Approx-MEU` can achieve high entropy utility gain with small values of k. (3) `Local-MEU` has greater reduction in the distance to ground truth but does not have effective entropy utility gain.

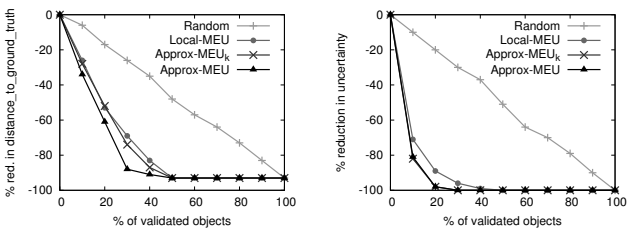
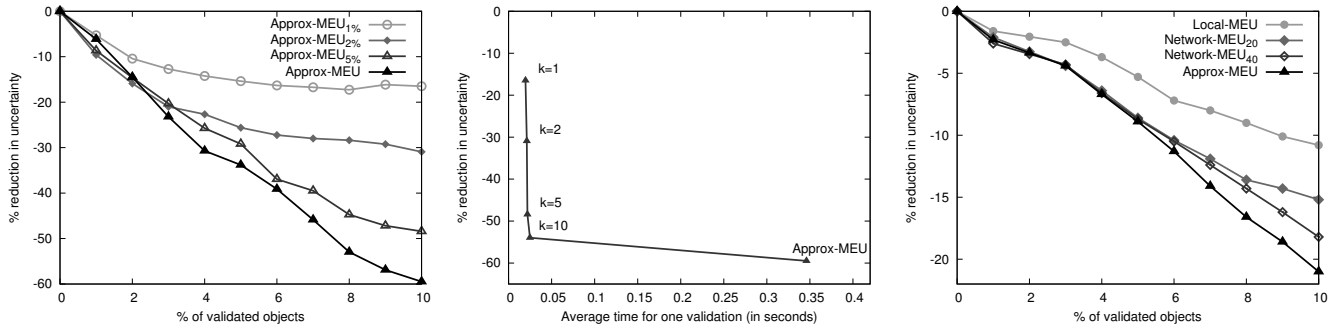


Figure 5: Effectiveness of `Local-MEU` and `Approx-MEU` on the Books dataset. $|O| = 1,263$, $|S| = 894$, $k = 100$.



(a) Effect of varying k on the effectiveness of `Approx-MEU`$_k$.

(b) Effectiveness vs. efficiency trade-off across different values of k.

(c) Effectiveness of `Network-MEU` vs. `Local-MEU` and `Approx-MEU`.

Figure 6: Effect of shrinking the set of candidates on synthetic sparse datasets. (a, b) $|O| = 2,025, |S| = 3,960$ (c) $|O| = 9,00, |S| = 1,740$. Density= 10% for both the datasets.

6.4.3 *Effect of shrinking the candidate set size on network data*

In this section, we compare the methods described in Section 5.4 to `Approx-MEU` on sparse synthetic datasets. Figure 6a demonstrates the effect of tuning the parameter k while considering the top-k objects ranked in decreasing order of uncertainties over truth finder probabilities. From top to bottom, the lines represent `Approx-MEU`$_k$ with increasing k and incrementally get closer to `Approx-MEU`. The result is as expected – the greater the value of k, the more the number of objects considered for validation and hence, the closer the method gets to `Approx-MEU`.

Further, Figure 6b presents the trade-off between effectiveness and efficiency of `Approx-MEU`$_k$ for different values of k. The point $k = 2$ represents the case when top-2% of objects were considered for validation – at which point, we were able to reduce uncertainty by about 30% while taking less than one-twentieth of a second. By comparison, the other end tells us that `Approx-MEU` takes the most time for one validation while achieving about 60% reduction in uncertainty. Interestingly, there is a sweet spot at $k = 10$ where the utility gain is very high ($> 50\%$) and one validation takes minimal time.

We now show how understanding the underlying network could help us in identifying "good" sources, in terms of allowing the propagation of change in probabilities, while also reducing the candidate set size. We tune the parameter $|S_g|$ such that the fraction of objects to consider for validation is less than 1, i.e., $|O_p|/|O| < 1$. In Figure 6c, we report the results with varying fractions of $|O_p|$. We start with shortlisting the top 0.5% sources with least votes on objects and go up to the 5%. As a result, $|O_p|$ ranges from 20% to 40%. When all the objects are considered, `Network-MEU` is the same as `Approx-MEU`. As expected, with fewer objects in the list of candidates, `Network-MEU` performs worse than `Approx-MEU` in terms of reduction in uncertainty. In fact, the fewer the objects considered, the worse is the performance with respect to `Approx-MEU`. `Network-MEU` still performs better than `Local-MEU` in early validations.

Experiment Takeaways. (1) On sparse data, `Approx-MEU`$_k$ can achieve effectiveness similar to `Approx-MEU` for small k. (2) The object-source network plays an important role in change propagation.

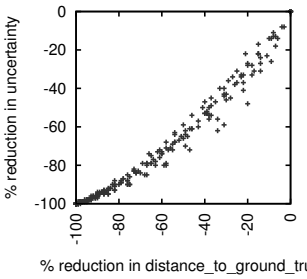


Figure 7: Scatterplot showing the correlation between reduction in `distance_to_ground_truth` and uncertainty. Correlation coefficient, $\rho = 0.8554$.

6.4.4 *Relation between evaluation metrics*

In this experiment, we observed that the effectiveness metrics follow a similar pattern in all the effectiveness experiments for the various datasets in this work. To study formally, we record the metrics for the fundamental methods, `GUB` and `MEU` (since these are our gold standards), on synthetic datasets. During data generation, we vary the number of objects from 100 to 300, number of sources from 5 to 15, source accuracies from 0.6 to 1 and density from 0.4 to 0.5.

In Figure 7, we observe a strong correlation between both the metrics. The Pearson's correlation coefficient of 0.8554 further bolsters the observation. Specifically, as the uncertainty in the probabilities of objects decreases, its distance to ground truth also decreases. We can, thus, confirm that in the absence of ground truth, uncertainty reduction is a good alternative to the ground truth utility function.

7. RELATED WORK

Truth finders. The problem of *data fusion* has been extensively studied in the past. A number of different techniques have been used. Cardinality-based methods [2, 14] counter majority voting by taking into account the reliabilities of sources to determine the correct claim for an object. Approaches based on Bayesian network analysis [32, 9, 6, 22, 29] regard the reliabilities of sources and the correctness of claims as latent variables dependent on each other.

Source dependence. The relationships between sources play an important role for truth discovery. Detecting copiers

[5, 26] is found to be very important in improving the accuracy of truth finders. A recent work [23] studies a broader notion of source dependencies, e.g., sources provide complementary data, use domain-specific extractors, and so on.
To the best of our knowledge, we are the first to leverage user feedback to improve the effectiveness of the state-of-the-art Bayesian-based truth finders. The integration of user feedback into other approaches as well as considering the dependencies among sources are considered as future work.
Leveraging user feedback. The idea of incorporating user feedback has been used in various data management problems. [12] have proposed a pay-as-you-go approach that relies on users to confirm some of the candidate schema matches and incrementally improve the effectiveness of the data integration system. [4] deal with the problem of integrating user feedback into schema matching tasks. [31] solicit user feedback to improve existing automatic data repair techniques in the presence of data integrity rules. In a recent work [21], feedback is utilized to validate generated correspondences between the attributes of database schemas. Alternatively termed as *active learning*, mechanisms for user feedback has a large body of work in the domain of machine-learning [24, 31]. Integrating user feedback into data fusion is, however, a fundamentally different problem and adapting solutions from unrelated domains is technically infeasible.
Crowdsourcing. Collecting feedback using a crowd of workers [8, 17, 18] is an ongoing area of research. Recently, [11] employ user feedback to validate crowd answers. Dealing with uncertainty of feedback, however, is orthogonal to the scope of the present work and considered as future work.
Utility functions. The foremost concern with pay-as-you-go approaches lies in determining the sequence in which user feedback is received. To this end, utility elicitation [1] details classical utility functions in order to narrow down user preferences under uncertainty. [12] uses the value of perfect information [25] to specify the gain of determining the next evidence and a framework to measure the utility of the dataspace to compute this value of perfect information whereas [21] uses Shannon entropy to compute the overall uncertainty of the network.

8. CONCLUSION

This paper proposed a novel decision-theoretic framework to improve the accuracy of truth finders by soliciting and incorporating feedback from users. We defined the utility function of a database and suggested alternatives to the ground truth utility function. We proposed an approach based on expected utility function gain to determine the next object for validation. Further, we presented two entropy-based techniques as alternative solutions. The first method achieves maximum utility function gain at the site of a single object while the second method aims at maximal utility gain across all objects in the database. We coupled the latter method of maximal utility gain with leveraging the network structure and agreement/disagreement of sources in order to achieve a trade-off between effectiveness and efficiency.

Our experimental evaluation showed that our techniques outperform the baseline methods and confirm that soliciting user feedback increases the effectiveness of truth finders. Further, we support online processing time for 100K objects.

We believe that the efficiency of the solutions can be further improved by utilizing the interdependence between objects and sources in greater detail. Another direction is to explore different aspects of the feedback solicitation problem, e.g., allowing uncertainty in user feedback, accepting different granularities of feedback (yes/no answers, complete answers). We plan to explore these issues in future work.

9. REFERENCES

[1] D. Braziunas. Computational approaches to preference elicitation. Technical report, 2006.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW*, 1998.

[3] G. B. E. Chapman and F. A. E. Sonnenberg. *Decision Making in Health Care: Theory, Psychology, and Applications*. Cambridge University Press, 2003.

[4] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD*. ACM, 2001.

[5] X. L. Dong, L. Berti-Equille, Y. Hu, and D. Srivastava. Global detection of complex copying relationships between sources. *VLDB*, 2010.

[6] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: The role of source dependence. *VLDB*, 2009.

[7] X. L. Dong and D. Srivastava. Compact explanation of data fusion decisions. In *WWW*, pages 379–390, 2013.

[8] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin. Crowddb: Answering queries with crowdsourcing. In *SIGMOD*. ACM, 2011.

[9] A. Galland, S. Abiteboul, A. Marian, and P. Senellart. Corroborating information from disagreeing views. In *WSDM*. ACM, 2010.

[10] H. O. Hartley. Maximum likelihood estimation from incomplete data. *Biometrics*, 14(2):174–194, 1958.

[11] N. Q. V. Hung, D. C. Thang, M. Weidlich, and K. Aberer. Minimizing efforts in validating crowd answers. In *SIGMOD*. ACM, 2015.

[12] S. R. Jeffery, M. J. Franklin, and A. Y. Halevy. Pay-as-you-go user feedback for dataspace systems. In *SIGMOD*. ACM, 2008.

[13] A. Kapoor, E. Horvitz, and S. Basu. Selective supervision: Guiding supervised learning with decision-theoretic active learning. In *IJCAI*, 2007.

[14] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *ACM*, 1999.

[15] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

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

[17] A. Marcus, D. Karger, S. Madden, R. Miller, and S. Oh. Counting with the crowd. In *PVLDB*, 2013.

[18] B. Mozafari, P. Sarkar, M. Franklin, M. Jordan, and S. Madden. Scaling up crowd-sourcing to very large datasets: A case for active learning. *PVLDB*, 2014.

[19] R. M. Neal and G. E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*. Springer, 1998.

[20] J. V. Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton Univ. Press, 1944.

[21] Q. V. H. Nguyen, T. T. Nguyen, Z. Miklos, K. Aberer, A. Gal, and M. Weidlich. Pay-as-you-go reconciliation in schema matching networks. In *ICDE*, 2014.
[22] J. Pasternack and D. Roth. Latent credibility analysis. In *WWW*, 2012.
[23] R. Pochampally, A. Das Sarma, X. L. Dong, A. Meliou, and D. Srivastava. Fusing data with correlations. In *SIGMOD*. ACM, 2014.
[24] N. Rubens, D. Kaplan, and M. Sugiyama. Active learning in recommender systems. In *Recommender Systems Handbook*. Springer, 2011.
[25] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2 edition, 2003.
[26] A. D. Sarma, X. L. Dong, and A. Halevy. Data integration with dependent sources. In *EDBT*, 2011.
[27] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 2001.
[28] D. Wang, T. F. Abdelzaher, L. M. Kaplan, and C. C. Aggarwal. Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications. In *ICDCS*, 2013.
[29] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. On truth discovery in social sensing: A maximum likelihood estimation approach. In *IPSN*, pages 233–244, 2012.
[30] J. M. Winn and C. M. Bishop. Variational message passing. In *Journal of Machine Learning Research*, 2005.
[31] M. Yakout, A. K. Elmagarmid, J. Neville, M. Ouzzani, and I. F. Ilyas. Guided data repair. *VLDB*, 2011.
[32] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. In *KDD*, 2007.

APPENDIX

A. CHANGE PROPAGATION BEYOND ONE HOP

In this section, we compute the change in probabilities in an object, o_j, that is more than one hop away from the validated object, o_i.

First, the change in probabilities of o_i are propagated to the sources that provide claims about it. This changes the accuracies of the sources by increasing the accuracy of those that provide a `true` claim and reducing the accuracy of those that provide an incorrect claim. From Equation 5, if source s provides claim v_i^l about object o_i, its accuracy changes as

$$\Delta A(s) = \frac{\Delta p_i^l}{N(s)}$$

Let us represent Equation 2 for object o_j as:

$$p_j^r = \frac{q}{t}$$

Rearrange the terms to obtain:

$$p_j^r t = q = \prod_{s \in S(v_j^r)} \frac{(|V_j| - 1|)A(s)}{1 - A(s)} \tag{13}$$

Now, let us compute the change in the quantity q following a few steps:

$$\log q = \sum_{s \in S(v_j^r)} \log \frac{(|V_j| - 1|)A(s)}{1 - A(s)}$$

Differentiating both sides, $\frac{dq}{q} = \sum_{s \in S(v_j^r)} d\left(\log \frac{(|V_j| - 1|)A(s)}{1 - A(s)}\right)$

We simplify an individual term in the summation to:

$$d\left(\log \frac{(|V_j| - 1|)A(s)}{1 - A(s)}\right) = \frac{dA(s)}{A(s)(1 - A(s))}$$

This transforms dq to: $dq = q\left(\sum_{s \in S(v_j^r)} \frac{dA(s)}{A(s)(1 - A(s))}\right)$

We express the change in probabilities of o_j by computing the first derivative of Equation 13 (as in Section 5.3):

$$p_j^r(dt) + (dp_j^r)t = dq$$

where t can be expressed as a sum of terms, t_k, similar to q for each $v_j^k \in V_j$. This differential equation can be seen as:

$$p_j^r \left(\sum_{v_j^k \in V_j} t_k \sum_{s \in S(v_j^k)} \frac{dA(s)}{A(s)(1 - A(s))}\right) + (dp_j^r)t = q\left(\sum_{s \in S(v_j^r)} \frac{dA(s)}{A(s)(1 - A(s))}\right)$$

Rearranging appropriately and replacing q/t by p_j^r,

$$dp_j^r = p_j^r \left(\sum_{s \in S(v_j^r)} \frac{dA(s)}{A(s)(1 - A(s))}\right) - p_j^r \left(\sum_{v_j^k \in V_j} \frac{t_k}{t} \sum_{s \in S(v_j^k)} \frac{dA(s)}{A(s)(1 - A(s))}\right)$$

We would like to analyze the upper bound of dp_j in order to get an idea of the maximum change that o_i would effect upon o_j. In the following, we follow a step-by-step conclusion of the same.

$$\begin{aligned}
|dp_j^r| &\le p_j^r \left|\sum_{s \in S(v_j^r)} \frac{dA(s)}{A(s)(1 - A(s))}\right| \le p_j^r \sum_{s \in S(v_j^r)} \left|\frac{dA(s)}{A(s)(1 - A(s))}\right| \\
&\le p_j^r |S(v_j^r)| \left|\frac{dA(s)}{A(s)(1 - A(s))}\right|_{max} \\
&\le p_j^r |S(v_j^r)| \left|\frac{dp_i^t}{N(s)A(s)(1 - A(s))}\right|_{max} \\
&\le p_j^r |S(v_j^r)| \left|\frac{dp_i^t}{N'A'(1 - A')}\right|_{max}
\end{aligned}$$

where $N' \le N(s)$ is the least number of objects any source votes for and A' is the accuracy of a source that yields the minimum for the function $A(s)(1 - A(s))$.

Real datasets are often faced with the situation of few sources providing information about too many objects. As a result, N' is usually more than half of the number of objects in the dataset. This, coupled with p_j, dp and $A'(1-A')$, contributes to the change in the probabilities of the object

one hop away being much less than the change in the probabilities of the validated object.

For an object, o_k, two hops away from the validated node, following similar analysis, if o_k is reachable from o_i through o_j, we could say that

$$\begin{aligned} |dp_k^l| &\leq \left(p_k^l |S(v_k^l)| \left| \frac{dp_j^r}{N' A'(1-A')} \right|_{max} \right) \\ &\leq \frac{dp_i^t}{N'^2} \left(\left| \frac{p_k^l p_j^r |S(v_k^l)||S(v_j^r)|}{(A'(1-A'))^2} \right|_{max} \right) \end{aligned}$$

We observe an exponential decay of the changes in probability distributions as we move away from the validated node. More specifically, the changes in probability distributions in the first hop are significantly higher than those from the second hop and so on. This is due to the sole reason that a typical source provides information about a large number of objects in the dataset.