The VLDB Journal (2023) 32:475–500
https://doi.org/10.1007/s00778-022-00757-x

REGULAR PAPER

Enhancing domain-aware multi-truth data fusion using copy-based source authority and value similarity

Fabio Azzalini[1,2] · Davide Piantella[1] · Emanuele Rabosio[2] · Letizia Tanca[1]

Received: 29 November 2021 / Revised: 18 June 2022 / Accepted: 21 June 2022 / Published online: 19 July 2022

Abstract
Data fusion, within the data integration pipeline, addresses the problem of discovering the true values of a data item when multiple sources provide different values for it. An important contribution to the solution of the problem can be given by assessing the quality of the involved sources and relying more on the values coming from trusted sources. State-of-the-art data fusion systems define source trustworthiness on the basis of the accuracy of the provided values and on the dependence on other sources, and recently it has been also recognized that the trustworthiness of the same source may vary with the domain of interest. In this paper we propose STORM, a novel domain-aware algorithm for data fusion designed for the multi-truth case, that is, when a data item can also have multiple true values. Like many other data-fusion techniques, STORM relies on Bayesian inference. However, differently from the other Bayesian approaches to the problem, it determines the trustworthiness of sources by taking into account their authority: Here, we define authoritative sources as those that have been copied by many other ones, assuming that, when source administrators decide to copy data from other sources, they choose the ones they perceive as the most reliable. To group together the values that have been recognized as variants representing the same real-world entity, STORM provides also a value-reconciliation step, thus reducing the possibility of making mistakes in the remaining part of the algorithm. The experimental results on multi-truth synthetic and real-world datasets show that STORM represents a solid step forward in data-fusion research.

Keywords Data integration · Multi-truth data fusion · Source authority · Copy detection · Value similarity

1 Introduction

In the recent years, an amazing amount of data that are generated by users and machines, and especially the tendency to transform every real-world interaction into digital data, have led to the problem of how to make sense of them. In this scenario, the number of data sources that can provide information relevant for a query increases dramatically even in very specific contexts, and each of these sources can store a previously unimaginable amount of data.

When many sources describe the same data items, it is virtually inevitable that conflicts arise. *Data integration* tackles this issue and operates according to the following three steps: *(i)* when the data sources have a schema, *schema alignment* has the purpose of aligning different sources' schemas and maps the attributes that have the same semantics to one another; *(ii) entity resolution* has the purpose of finding, across the data sources, the records that represent the same entities; and *(iii) data fusion* has the purpose of deciding the true value(s) of a data item (from now on called "object") when multiple ones are provided by the different sources. This work addresses the last phase, i.e., data fusion [3,4,10,23].

Past research on data fusion has widely shown that majority voting among the available sources is not enough to obtain good quality results. In particular, in [22], the authors demon-

✉ Emanuele Rabosio
emanuele.rabosio@polimi.it

Fabio Azzalini
fabio.azzalini@polimi.it

Davide Piantella
davide.piantella@polimi.it

Letizia Tanca
letizia.tanca@polimi.it

[1] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Via G. Ponzio 34/5, I-20133 Milano, Italy

[2] Center for Health Data Science, Human Technopole, Viale R. Levi-Montalcini 1, I-20157 Milano, Italy

Table 1 Authors provided by the data sources in our bookstore running example

Source	Book	Category	**Authors**
S_1	Book1	Literature	Jean Cooney, John Golder,Margaret Williams
	Book2	Literature	Clive Cussler, Graham Brown
	Book3	Literature	James Patterson
	Book4	History	Curtis Cate
S_2	Book1	Literature	Jean Cooney, Johmx Golder,Margaret Williams
	Book2	Literature	Clive Cussler, Paul Kemprecos
S_3	Book1	Literature	J. Cooney, John Golder
	Book2	Literature	Clive Cussler
	Book5	History	Sarah Bradford

strated that, even in stock exchange and flight scenarios, which are contexts usually deemed highly reliable, 70% of the objects have more than one value provided, and only 70% of the correct values are provided by the majority of the sources. Therefore, algorithms more sophisticated than majority voting are needed and are proposed.

Data-fusion algorithms can be divided into two subclasses: single-truth and multi-truth ones, the latter denoting the case when an object may have multiple true values. If we look at this from the viewpoint of databases theory, we have to say that these tables do not satisfy the first normal form. However, in practice, modern databases often contain non-atomic values in one attribute, considering them as atomic. Such scenarios are common in everyday life, where many actors play in a movie, or a book may have several authors. As a consequence, we decided to design our model to work also with *multi-valued attributes*[1], which generates a very challenging problem.

Running Example We consider a running example in the context of online bookstores. Each bookstore is a data source, and may record several books; for each book, a bookstore specifies one or more authors. The category of the book is known as well. Our example (shown in Table 1) includes three bookstores (S_1, S_2, S_3) providing information about four books in two categories (literature books and history books). Note that the bookstores provide conflicting information, for instance, S_1 and S_3 specify the author "John Golder" for Book1, while S_2 indicates "Johmx Golder," which probably contains typos. Moreover, all bookstores supply Book2, but only S_1 specifies "Graham Brown" among its authors. The aim is to discover the correct set of authors for each book. Therefore, we are facing a *multi-truth* problem.

The data-fusion literature has recognized the importance of determining the trustworthiness of the individual sources in order to correctly decide the true values. In particular, DART [26], a relatively recent state-of-the-art algorithm, has confronted the multi-truth scenario by learning the trustworthiness of sources in a Bayesian framework, considering that the trustworthiness of a source may vary with the domain of interest, e.g., a data source that is reliable about horror books might provide wrong information about history books. However, DART is built on the assumption that sources are independent of one another, which is a clear oversimplification of the real world; indeed, when thousands of websites describe the same book, it is unrealistic to think that none of them has copied from some of the others.

Moreover, note that the values of the objects are often constituted by textual strings, and that the same entity of the real world may be represented by several different such strings. For instance, the strings "Stephen King" and "Stephen Edwin King" are very likely to refer to the same writer, and therefore they should be considered as equivalent variants. The identification of variant values has been scarcely considered in the past data-fusion literature, and, to the best of our knowledge, it has never been taken into account by multi-truth approaches.

In this paper we propose STORM, an improved algorithm for domain-aware multi-truth data fusion. Similarly to DART, which is the state-of-the-art domain-aware algorithm, STORM learns the trustworthiness of sources in a Bayesian framework but, differently from it, it relaxes the assumption of source independence. In particular, in STORM the trustworthiness is determined on the basis of the *authority* of sources, where authoritative sources are defined as the ones that have been copied by many others. In fact, the key idea is that, in general, when source administrators decide to copy data, they will choose the sources that they perceive as most trustworthy. In addition, STORM also includes a novel technique to cluster together variant values when they are represented by textual strings, relying on a novel token-based similarity measure.

To summarize, in this paper we make the following contributions:

– We present STORM, a new unsupervised, domain-aware algorithm to discover the true values of objects, exploiting Bayesian inference and handling the more complex

[1] The value of these attributes usually contains a special character (e.g., ";", "&") used as separator.

scenario of multi-truth. Moreover, with STORM we can compute the directional copying probabilities between sources, and positively reward the sources on the basis of their authority, which in turn is determined using those probabilities.
– We propose a novel similarity measure (exploited in the value-reconciliation step of STORM) to identify and cluster together the textual strings representing variant values. It is the first time that this approach is used in a multi-truth data-fusion algorithm.
– We demonstrate the effectiveness of the technique by means of an extensive experimental campaign using one synthetic and three real-world datasets in the books, movies and flights scenarios.

Paper Structure The paper is organized as follows. Section 2 contains the related work. Section 3 formally defines the problem and the employed data model and overviews the phases composing the methodology. Section 4 describes the STORM's value reconciliation phase, while Sect. 5 explains the authority-based Bayesian inference based on copy detection. Section 6 presents the experiments and, finally, Sect. 7 concludes the paper.

2 Related work

The basic and most intuitive technique currently used to perform data fusion is *majority voting*, according to which the value(s) claimed by most sources are selected as true. However, given the known shortcomings [22] of this basic method, different research approaches have been proposed in the last decade, resulting in more complex data-fusion algorithms.

We now analyze some of these algorithms from the literature. All of them are more or less related to a voting strategy on the values provided by each source, where each source is assigned a different vote weight depending on its trustworthiness, being the latter typically not known a priori. These algorithms are therefore iterative, which means that at each iteration they refine the measures assessing the truth of the values and the trustworthiness of the sources, until convergence. Many strategies perform the iterative computation within a Bayesian framework. As mentioned in the Introduction, most algorithms focus on the single-truth scenario, but some recent techniques also support the multi-truth case. Since we aim at proposing an unsupervised technique, supervised and semi-supervised approaches (e.g., [9,28,33,45]) are omitted from the discussion.

In the following, we begin with describing the single-truth methods (Sect. 2.1) and then move to the multi-truth ones (Sect. 2.2).

2.1 Single-truth methods

A set of single-truth data-fusion algorithms has been inspired by the analysis of links in the web domain. HubAuthority [17] computes the trustworthiness of each source as the sum of the votes obtained by the values that it provides, when the vote of a value is the sum of the trustworthiness of its providers; in this case the trustworthiness of a source is influenced by the number of values that it specifies. AvgLog [31] is similar to HubAuthority and tries to mitigate the impact of the number of values provided by each source by scaling the trustworthiness through a logarithmic factor. Invest [31] uniformly distributes the trustworthiness of a source among its provided values and computes it as a weighted sum of the votes in the claimed values. PooledInvest [31] modifies Invest by scaling linearly the vote count of the values.

Paper [13] considers that, when a source specifies a value for an object, it implicitly votes against the other ones and proposes three algorithms that iteratively estimate the truthfulness of facts and the trustworthiness of sources. The cosine algorithm computes the trustworthiness of a source on the basis of the cosine similarity between a vector representing the votes of the source and a vector describing the currently predicted truth. 2-Estimates, on the contrary, evaluates the trustworthiness by relying on the average error of the claimed facts with respect to the currently predicted truth. Finally, 3-Estimates refines 2-Estimates by also introducing a measure of how hard it is to obtain each data record.

Many algorithms employ Bayesian inference to predict the veracity of each value and the trustworthiness of the sources. TruthFinder [44] applies Bayesian analysis to compute the probability of a value being true, conditioned on the observation of the values provided by all the sources; the similarity between values is also considered, by increasing the vote count of a value on the basis of the vote counts of the similar ones. Accu [7] assumes a uniform distribution of false values and computes the accuracy of a source as the average probability of its values being true; then, Bayesian inference is used to determine the truth of the values. The paper [7] also proposes two enhancements of Accu: AccuSim considers the value similarities as in TruthFinder, while AccuCopy introduces the modeling of correlation between sources. PopAccu [9] extends Accu by removing the assumption of uniform distribution of false values. GTM [50] uses a Bayesian model designed specifically for numerical data. LCA [32] is an approach focused on having a clear semantics, easily to be analyzed and adapted; in more detail, it is based on a probabilistic model where the truth of a claim is a latent variable, and the credibility of a source is represented by model parameters. MultiLayer [8] jointly estimates the correctness of facts and the accuracy of sources using inference in a Bayesian probabilistic graphical model. IATD [47] determines the trustworthiness of a claim by a source considering also the

trustworthiness of the sources that influence the one making the claim. SlimFast [36] introduces a framework, based on Bayesian inference and statistical learning, capable to exploit the domain features reflecting the reliability of sources; also, SlimFast can automatically choose the best algorithm for the learning task. LTD [49] defines a probabilistic graphical model separating the trustworthy and untrustworthy components in each source.

Further relevant approaches include CRH [21], dealing with heterogeneous data types, CATD [20], which considers the issues related to sources providing just few items, and ETCIBoot [41], which can compute confidence intervals for the values provided by sources. More recently, CTD [43] formulates truth discovery as an optimization problem, while CASE [29] builds a network including source–claim and source–source relationships and embeds it in a low-dimensional space where truth discovery can be conveniently performed. The paper [27] studies data fusion within Linked Data, [25,30,48] analyze the crowdsourcing domain, and [42,46] deal with social networks. Paper [5] describes optimization methods to select significant sources.

The techniques discussed so far, though containing interesting elements, focus on single-truth data fusion, and therefore cannot solve effectively the multi-truth problem we are considering.

2.2 Multi-truth methods

The first approach addressing the multi-truth scenario is LTM, proposed by Zhao et al. [51]. LTM builds a probabilistic graphical model where source quality and value truth are treated as latent variables and performs inference through Gibbs sampling; the model estimates a probability for every value associated with an object, and the values with probabilities greater than, or equal to 0.5, are considered true. LTM postulates that the sources are independent of one another, neglecting the effect of copying.

Subsequently, Wang et al. have proposed MBM [38], which tackles multi-truth data fusion by introducing a copy-detection phase to discover dependencies between sources. It computes, for each group of sources and set of values, an *independence score*, which is then used to discredit in the voting phase the sources that do not provide their values independently. The two main drawbacks of this method are the assumption that there is no mutual copying between sources in the whole dataset and the fact that the algorithm cannot distinguish the direction of copying. A variant of this technique [39] neglects copy detection but considers balance between positive and negative claims of a source (i.e., the values provided by the source and the values provided only by other sources, respectively), and value co-occurrence.

The state-of-the-art algorithms for multi-truth data fusion are SmartVote [11], proposed by the same research group of MBM, and DART [26], by Lin et al.

SmartVote determines source trustworthiness on the basis of a concept of authority defining a source as trustworthy if its claims are endorsed by many other sources. Trustworthiness scores are derived using random walks on graphs in which nodes represent sources and edge weights represent endorsement probabilities between sources. The probabilities are computed taking into account also object popularity and long-tail factors. In SmartVote, the trustworthiness of a source is entirely based on its authority.

DART, on the contrary, is based on an iterative domain-aware Bayesian approach: The key intuition is that a source may have different quality levels in different domains of interest. For each source, the authors define the *domain expertise score*, measuring the source's experience in a given domain of interest, and use it to improve the importance of votes coming from sources that are expert in the given domain of interest. The main shortcoming of DART, like LTM, is the assumption of independence of the sources.

Our algorithm, STORM, is based on a domain-aware Bayesian framework which relaxes the source-independence assumption. Specifically, our method expands the framework presented in DART by adding the following contributions: (i) we designed *source authority*, a novel score that is used to recognize trustworthy sources by analyzing their copying behaviors; (ii) we enhanced the copy-detection methodology presented in MBM, introducing the possibility to identify, given two related sources, which one is the copier. Differently from SmartVote, however, STORM does not weigh sources only on the basis of their (domain-unaware) authority; rather, authority concurs to shape the trustworthiness of sources, along with their expertise, in a domain-aware Bayesian scenario. Moreover, STORM is also enriched with a value-reconciliation phase leveraging a new measure of value similarity. The use of a value-reconciliation step, to the best of our knowledge, has never been considered in the multi-truth data fusion scenario. In the experimental section we will compare STORM with LTM, SmartVote and DART, showing its superior performance.

3 Data-fusion framework

This section introduces our data model and formally defines the multi-truth data-fusion problem tackled by our approach. We also give an overview of the two phases composing our strategy.

We consider a set $\mathscr{S}$ of data *sources* and a set $\mathscr{O}$ of *objects*. A source $s \in \mathscr{S}$ may specify one or more *values* for an attribute of an object. Similarly to other data-fusion algorithms, the method we propose is concentrated on one

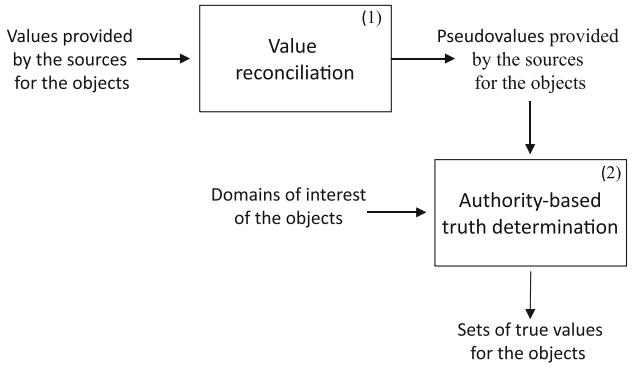

Fig. 1 Phases of our data-fusion strategy

attribute at a time (e.g., deciding the true values of all the authors of a book).[2] The set of values provided by a source s for the object o is denoted as $V_s(o)$, while $V(o)$ is the set of all the values provided by any source of o; therefore, $V(o) = \bigcup_s V_s(o)$. Without loss of generality, we will refer to the values of the attribute of interest of the object o simply as the values of the object o. Let $O(s) \subseteq \mathscr{O}$ be the set of objects for which a source s provides values. The sources may make mistakes, so a value provided by a source for an object may be *true* or *false*. Finally, an object belongs to one or more *domains of interest* of a set $\mathscr{D}$, e.g., a book may belong to multiple categories.

In our running example on bookstores, the *sources* are online bookstores, the *objects* are books sold by the bookstores and the *attribute of interest* is the author. The *values* are the authors of the books; each book may have multiple authors, and the sources may specify correct or wrong authors for the books, possibly in conflict with each other. The *domain of interest* is represented in Table 1 by the attribute Category (e.g., literature or history).

It is now possible to formally define the problem considered in this paper:

Problem statement Let $\mathscr{S}$ be a set of data sources, each providing one or more values of the attribute of interest for objects in the set $\mathscr{O}$, where each object belongs to one or more domains of interest in the set $\mathscr{D}$. Then, the *domain-aware multi-truth data-fusion* problem is defined as *deciding the set of true values for each object in* $\mathscr{O}$.

Our approach to the domain-aware multi-truth data-fusion problem is composed of two phases, summarized in Fig. 1: *value reconciliation* and *authority-based truth determination*.

The *value reconciliation* phase (detailed in Sect. 4) takes as input the sources and the values they specify for the objects, and, for each object, clusters together the values – provided by different sources – that are recognized as variants representing the same real-world entity. For instance, this phase is expected to place "J. R. R. Tolkien" and "John R. R. Tolkien" in the same cluster. At the end of this phase, each source is associated no more with a set of values, but with a set of *pseudovalues*, where each pseudovalue represents a cluster of values that will be considered as a single value in the next phase. Value reconciliation simplifies the task of the subsequent phase, because the number of pseudovalues for an object is in general much smaller than the number of its values.

The *authority-based truth-determination* phase, detailed in Sect. 5, receives as input the association between sources and pseudovalues, and the domains of interest to which the objects belong. A Bayesian inference algorithm leveraging the concept of source authority is employed to determine the set of true pseudovalues for each object. Then, for each true pseudovalue, a representative value is also selected and output.

Table 2 summarizes the notation used in the following sections.

4 Value reconciliation

Our data-fusion method, like many others, exploits a voting strategy to assign each object its own set of true values. It is thus very important that all the votes intended for a certain value are actually attributed to it, despite its different representations. Unfortunately this is rarely the case, given the multitude of ways the true values can be addressed and the presence of dirty values in nowadays data sources [1]. In this context, being able to recognize when different values refer to the same concept is crucial in order to have good performances.

To better introduce the problem and the challenges it encompasses, let us analyze the situation emerging in the real-world Books dataset that we will use for the experimental evaluation in Sect. 6. In particular, let us consider the book "The Hidden Staircase" whose different values for the "author" attribute are shown in Table 3. In this example, all the different values refer to the same person (i.e., Carolyn Keene), and the application of the value reconciliation step will greatly improve the job of the subsequent Bayesian inference procedure. Note that reconciling the values beforehand also allows to better detect copying behaviors between sources, because a source may copy values from another one and then adjust them to its own format, somehow masking the copy.

Analyzing the example reported in Table 3, we can notice two different types of heterogeneity present in the values:

– *Elementary differences*: the use of uppercase and lowercase letters, punctuation marks often used to divide name

[2] The Conclusion and Future Work section contains some comments on the problem of dealing with more than one attribute.

Table 2 Notation used throughout the paper

Notation	bf Description
$\mathscr{S}$	Set of all sources
$\mathscr{O}$	Set of all objects
$\mathscr{D}$	Set of all domains of interest
$V_s(o),\ C_s(o)$	Set of the values (resp. pseudovalues) provided by source s for object o
$V(o),\ C(o)$	Union of the sets of values (resp. pseudovalues) provided by any source for object o
$\overline{C_s(o)}$	Set of all pseudovalues not provided by source s for object o, but provided by other sources
$S_o(v),\ S_o(c)$	Set of sources that provide value v (resp. pseudovalue c) for object o
$S_o(\bar{c})$	Set of sources that provide an object o but not its pseudovalue c
$O^d(s)$	Set of objects provided by source s in the domain of interest d
$s_i \xrightarrow{c} s_j$	Event whereby source s_i is copying pseudovalue c from source s_j
$s_i \perp_c s_j$	Event whereby sources s_i and s_j provide pseudovalue c independently
$s_i \xrightarrow{o} s_j$	Event whereby source s_i is copying from source s_j a common pseudovalue for object o
$s_i \xrightarrow{\mathrm{d}} s_j$	Event whereby source s_i is copying from source s_j in domain of interest d
Θ_{ij}^d	Set of objects in domain of interest d common between the sources s_i and s_j
$\psi(o)$	Observation of the pseudovalues provided for object o
ψ_c	Observation that two sources agree on the same claim (positive or negative) about a specific pseudovalue c
ψ_o	Observation that two sources provide the same object o
$A_d(s)$	Authority of source s in domain of interest d
$e_d(s)$	Expertise of source s in domain of interest d
$c_s(c)$	Confidence score of pseudovalue c provided by source s
$\sigma(c)$	Veracity score of pseudovalue c
$\tau_d^{pre}(s)$	Precision of source s in domain of interest d
$\tau_d^{npv}(s)$	Negative predictive value of source s in domain of interest d

Table 3 Authors provided by the data sources for the book "The Hidden Staircase" (ISBN: 0448095025), in one of the real-world datasets employed in our experimental evaluation (described in Sect. 6)

Keene, Carolyn
CAROLYN G. KEENE
Carolyn Keene
CAROLYN KEENE
None
Carolyn G. Keene
C. Keene
Keene, C.
Keene
Keene, Carolyn (Author)
Carolyn Keen

and surname or in abbreviations, the order of words composing a name, and unwanted symbols resulting from inaccurate crawling operations.

– *More complex differences*: the common use of abbreviations in the first and the second name, partial information deriving from sources reporting only last names or omitting second names when present, and typos often present in dirty values.

The first type of discrepancy is easy to handle: We employ a simple *data cleaning* pipeline that yields as output a dataset where the majority of the elementary heterogeneities are removed.

The second type of discrepancy is harder to identify and solve. To this scope, we have designed a *reconciliation* algorithm that, exploiting an iterative clustering procedure and a novel string similarity measure, is able to place inside the same cluster all the values referring to the same concept, i.e., in our example, referring to the same author. Each value is then replaced by the *pseudovalue* representing the cluster it belongs to, and the sources instead of voting for values will vote for pseudovalues.

The two procedures we now describe have been specifically designed to work well with datasets containing person names or location addresses. This choice could seem very restrictive for the actual applicability of the method, but it is not: Looking at the most widely used and diversified repository of data-fusion datasets (Luna Dong data-fusion datasets[3]), most of the data sources either belong to one of the two above-stated contexts or contain numeric values that do not require any cleaning or reconciliation. Specifically, our methods exploit the observation that in most cases the differences concern: abbreviations ("C. Keene," "Carolyn

[3] http://lunadong.com/fusionDataSets.htm.

Keene"), provision of a partial value ("Keene," "Carolyn G. Keene"), typos ("Carolyn Keen," "Carolyn Keene") or the presence of a particular context-specific token ("Keene, Carolyn (Author)").

We are aware that also other types of data are available; anyway, ours, like most of the other data-fusion algorithms present in literature, is designed to work with structured data sources featuring values composed of few tokens. We leave the study of data-fusion methods for dealing with long textual descriptions written in natural language, such as movie descriptions or product reviews, to a future extension of this work.

Before presenting our method for tackling the second type of heterogeneity, we describe the simple data cleaning pipeline that we employ to solve the first type. Please note that this first stage will be used as preprocessing step for all the data-fusion methods with which we compare STORM in the experiments of Sect. 6.

4.1 Cleaning the values

The steps of the simple cleaning pipeline responsible for solving the elementary discrepancies are the following:

- Removal of punctuation marks.
- Replacement of all uppercase characters with their lowercase counterparts.
- Sorting the tokens composing the string according to the alphabetical order.
- Removal of context-specific tokens: "illustrator," "author," "translator," "director," etc. Note that the identification of the context-specific tokens to be removed can be performed employing a tf-idf algorithm to find these special words in a semi-automatic fashion.
- Removal of numbers and special symbols.
- Deletion of values composed of more than a pre-determined number of tokens.

4.2 Similarity measure

At this point a natural way of proceeding would be to compare the cleaned values through some string similarity metric and group together the values regarded as similar. We tested seven classic similarity measures (Hamming [14], Levenshtein [19], Jaro-Winkler [40], Jaccard [16], Sørensen [37], Ratcliff-Obershelp [35], and Longest Common Subsequence (LCS) [2]) and discovered that unfortunately these traditional metrics do not work properly in our context. For example:

- *When comparing two values referring to the same author, "carolyn keene" and "keen"*, none of the metrics is able to assign a score higher than 0.5.
- *When comparing two values referring to two different authors, "carolyn keene" and "carolyn brown"*, all the metrics assign a score higher than 0.5.

Therefore, in some cases these metrics show the opposite behavior with respect to the expected one. These examples also emphasize how difficult it is to choose a threshold to discern whether two values refer to the same concept or not.

To solve these problems we decided to introduce *STORM similarity*, a string similarity metric to exploit the specific properties of the values belonging to contexts where abbreviations, provision of a partial value and typos are common. The STORM similarity metric is defined in Algorithm 1: The input is composed of two values, each in its turn possibly composed of multiple tokens; the output is a number between 0 and 1 representing how similar the two values are.

At its core, the algorithm works by counting how many tokens of the two input values are similar, and to accomplish this goal we propose an implementation based on matrices. The algorithm starts by constructing the similarity matrix **S** (Lines 5-15), where the value of each cell, s_{ij}, contains the matching score between the i^{th} token of the first value and the j^{th} token of the second value (e.g., see matrix **S** in Eq. 1).

To determine to what extent two tokens are similar, our similarity algorithm employs two different types of *token-matching*:

- *Perfect token-matching*: when the two tokens are identical;
- *Partial token-matching*: when only the prefix of the tokens is equal; this second matching notion, denoted in Algorithm 1 by the symbol $\approx$, is very important to identify abbreviations and typos.

We assign score 1 to each perfect match. If a partial match is detected, we assign a score equal to the ratio (indicated as $prefix\_\%$ in the algorithm) between the length of the matched prefix and the length of the shortest of the two tokens. Now that we have identified the similar tokens between the two values, we need to find a way to aggregate the information contained in the similarity matrix in order to design a global score, the *value-matching score*, indicating whether the two input values are similar or not.

Observe that there might be cases in which a token of the first value is similar to more than one token of the second value, or vice-versa. For instance, see the similarity matrix **S** for the values "Clive Cussler" and "Clive Cusler" in Eq. 1.

$$\mathbf{S} = \begin{array}{c} \\ Clive \\ Cussler \end{array} \begin{array}{c} \begin{array}{cc} Clive & Cusler \end{array} \\ \begin{bmatrix} 1 & 0.2 \\ 0.2 & 0.5 \end{bmatrix} \end{array} \tag{1}$$

To avoid this undesired behavior we have to make sure that, when we aggregate the scores contained in the similarity matrix, we select at most one token-match value for each row and for each column. In this formulation, our solution can be traced back to the *assignment problem*, a fundamental combinatorial optimization problem.

The *assignment problem* can be generally defined as the task of assigning a number of resources to an equal number of activities so as to minimize the total allocation cost. In our case, we want to create correspondences between the tokens of the two values; specifically, we are not interested in minimizing a cost, therefore we opted for the maximization formulation of the assignment problem instead of its standard form. Moreover, since the two values may not share the same cardinality we rely on the unbalanced version of the assignment problem [34].

Given the similarity matrix created at the previous step, the assignment problem can be mathematically stated as the problem of maximizing the *match score:*

$$M = \sum_{i=1}^{n_i} \sum_{j=1}^{n_j} s_{ij} \cdot x_{ij}$$

where:

- n_i and n_j are the numbers of tokens composing the two input values
- x_{ij} is a variable whose value is 1 if we established a correspondence between the i^{th} token of the first value and the j^{th} token of the second value, 0 otherwise

subject to the following constraints:

- $\sum_{i=1}^{n_i} x_{ij} = 1, j = 1, 2, ..., n_j$; ensuring that each token of the second value matches at most one token of the first value
- $\sum_{j=1}^{n_j} x_{ij} = 1, i = 1, 2, ..., n_i$; ensuring that each token of the first value matches at most one token of the second value

The *Hungarian algorithm* [18] finds an optimal solution to the assignment problem in polynomial time. The method takes as input the similarity matrix and returns as output the correspondences between the tokens that maximize the similarity between the two values (Line 16 of Algorithm 1).

The result of the Hungarian algorithm (M) is the sum of the selected token-matching scores between the two values. We define the value-matching score as M divided by number of tokens in the two values (Line 17 of Algorithm 1).

The two values v_1 and v_2 can be regarded as matching, and hence defined as representing the same concept, if their similarity score $\textsc{Sim}(v_1, v_2)$ is greater than 0.5. The value 0.5 should not to be intended as a threshold to be varied according to the specific application or dataset: It simply defines that two values are similar if the number of tokens they share is greater than half the average number of tokens contained in the two values v_1 and v_2.

Algorithm 1 Compute STORM similarity

```
1: function SIM(v1, v2)
2:     c1 ← |v1|
3:     c2 ← |v2|
4:     S ← [c1][c2]

5:     for all token_i ∈ v1 do
6:         for all token_j ∈ v2 do
7:             if token_i = token_j then
8:                 s_ij ← 1
9:             else if token_i ≈ token_j then
10:                s_ij ← {prefix_%}
11:            else
12:                s_ij ← 0
13:            end if
14:        end for
15:    end for

16:    M = Hungarian(S)
17:    match ← M / Mean(c1,c2)
18:    return match
19: end function
```

Example 1 (STORM similarity) Let us compute the similarity between two values from the Book1 of our running example, "John Golder" and "Johmx Golder." We have a perfect token matching between the second token of the two values and a partial token matching between the first token of the two values; as a result there is one full point for the perfect token match and $\frac{3}{4}$ for the partial token match (since the first three characters of the two tokens are identical, and the shortest token is composed of four characters). We sum these values and divide the sum by the average number of tokens in the two values, in this case 2, getting a STORM similarity of 0.875 which is higher than 0.5 and thus the two values represent the same author.

Regarding the comparison of "carolyn keene" and "keen," which was failed by all the traditional string similarity metrics, STORM rightfully define them as matching. The algorithm finds a partial token match between "keen" and "keene" with a score of 1, then after dividing this score by the average number of tokens in the two values (1.5) the score is 0.667 and hence higher than the threshold (0.5) to consider two values as matching. For a deeper analysis of the STORM similarity measure against the traditional ones we refer the reader to the experiments presented in Sect. 6.6.

4.3 Value reconciliation algorithm

Now that we have a string similarity measure that performs well with the majority of data-fusion datasets, we can describe the iterative clustering algorithm we use to group together the values referring to the same concept in order to produce the pseudovalues. The reconciliation procedure is presented in Algorithm 2.

The process starts by selecting the most frequent value *freq* among the values provided for a specific object (Line 3) and removing it from the values to be reconciled (Line 4). *freq* is then compared with all the other values (Line 7). In case a value matches (Line 8), i.e., the similarity is greater than 0.5, it is placed inside the same cluster as *freq* (Line 9). All the values that matched with *freq* are removed from the set of values to be reconciled (Line 10). The newly created cluster is then added to the pseudovalues of the object under consideration (Line 13), and the algorithm continues selecting the next most frequent value among the ones that have not been reconciled yet (Line 14), comparing it with all the other values still not clustered, and the algorithm stops when the set of values to be reconciled becomes empty (Line 6). The reconciliation is performed for each object, and at the end of the execution of the algorithm each object $o \in \mathcal{O}$ is associated with its set of pseudovalues $C(o)$.

Supposing there are N objects and P values per object, and considering the number of tokens per object as negligible, the complexity of the reconciliation algorithm is $O(NP^2)$. Note also that the computations for the different objects are independent of each other, therefore in a multi-core environment Algorithm 2 can be easily parallelized.

Example 2 (Reconciliation algorithm) Let us apply the value reconciliation algorithm to our running example, starting with Book1. The algorithm proceeds as follows:

- Select the most frequent value. In this case "John Golder," "Jean Cooney" and "Margaret Williams" are all provided by three sources; suppose that we start with "John Golder."
- Compare "John Golder" with all the other values in order to find value matches. The following value match is detected: "Johmx Golder" with score 0.875. Non-matching scores are computed for "Jean Cooney" (0.125), "Margaret Williams" (0), "J. Cooney" (0.5).
- Create a pseudovalue containing "John Golder" and "Johmx Golder."
- Remove the values added to the pseudovalue created in the previous step from the values that still need to be reconciled.
- Select the most frequent value, suppose it is "Jean Cooney."
- Compare "Jean Cooney" with all the values not reconciled yet in order to find value matches. The following value match is detected: "J. Cooney" with score 1. The remaining value ("Margaret Williams") has match score 0.
- Create a pseudovalue containing "Jean Cooney" and "J. Cooney."
- Remove the values added to the pseudovalue created in the previous step from the values that still need to be reconciled.
- The other values and objects are similarly examined, leading to the clustering in Table 4. In the Bayesian step of STORM, the sources instead of voting for the individual values will vote for the pseudovalues identified by the reconciliation algorithm.

Table 4 Pseudovalues provided by the data sources in our bookstore running example

Source	Book	Category	**Pseudovalues**
S_1	Book1	Literature	c_{11}, c_{12}, c_{13}
	Book2	Literature	c_{21}, c_{22}
	Book3	Literature	c_{31}
	Book4	History	c_{41}
S_2	Book1	Literature	c_{11}, c_{12}, c_{13}
	Book2	Literature	c_{21}, c_{23}
S_3	Book1	Literature	c_{11}, c_{12}
	Book2	Literature	c_{21}
	Book5	History	c_{51}

c_{11} = {Jean Cooney, J. Cooney}, c_{12} = {John Golder, Johmx Golder}, c_{13} = {Margaret Williams}, c_{21} = {Clive Cussler}, c_{22} = {Graham Brown}, c_{23} = {Paul Kemprecos}, c_{31} = {James Patterson}, c_{41} = {Curtis Cate}, c_{51} = {Sarah Bradford}

Algorithm 2 STORM's reconciliation

Input: objects $\mathcal{O}$, values provided by all the sources for objects $V(o)$
Output: $\forall o \in \mathcal{O}$, set $C(o)$ of its pseudovalues

```
1: for all o ∈ O do
2:     C(o) ← ∅
3:     freq ← MostFrequent(V(o))
4:     others ← V(o) \ {freq}
5:     cluster_freq ← {freq}
6:     while others ≠ ∅ do
7:         for all v ∈ others do
8:             if Sim(freq, v) > 0.5 then
9:                 cluster_freq ← cluster_freq ∪ {v}
10:                others ← others \ {v}
11:            end if
12:        end for
13:        C(o) ← C(o) ∪ {cluster_freq}
14:        freq ← MostFrequent(others)
15:        others ← others \ {freq}
16:        cluster_freq ← {freq}
17:    end while
18: end for
```

The reconciliation methodology might be enhanced, in a future work, with the addition of a step to exploit the semantic information included in open-source knowledge bases. In this regard, Wikidata, a famous knowledge base, provides for each entity a list of aliases that might be useful to reconcile values with abbreviations, or that feature just a subset of the tokens contained in the complete name of the object. For instance, for "J. R. R. Tolkien," that in Wikidata is identified by unique identifier Q892[4], the following aliases are present: "John Ronald Reuel Tolkien," "John R. R. Tolkien" and "Tolkien." On the other hand, since less famous terms and values containing typos cannot be reconciled in the same way, STORM's reconciliation methodology still provides a general and valid solution able to deal also with values affected by these issues.

Techniques that have a similar goal to the one of our reconciliation strategy are author disambiguation methods in the bibliographic domain. When disambiguating names in the bibliographic context two main approaches are available: *(i)* grouping together citation records referring to the same author (i.e., author grouping methods), *(ii)* directly assigning each citation record to the right author (i.e., author assignment methods) [12]. Both approaches exploit supervised techniques, either by learning similarity metrics or classification algorithms from a labeled training dataset or by interacting with a user during the parametrization of the framework. Author disambiguation methods need citation information such as author/coauthor names, work title, publication venue title, year, and so on. These attributes are usually not sufficient to perfectly disambiguate all the references: Some methods require also additional information (e.g., emails, affiliations, paper headers, etc.). New evidences usually improves the performance of the disambiguation task, but often requires additional effort for extracting all the needed information. By comparing our system with these methods, we can identify two clear differences: (i) Our system works in a completely unsupervised way, no labeled training data are required; (ii) our system just considers the values of the attribute under consideration to perform the reconciliation task, no other information are required. To conclude, the goal of our reconciliation strategy is to improve the performances of the data fusion step by presenting a simple string similarity function that works better than the ones present in literature and commonly used in data cleaning tasks. We agree that, in the cases in which the currently available author disambiguation methods designed for the bibliographic domain can be used, the user should exploit their functionalities. We also believe that in many domains this is not possible and that our solution can provide a substantial improvement in the performances of the data fusion procedure.

[4] https://www.wikidata.org/wiki/Q892.

Once the pseudovalues are created, different strategies can be employed to select the representative of each pseudovalue. The selection of the representative value of the pseudovalues will be discussed in Sect. 5.5.

5 Authority-based truth determination

The authority-based truth-determination phase of STORM receives as input the pseudovalues provided by the sources and the domains to which the objects belong and produces as output the set of true values for each object.

Our truth determination technique relies on Bayesian inference, which is a statistical method often employed to perform data fusion [7–9,26,32,36,38,44,47,49], but substantially enriches the traditional approaches by leveraging source authority. In particular, we exploit *copy-based source authority*, according to which a data source is deemed authoritative if it is often copied by other sources.

Bayesian approaches are very popular in the data fusion literature because they are very effective in modeling the inter-dependence between different quantities, thus favoring their joint iterative estimation. In more detail, in STORM we need to model the inter-dependence between sources' trustworthiness, value veracity and copying probability (then used to compute source authority).

We remark again the importance of the copy-based source authority in determining the true values. Just think to the online bookstores scenario: It is natural that in the different domains of interest there are sources that are more expert and reliable and that they will be the most copied ones. Additionally, in the movies scenario, important and reliable sources like IMDB are clearly considered as a reference by the other players, and thus they are expected to be often copied. Copy-based source authority allows us to gather this behavior and exploit it to improve the veracity estimation. In Sect. 6.5, within the experimental part of the paper, we will also propose a case study based on one of our real datasets permitting to further intuitively understand and appreciate the relevance of the copy-based source authority.

The rest of this section goes as follows. We start providing basic notions and an outline of the authority-based truth determination procedure (Sect. 5.1). Then, the individual steps of the procedure are described in detail (Sects. 5.2 through 5.5). Finally, the complete algorithm is illustrated (Sect. 5.6).

5.1 Procedure outline and basic measures

In order to determine the true values of each object, STORM estimates, for each pseudovalue, a *veracity* score, defined as in [26]:

Definition 1 (Veracity) The veracity score of a pseudovalue c, denoted by $\sigma(c)$, is the probability of c being true.

The veracity is estimated through an iterative Bayesian inference algorithm. In this subsection we first derive the veracity updating formula to be used in the iterations and then sketch the procedure that will be explained in detail throughout the whole Sect. 5. Finally, we present the measures that we employ to assess the trustworthiness of the sources.

Let $\psi(o)$ be the observation of the pseudovalues provided for object o. Applying Bayes' theorem, we can express the probability that a certain pseudovalue $c \in C(o)$ is true, given the observation of the pseudovalues provided for object o, as follows:

$$P(c|\psi(o)) = \frac{P(\psi(o)|c)P(c)}{P(\psi(o))} \tag{2}$$

Note that, according to its definition, the veracity is the probability of c being true, thus it corresponds to the *prior probability* $P(c)$ in Eq. 2. Considering $\bar{c}$ as the event according to which the pseudovalue c is false, substituting $P(c)$ with $\sigma(c)$ and performing some calculations we obtain:

$$\begin{aligned} P(c|\psi(o)) &= \frac{P(\psi(o)|c)\sigma(c)}{P(\psi(o)|c)\sigma(c) + P(\psi(o)|\bar{c})(1-\sigma(c))} \\ &= \frac{1}{1 + \frac{1-\sigma(c)}{\sigma(c)} \cdot \frac{P(\psi(o)|\bar{c})}{P(\psi(o)|c)}} \end{aligned} \tag{3}$$

The veracity value $\sigma(c)$ is refined at every iteration of the algorithm using Eq. 3. $\sigma(c)$ is initialized with a default value that is employed to compute $P(c|\psi(o))$ at the first iteration; in order not to introduce bias in the computation, it is reasonable to initialize the veracity to 0.5. Then, this computed value of $P(c|\psi(o))$ is used as the new $\sigma(c)$ at the subsequent iteration, and an updated estimation of $P(c|\psi(o))$ is derived. The iterative process proceeds until convergence. According to Eq. 3 to provide an updated estimation of the posterior $P(c|\psi(o))$ at each iteration we need to devise a way to compute the likelihoods $P(\psi(o)|c)$ and $P(\psi(o)|\bar{c})$, which are the probabilities of observing $\psi(o)$ when c is true or false, respectively.

The iterative algorithm that we use to update the veracities estimation is sketched in Fig. 2. In more detail, each iteration consists of three steps:

(a) STORM performs *copy detection*, determining the copying probabilities between sources in every domain of interest, considering also the direction of copying. Copy detection is described in Sect. 5.2.
(b) Copying probabilities are used to compute the *authority* of sources in each domain of interest. Source authority computation is described in Sect. 5.3.

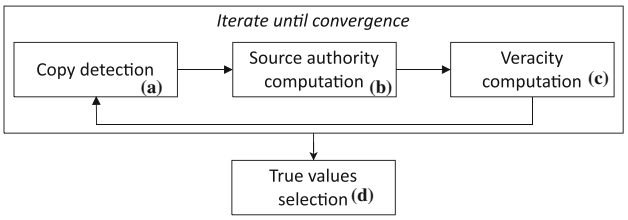

Fig. 2 Steps composing STORM's authority-based truth-determination procedure

(c) The *veracities* of the pseudovalues are computed through Eq. 3; in particular, source authorities are employed to improve the estimation of $P(\psi(o)|c)$ and $P(\psi(o)|\bar{c})$. Veracity computation is described in Sect. 5.4

Then, a final step is performed when the iterative process reaches convergence:

(d) STORM produces the set of *true pseudovalues* for each object by selecting those whose veracity exceeds a veracity threshold, and chooses a representative value for each true pseudovalue. True values' selection is described in Sect. 5.5.

In order to properly perform both copy detection and veracity computation, STORM needs an assessment of the trustworthiness of the sources in the different domains, to know how much their claims about the objects can be trusted. We employ two source trustworthiness indicators, whose values are refined at each iteration.

The first indicator is *precision*, which measures how many pseudovalues provided by a source are actually true. During the iterative process it is not known which are the true pseudovalues, so we use the current estimation of their veracity. Let $O^d(s)$ be the set of objects provided by source s in domain of interest d, and $C_s(o)$ the set of pseudovalues provided by s for object o. *Precision* is defined as follows:

Definition 2 (Precision) The precision $\tau_d^{pre}(s)$ of a source s in domain of interest d is the probability that the pseudovalues provided by s in d are true.

$$\tau_d^{pre}(s) = \frac{\sum_{o \in O^d(s)} \sum_{c \in C_s(o)} \sigma(c)}{\sum_{c \in O^d(s)} |C_s(o)|} \tag{4}$$

As discussed in several works [26,38,51], in the multi-truth scenario the precision measure is not enough to assess the trustworthiness of a data source, because it considers only positive claims. This means that a source providing only correct pseudovalues for the objects it supplies would have precision 1 even if it provides just a subset of the actual true pseudovalues for those objects. In order to evaluate the reliability of a source when it does not provide a certain pseudovalue specified by another source for the same object, we

use – as in [38] – the *negative predictive value*, which measures how many pseudovalues not provided by a source (and provided by other sources for the same objects) are actually false. Let $\overline{C_s(o)}$ be the set of pseudovalues not provided by source s for object o but provided by other sources. The *negative predictive value* is defined as:

Definition 3 (Negative Predictive Value) The negative predictive value $\tau_d^{npv}(s)$ of a source s in domain of interest d is the probability that the pseudovalues not provided by s in d are false.

$$\tau_d^{npv}(s) = \frac{\sum_{o \in O^d(s)} \sum_{c \in \overline{C_s(o)}} (1 - \sigma(c))}{\sum_{o \in O^d(s)} \left|\overline{C_s(o)}\right|} \tag{5}$$

Example 3 (Source trustworthiness) Consider source S_3 in domain of interest $d =$ "literature." Suppose that, after a certain iteration of the Bayesian inference algorithm, we have computed these veracities for the pseudovalues: for Book1: $\sigma(c_{11}) = 0.95$, $\sigma(c_{12}) = 0.85$, $\sigma(c_{13}) = 0.9$; for Book2: $\sigma(c_{21}) = 0.97$, $\sigma(c_{22}) = 0.82$, $\sigma(c_{23}) = 0.2$. Applying Eqs. 4 and 5 we obtain the following precision and negative predictive value:

$\tau_d^{pre}(S_3) = \frac{0.95+0.85+0.97}{3} = 0.9233$

$\tau_d^{npv}(S_3) = \frac{(1-0.9)+(1-0.82)+(1-0.2)}{3} = 0.36$

This means that source S_3 provides correct pseudovalues, but seems to miss other relevant ones.

5.2 Copy detection

We describe now our copy-detection strategy. Our approach is inspired by [38]: specifically, we borrowed their initial considerations on "copying a value" (Eqs. 6, 7, 8 and 9) and improved their methodology in two respects: *(i)* we assign different probabilities to the two directions of copying, which is fundamental to determine source *authority*; and *(ii)* our copy detection strategy is domain-aware.

Most of the data fusion systems that include the study of copying behaviors are based on the assumption that there is no mutual copying between any pair of sources, that is, s_1 copying from s_2 and s_2 copying from s_1 do not happen at the same time [10]. We partially relax this assumption, in fact, our methodology only requires that there is no mutual copying *at domain level*, i.e., if source s_1 copies from source s_2 regarding domain d_1, then s_2 can copy pseudovalues from s_1 only for objects in domains $d_j \neq d_1$.

For each pair of sources (s_i, s_j) in every domain of interest d, the aim is to compute the probability that s_i copies from s_j (i.e., $P(s_i \xrightarrow{d} s_j | \Theta_{ij}^d)$) and that s_j copies from s_i (i.e., $P(s_j \xrightarrow{d} s_i | \Theta_{ij}^d)$) the common pseudovalues they provide for their common objects Θ_{ij}^d in the domain of interest d. To compute these probabilities we begin with examining the various copying relations between s_i and s_j for a specific common pseudovalue.

5.2.1 Copying a pseudovalue

Let c be a pseudovalue for object o in the domain of interest d, provided by both sources s_i and s_j. Let $s_i \xrightarrow{c} s_j$ denote the event according to which s_i copies c from s_j. Let $s_i \perp_c s_j$ be the event according to which s_i and s_j provide c independently of each other, and ψ_c be the observation that s_i and s_j have the same claim (positive or negative) about pseudovalue c. We compute the probability of observing ψ_c in different cases of source dependence and truthfulness of c.

First, similarly to [38], we state that, if s_i copies from s_j or the other way round, then they have the same claim about c, no matter the veracity of c:

$$\begin{cases} P(\psi_c | s_i \xrightarrow{c} s_j,\ c) = P(\psi_c | s_i \xrightarrow{c} s_j,\ \bar{c}) = 1 \\ P(\psi_c | s_j \xrightarrow{c} s_i,\ c) = P(\psi_c | s_j \xrightarrow{c} s_i,\ \bar{c}) = 1 \end{cases} \tag{6}$$

For instance, considering Table 4, if S_1 copies the pseudovalue c_{21} of Book2 (corresponding to the author Clive Cussler) from S_2, or vice-versa, then either they both provide the pseudovalue c_{21} or neither of the two provides it.

On the other hand, the probabilities that the two sources have the same claim about the pseudovalue c independently of each other, in the two cases that c is true or false, are defined as follows:

$$\begin{aligned} P(\psi_c | s_i \perp_c s_j,\ c) &= \tau_d^{pre}(s_i)\, \tau_d^{pre}(s_j) \\ &+ [1 - \tau_d^{npv}(s_i)]\, [1 - \tau_d^{npv}(s_j)] \end{aligned} \tag{7}$$

$$\begin{aligned} P(\psi_c | s_i \perp_c s_j,\ \bar{c}) &= \tau_d^{npv}(s_i)\, \tau_d^{npv}(s_j) \\ &+ [1 - \tau_d^{pre}(s_i)]\, [1 - \tau_d^{pre}(s_j)] \end{aligned} \tag{8}$$

Suppose that Clive Cussler is actually an author of Book2. Then, the probability that S_1 and S_2 have the same claim about c_{21} (Eq. 7) is the probability that they both correctly provide a pseudovalue (equal to $\tau_d^{pre}(S_1)\tau_d^{pre}(S_2)$) plus the probability that they are both wrong when not providing a pseudovalue (equal to $[1 - \tau_d^{npv}(S_1)][1 - \tau_d^{npv}(S_2)]$). The same reasoning can be applied to Eq. 8.

Bayes' theorem can now be applied to compute the probability of two sources being dependent or independent with respect to the pseudovalue c, given that they have the same claim about c. In the first case we can also understand which of the two sources is the copier.

Let $Y = \{s_i \xrightarrow{c} s_j, s_j \xrightarrow{c} s_i, s_i \perp_c s_j\}$. For each $y \in Y$, we can compute the following probability:

$$\begin{aligned} P(y | \psi_c) &= \frac{P(\psi_c | y) P(y)}{P(\psi_c)} = \frac{P(\psi_c | y) P(y)}{\sum_{y' \in Y} P(\psi_c | y') P(y')} \\ &= \frac{P(y)[P(\psi_c | y, c)\sigma(c) + P(\psi_c | y, \bar{c})(1 - \sigma(c))]}{\sum_{y' \in Y} P(y')[P(\psi_c | y', c)\ \sigma(c) + P(\psi_c | y', \bar{c})(1 - \sigma(c))]} \end{aligned} \tag{9}$$

For ease of notation, η_{ij}^d is used to denote the *prior probability* $P(s_i \xrightarrow{c} s_j)$ while η_{ji}^d denotes the prior probability $P(s_j \xrightarrow{c} s_i)$, in the domain of interest d of the object we are considering. Given the assumption of no mutual copying at domain level, it also holds that:

$$P(s_i \perp_c s_j) = 1 - \eta_{ij}^d - \eta_{ji}^d \tag{10}$$

Let us compute $P(s_i \xrightarrow{c} s_j|\psi_c)$ (the computation of $P(s_j \xrightarrow{c} s_i|\psi_c)$ is analogous) by substituting Eqs 6, 7, 8, 10 into Eq. 9:

$$P(s_i \xrightarrow{c} s_j|\psi_c) = \frac{\eta_{ij}^d}{\eta_{ij}^d + \eta_{ji}^d + \left(1 - \eta_{ij}^d - \eta_{ji}^d\right) P_u} \tag{11}$$

where

$$\begin{aligned} P_u =& \sigma(c)\left[\tau_d^{pre}(s_i) \cdot \tau_d^{pre}(s_j) + \left(1 - \tau_d^{npv}(s_i)\right) \cdot \left(1 - \tau_d^{npv}(s_j)\right)\right] \\ &+ (1 - \sigma(c))\left[\tau_d^{npv}(s_i) \cdot \tau_d^{npv}(s_j)\right. \\ &\left.+ \left(1 - \tau_d^{pre}(s_i)\right) \cdot \left(1 - \tau_d^{pre}(s_j)\right)\right] \end{aligned} \tag{12}$$

Now we need to find a way to estimate the prior probabilities η_{ij}^d and η_{ji}^d of the Bayesian model. We define them as the copying probabilities, $P(s_i \xrightarrow{\mathrm{d}} s_j|\Theta_{ij}^d)$ and $P(s_j \xrightarrow{\mathrm{d}} s_i|\Theta_{ij}^d)$, of the two sources in the domain d of the object we are considering; we will define these probabilities in Sect. 5.2.2.

Example 4 (Copying a pseudovalue) Let us compute, at a certain iteration *i*, the probability that source S_2 has copied from S_1 the common pseudovalue c_{13} (corresponding to the author Margaret Williams) of Book1 ($P(S_1 \xrightarrow{c_{13}} S_2|\psi_{c_{13}})$) in domain d ="literature." Considering that after iteration *i-1* the veracities hypothesized in Example 3 have been computed, and adding that for Book3 $\sigma(c_{31}) = 0.96$, the precisions and negative predictive values after iteration *i-1* are as follows: $\tau_d^{pre}(S_1) = 0.9083$, $\tau_d^{npv}(S_1) = 0.8$, $\tau_d^{pre}(S_2) = 0.774$, $\tau_d^{npv}(S_2) = 0.18$. Moreover, suppose that after iteration *i-1* we have computed: $P(S_1 \xrightarrow{\mathrm{d}} S_2|\Theta_{12}^d) = 0.25$, $P(S_2 \xrightarrow{\mathrm{d}} S_1|\Theta_{12}^d) = 0.7$. We use Eqs. 11 and 12, recalling that the prior probabilities η_{12} and η_{21} have been defined as $P(S_1 \xrightarrow{\mathrm{d}} S_2|\Theta_{12}^d)$ and $P(S_2 \xrightarrow{\mathrm{d}} S_1|\Theta_{12}^d)$:
$P_u = 0.9[(0.9083 \cdot 0.774) + (1 - 0.8)(1 - 0.18)] +$
$(1 - 0.9)[(0.8 \cdot 0.18) + (1 - 0.9083)(1 - 0.774)] = 0.7968$
$P(S_2 \xrightarrow{c_{13}} S_1|\psi_{c_{13}}) = \frac{0.7}{0.25+0.7+(1-0.7-0.25)\cdot 0.7968} = 0.7072$

5.2.2 Copying at the level of the domain of interest

So far we have computed the probability that a source s_i has copied a specific common pseudovalue c from source s_j given the observation that s_i and s_j have the same claim on c. Let $s_i \xrightarrow{o} s_j$ be the event according to which s_i copies from s_j a common pseudovalue related to object o, and ψ_o the observation that the two sources provide the same object o. The probability that s_i copies a common pseudovalue for object o from s_j is defined as the average of the probabilities related to all the common pseudovalues associated with o:

$$P(s_i \xrightarrow{o} s_j|\psi_o) = \frac{\sum_{c \in C_{s_i}(o) \cap C_{s_j}(o)} P(s_i \xrightarrow{c} s_j|\psi_c)}{|C_{s_i}(o) \cap C_{s_j}(o)|} \tag{13}$$

Equation 13 expresses the probability that a source s_i copies from another source s_j a common pseudovalue for object o. However, also the other possible non-common pseudovalues must be taken into consideration in order to appropriately compute the probability that the common ones were really copied. Indeed, the meaning of a high $P(s_i \xrightarrow{o} s_j|\psi_o)$ is different according to the fact that s_i and s_j share all the pseudovalues they provide for o, or that the common pseudovalues are just a small fraction.

Therefore, the copying probability at the level of the domain of interest can be finally defined as:

$$P(s_i \xrightarrow{\mathrm{d}} s_j|\Theta_{ij}^d) = \frac{\sum_{o \in \Theta_{ij}^d} P(s_i \xrightarrow{o} s_j|\psi_o) \cdot J_{ij}(o)}{|\Theta_{ij}^d|} \tag{14}$$

where $J_{ij}(o)$ is the Jaccard similarity of the two sets of pseudovalues of o provided by the two sources s_i and s_j:

$$J_{ij}(o) = J_{ji}(o) = \frac{\left|C_{s_i}(o) \cap C_{s_j}(o)\right|}{\left|C_{s_i}(o) \cup C_{s_j}(o)\right|} \tag{15}$$

Example 5 (Copying at the level of the domain of interest) We would like to compute the probability that S_2 copies from S_1 in the domain d = "literature." The two sources provide two common books Book1 and Book2, with three common pseudovalues for Book1 and one common pseudovalue for Book2. In Example 4 we have computed $P(S_2 \xrightarrow{c_{13}} S_1|\psi_{c_{13}}) = 0.7072$ for Book1. Similarly, we can perform the computation for the other common pseudovalues, obtaining for Book1 $P(S_2 \xrightarrow{c_{11}} S_1|\psi_{c_{11}}) = 0.7059$, $P(S_2 \xrightarrow{c_{12}} S_1|\psi_{c_{12}}) = 0.7084$, and for Book2 $P(S_2 \xrightarrow{c_{21}} S_1|\psi_{c_{21}}) = 0.7054$.

We apply Eq. 13 to derive the copying probabilities at the object level:
$P(S_2 \xrightarrow{\text{Book1}} S_1|\psi_{\text{Book1}}) = \frac{0.7072+0.7059+0.7084}{3} = 0.7072$
$P(S_2 \xrightarrow{\text{Book2}} S_1|\psi_{\text{Book2}}) = \frac{0.7054}{1} = 0.7054$
Using Eq. 15, Jaccard coefficients are as follows:
$J_{12}(\text{Book1}) = 1$
$J_{12}(\text{Book2}) = 1/3 = 0.3333$
Finally, the copying probability at domain level is:
$P(S_2 \xrightarrow{\mathrm{d}} S_1|\Theta_{12}^d) = \frac{0.7072 \cdot 1 + 0.7054 \cdot 0.3333}{2} = 0.4712$

5.2.3 Initialization

The copying probability at the level of the domain of interest is computed from the copying probabilities associated with the individual pseudovalues, and in the Bayesian model for the copying probabilities of the pseudovalues we have chosen to employ the probabilities at the level of the domain of interest $P(s_i \xrightarrow{\text{d}} s_j|\Theta^d_{ij})$ as the priors η^d_{ij}. At iteration k, the probabilities at the level of the domain of interest computed at iteration $k-1$ are used to derive the probabilities associated with the pseudovalues, but for the first iteration an effective initialization is needed.

To this aim, we exploit the concept of *domain expertise* of a source, introduced in [26]. The expertise of source s in domain of interest d, denoted by $e_d(s)$, is defined on the basis of the percentage of objects belonging to d that are provided by s. Also, the expertise score is adjusted by means of a term taking into account the fact that objects may belong to multiple domains, because some domains of interest may be overlapping (e.g., a book may be both a biography and a history book). For the sake of conciseness we do not delve into the details of the formulas to compute domain expertise, and refer the interested readers to [26]. We just notice that in [26] the expertise computation relies on two parameters to be chosen: ρ, which should be set higher when the domains are very overlapped (i.e., they have a high percentage of common values), and α, which is an adjust factor accommodating possibly uneven distributions of objects among the sources. In our experiments, these parameters will be set according to the guidelines provided in [26].

Our initialization relies on the assumption that sources with high expertise in domain d are less likely to be copiers for domain d and that sources with low expertise in d tend to copy from sources with higher expertise in d. These ideas are summarized in this formula:

$$\eta^d_{ij} = [1 - e_d(s_i)]\, e_d(s_j) \qquad \forall s_i,\ s_j \in \mathscr{S} \wedge s_i \neq s_j \quad (16)$$

5.3 Source authority

The key idea to define the authority of a source, in a specific domain of interest, based on the detection of which sources copy from which ones, is that if many sources copy some values from the same source s_a, it is because s_a is considered authoritative and more trustworthy.

The *unadjusted authority score* of source s in domain d measures *how much of source s is copied in d with respect to how much all sources are copied in d* (Eq. 17):

$$a_d(s_j) = \frac{\sum_{s_i \in \mathscr{S}} P(s_i \xrightarrow{\text{d}} s_j|\Theta^d_{ij})}{\sum_{s_k \in \mathscr{S}} \sum_{s_l \in \mathscr{S}} P(s_l \xrightarrow{\text{d}} s_k|\Theta^d_{kl})} \quad (17)$$

Note that $a_d(s)$ is actually an absolute measure, while we would like it to be a value between 0 and 1 as for all the other scores of this study. We can then apply a linear conversion to $a_d(s)$ in order to map it on the interval [0, 1]. We denote this new score as $A_d(s)$ or *authority of source s* in domain of interest d, computed as:

$$A_d(s) = \frac{a_d(s) - a^{min}_d}{a^{max}_d - a^{min}_d} \quad (18)$$

where a^{max}_d and a^{min}_d represent, respectively, the maximum and minimum unadjusted authority scores observed in domain d.

Example 6 (Source Authority) Let us compute the authorities of sources S_1 and S_2 in the domain of interest d = "literature." In Example 5 we have computed $P(S_2 \xrightarrow{\text{d}} S_1|\Theta^d_{12}) = 0.4712$. Exploiting the information provided in Examples 4 and 5, it is easy to derive $P(S_1 \xrightarrow{\text{d}} S_2|\Theta^d_{12}) = 0.1683$. Moreover, let us assume that $P(S_1 \xrightarrow{\text{d}} S_3|\Theta^d_{12}) = 0.1$, $P(S_3 \xrightarrow{\text{d}} S_1|\ \Theta^d_{12}) = 0.6$, $P(S_2 \xrightarrow{\text{d}} S_3|\Theta^d_{12}) = 0.2$, and $P(S_3 \xrightarrow{\text{d}} S_2|\Theta^d_{12}) = 0.3$.
Applying Eqs. 17 and 18:
$a_d(S_1) = \frac{0.4712+0.6}{0.4712+0.1683+0.1+0.6+0.2+0.3} = 0.5823$
$a_d(S_2) = \frac{0.1683+0.3}{0.4712+0.1683+0.1+0.6+0.2+0.3} = 0.2546$
$a_d(S_3) = \frac{0.1+0.2}{0.4712+0.1683+0.1+0.6+0.2+0.3} = 0.1631$
$A_d(S_1) = \frac{0.5823-0.1631}{0.5823-0.1631} = 1$
$A_d(S_2) = \frac{0.2546-0.1631}{0.5823-0.1631} = 0.2183$
$A_d(S_3) = \frac{0.1627-0.1627}{0.5827-0.1627} = 0$

Note that the copying probabilities at the level of the domain of interest indicate that S_1 is likely to be copied more often than S_2 and S_3 for literature books, and this results in a higher authority in that domain.

5.4 Veracity computation

In Sect. 5.1 we have explained that the veracity of a pseudovalue c is evaluated through a Bayesian inference iterative algorithm. At each iteration, the veracity estimations are updated with Eq. 3 on the basis of the veracities computed at the previous iteration and of the probabilities of the observations $P(\psi(o)|c)$ and $P(\psi(o)|\bar{c})$. In this work we compute these probabilities extending the formulas proposed in [26], by *leveraging the authority score of each source*.

The model of [26] relies on the expertise of the sources (already explained in Sect. 5.2.3) and on the confidence of the sources in the pseudovalues (i.e., in the clusters containing the values they expose). In our multi-truth scenario, the confidence of source s in pseudovalue c of object o, denoted by $c_s(c)$, reflects how much s is convinced that c is part of the truth of object o (if s provides c) or not part of the truth of

object o (if s does not provide c). Also notice that in a multi-truth scenario each source s might provide a *partial* truth, therefore we should not set $c_s(c) = 0$ for all pseudovalues c not provided by s (and possibly provided by some other source). The evaluation of $c_s(c)$ can be expressed as follows:

$$c_s(c) = \begin{cases} \left(1 - \frac{|C(o) \setminus C_s(o)|}{|C(o)|^h}\right) \frac{1}{|C_s(o)|} & \text{if } c \in C_s(o) \\ \frac{1}{|C(o)|^h} & \text{if } c \notin C_s(o) \end{cases} \tag{19}$$

In [26], h is not a parameter, and is set to 2. Here we adopt a more flexible solution in order to comply with the needs of different data domains. The higher the value of h, the more confident the source is in the pseudovalues it provides. High confidence in positive claims with respect to negative ones may be appropriate when the set of alternative pseudovalues provided by the sources is large, and therefore there might be many negative claims. On the contrary, when negative claims are less numerous, then a more significant confidence should be associated with them. For instance, consider $C(o) = \{c_1, c_2, c_3, c_4\}$ and $C_s(o) = \{c_1, c_2\}$. For source s, if $h = 1$ all the four pseudovalues have confidence $\frac{1}{4}$, while if $h = 1.5$ the positive claims c_1 and c_2 have confidence $\frac{3}{8}$ and the negative claims c_3 and c_4 have confidence $\frac{1}{8}$.

Our key idea in computing the probability of the observations is to reward the sources positively according to their authority. Let $S_o(c)$ be the set of sources providing pseudovalue c for object o, and $S_o(\bar{c})$ the set of sources providing the object o but not indicating c among its pseudovalues. The probability of the observations is as follows:

$$P(\psi(o)|c) = \prod_{s \in S_o(c)} \tau_d^{pre}(s)^{\beta^d_{s,c}} \times \prod_{s \in S_o(\bar{c})} \left(1 - \tau_d^{npv}(s)\right)^{\beta^d_{s,c}} \tag{20}$$

$$P(\psi(o)|\bar{c}) = \prod_{s \in S_o(\bar{c})} \tau_d^{npv}(s)^{\beta^d_{s,c}} \times \prod_{s \in S_o(c)} \left(1 - \tau_d^{pre}(s)\right)^{\beta^d_{s,c}} \tag{21}$$

where

$$\beta^d_{s,c} = \min\left(c_s(c) \cdot (e_d(s) + A_d(s)^k),\ 1\right)$$

In Eq. 20, the probability of observing $\psi(o)$ knowing that c is true is the probability that the sources providing c are right in specifying a certain pseudovalue (the product involving $\tau_d^{pre}(s)$) and the sources not providing c make a mistake not specifying it (the product involving $(1 - \tau_d^{pre}(s))$). Precision and negative predictive value are adjusted on the basis of confidence, expertise and authority. Analogous reasoning holds for Eq. 21.

Note that an object o can belong to multiple domains of interest. If this is the case, the quantities $\tau_d^{pre}(s)$, $\tau_d^{npv}(s)$, $e_d(s)$, $A_d(s)$ in Eqs. 20-21 are those referred to the domain d for which s has the greatest expertise $e_d(s)$; indeed, if the source provides more data in a certain domain of interest the scores related to that domain of interest are expected to be more reliable. Parameter k modulates the contribution of the authority to the exponent $\beta^d_{s,c}$. Since $0 \leq A_d(s) \leq 1$, the lower is k, the greater is the contribution of $A_d(s)$. When the authority can represent a clear and reliable distinction between the sources we can use a low value for k, otherwise more caution is advised.

Example 7 (Veracity computation) Let us compute the veracity of the pseudovalue c_{13} of Book1, corresponding to Margaret Williams, in the domain of interest d="literature," at iteration i. Note that Book1 is provided by all the three sources, but only S_1 and S_2 specify c_{13} among its pseudovalues. Suppose that precisions, negative predictive values and veracities after iteration *i-1* are those indicated in Examples 3 and 4. The expertise of the sources, computed as described in [26], is as follows: $e_d(S_1) = 0.821$, $e_d(S_2) = e_d(S_3) = 0.7$. Eq. 19 can be used to derive the confidence of the sources in their claims about Margaret Williams. Setting $h = 2$: $c_{S_1}(c_{13}) = c_{S_2}(c_{13}) = \left(1 - \frac{3-3}{3^2}\right) \cdot \frac{1}{3} = \frac{1}{3}$ and $c_{S_3}(c_{13}) = \frac{1}{3^2} = \frac{1}{9}$ Eqs. 20-21 allow to determine the probability of the observations. Setting $k = 1$:

$\beta^d_{S_1,c_{13}} = \min((1/3) \cdot (0.821 + 1),\ 1) = 0.6070$;
$\beta^d_{S_2,c_{13}} = \min((1/3) \cdot (0.7 + 0.2183),\ 1) = 0.3061$;
$\beta^d_{S_3,c_{13}} = \min((1/9) \cdot (0.7 + 0),\ 1) = 0.0778$;
$P(\psi(Book1)|c_{13}) = 0.9083^{0.6070} \cdot 0.774^{0.3061} \cdot (1 - 0.36)^{0.0778} = 0.8424$;
$P(\psi(Book1)|\bar{c}_{13}) = (1-0.9083)^{0.6070} \cdot (1-0.774)^{0.3061} \cdot 0.36^{0.0778} = 0.1374$;

Finally, the value of the veracity of c_{13} at iteration i is computed through Eq. 3, presented at the beginning of Sect. 5: $P(c_{13}|\psi(Book1)) = \frac{1}{1+\frac{1-0.9}{0.9} \cdot \frac{0.1374}{0.8424}} = 0.9822$ Note that the veracity of c_{13} at the previous iteration was 0.9: the fact that Margaret Williams is present in authoritative sources makes the veracity increase through the iterations.

5.5 Selection of the true values

The last step we have described ends the iterative procedure, providing a veracity score for each pseudovalue for each object. Now, for each object, we have to choose the set of true *values*. This happens in two steps: First, the true pseudovalues are selected, and then a representative value is extracted from each of them.

The true pseudovalues of an object o are selected by applying a threshold to the veracity. Given that the veracities are probabilities, it is reasonable to set the threshold to 0.5 rather

than considering it as a parameter to be tuned. Indeed, the fact that a value has veracity greater than 0.5 indicates that the value is more likely to be true than false:

$$true\_pseudovalues(o) = \{c \in C(o) : \sigma(c) > 0.5\} \quad (22)$$

In order to associate a representative value with each pseudovalue $c \in C(o)$, we apply as first criterion the number of sources providing each value, because this reflects the way in which the pseudovalues are built. Therefore, we begin with computing the set $most\_freq\_vals(c)$ of values provided by the highest number of sources:

$$most\_freq\_vals(c) = \underset{v \in c}{\text{argmax}} |S_o(v)| \quad (23)$$

However, it may well happen that the cardinality of *most_freq_vals(c)* is greater than 1, especially when the total number of sources is not very high. In this case, the value provided by the most authoritative sources is selected:

$$represent\_val(c) = \underset{v \in most\_freq\_vals(c)}{\text{argmax}} \sum_{s \in S_o(v)} A_d(s) \quad (24)$$

In case o belongs to multiple domains of interest, the domain of interest d to be used to compute $A_d(s)$ is that for which $A_d(s)$ is the greatest. Please notice that when computing authority we adopt high arithmetic precision, thus it is unlikely that $|represent\_val(c)| > 1$.

Finally, the true values associated with object o are the following:

$$truth(o) = \{represent\_val(c) : c \in true\_pseudovalues(o)\} \quad (25)$$

Example 8 (Selection of the True Values) Let us consider Book1. In Example 7 we have computed the veracity of the pseudovalue c_{13}, representing the author Margaret Williams, after iteration i, i.e., $\sigma(c_{13}) = 0.9822$. We can easily derive also $\sigma(c_{11}) = 0.9959$ and $\sigma(c_{12}) = 0.9863$. Suppose that the procedure has reached convergence, and that we apply a very common veracity threshold equal to 0.5. We find that all the pseudovalues exceed the threshold: *true_pseudovalues*(Book1) = $\{c_{11}, c_{12}, c_{13}\}$. Let us now select the true *values* by computing the most frequent ones: *most_freq_vals*(c_{11}) = {"Jean Cooney"}, *most_freq_vals*(c_{12}) = {"John Golder"}, *most_freq_vals*(c_{13}) = {"Margaret Williams"}. All the sets include just one element, therefore we do not need to use the source authorities to solve: truth(Book1)= {"Jean Cooney," "John Golder," "Margaret Williams"}.

Let us now consider Book2. The veracities can be easily computed as follows: $\sigma(c_{21}) = 0.9992$, $\sigma(c_{22}) = 0.9727$, $\sigma(c_{23}) = 0.2461$. Applying the threshold, we find *true_pseudovalues*(Book2) = $\{c_{21}, c_{22}\}$. Both the selected pseudovalues contain a single value, so the result is truth(Book2)= {"Clive Cussler", "Graham Brown"}.

5.6 The authority-based Bayesian inference algorithm

Algorithm 3 formally describes STORM's Bayesian inference procedure, schematized in Fig. 2 and detailed in the previous paragraphs.

The algorithm takes as input the sources, objects and domains of interest, the correspondence between objects and domains of interest, and the pseudovalues as computed in Sect. 4. Also, some parameters need to be specified: the initial default values for the source quality measures, the value for h in Eq. 19, the value for k in Eqs. 20-21, and finally the parameters ρ and α to compute the expertise as defined in paper [26]. The output is a set of true values for each object.

At Lines 1-19 the algorithm performs the necessary initializations. In particular, the *for* loop at Lines 1-9 initializes the source quality measures with their default values and computes expertise and confidences, while the *for* at Lines 10-12 sets the default values for the veracities. At Lines 13-19 the copying probabilities are initialized using Eq. 16; also, the sets of common objects between pairs of sources are computed, as well as the Jaccard distances between the sets of common pseudovalues.

Algorithm 3 then starts the iterative loop that proceeds until convergence (Lines 20-41). In more detail, Lines 21-30 manage the copy detection step described in Sect. 5.2, while Lines 31-33 compute the authorities as explained in Sect. 5.3. The formulas for veracity computation reported in Sects. 5.1 and 5.4 are applied at Lines 34-37. Finally, precision and negative predictive value are updated by the loop at Lines 38-40. Note that the iterations terminate when the algorithm converges; for us this means that no pseudovalue veracity has changed more than a certain threshold with respect to the previous iteration, or that the set of pseudovalues exceeding the veracity threshold has remained unchanged for a certain number of consecutive iterations.

The algorithm terminates with the choice of the true values for each object, using Eq. 25 and relying on the veracity threshold to discriminate between the pseudovalues (Lines 42-44).

Let us now analyze the complexity of Algorithm 3. Suppose there are N objects and M sources, and that on average an object belongs to d domains of interest, a source provides c pseudovalues for an object, and a pseudovalues are globally associated with an object. Regarding the initialization, the computation of the expertise, as declared in [26], can be performed in $O(d^2M + 3dM)$. Moreover, determining the confidence for each source and pseudovalue requires

Algorithm 3 STORM's authority-based Bayesian inference

Input: sources $\mathscr{S}$, objects $\mathscr{O}$, domains of interest $\mathscr{D}$ and the mapping between objects and domains of interest, pseudovalues provided by sources for objects $C_s(o)$, default values for $\tau_d^{pre}(s)$ and $\tau_d^{npv}(s)$, parameter h (Eq. 19), parameter k (Eqs. 20-21), parameters ρ and α for the expertise
Output: $\forall o \in \mathscr{O}$, set *truth(o)* of true values

// Initializations
for all $s \in \mathscr{S}$ **do**
  for all $d \in \mathscr{D}$ **do**
    Compute $e_d(s)$ as described in [26]
    $\tau_d^{pre}(s), \tau_d^{npv}(s) \leftarrow$ default values
  end for
  for all $o \in \mathscr{O}, c \in C(o)$ **do**
    $c_s(c) \leftarrow$ Eq. 19
  end for
end for
for all $o \in \mathscr{O}, c \in C(o)$ **do**
  $\sigma(c) \leftarrow$ default value (0.5)
end for
for all $d \in \mathscr{D}, s_i, s_j \in \mathscr{S}, s_i \neq s_j$ **do**
  Initialize $\eta_{ij}^d \leftarrow$ Eq. 16
  Compute the set of common objects Θ_{ij}^d
  for all $o \in \Theta_{ij}^d$ **do**
    $J_{ij}(o) \leftarrow$ Eq. 15
  end for
end for
// Iterate until convergence
repeat
  // Copy detection
  for all $d \in \mathscr{D}, s_i, s_j \in \mathscr{S}$ **do**
    for all $o \in \Theta_{ij}^d$ **do**
      for all $c \in C_{s_i}(o) \cap C_{s_j}(o)$ **do**
        $P(s_i \xrightarrow{c} s_j|\psi_c) \leftarrow$ Eq. 11
      end for
      $P(s_i \xrightarrow{o} s_j|\psi_o) \leftarrow$ Eq. 13
    end for
    $P(s_i \xrightarrow{d} s_j|\Theta_{ij}^d) \leftarrow$ Eq. 14
    $\eta_{ij}^d \leftarrow P(s_i \xrightarrow{d} s_j|\Theta_{ij}^d)$
  end for
  // Source authority computation
  for all $s \in \mathscr{S}$ **do**
    $A_d(s) \leftarrow$ Eqs. 17-18
  end for
  // Veracity computation
  for all $o \in \mathscr{O}, c \in C(o)$ **do**
    $P(\psi(o)|c), P(\psi(o)|\bar{c}) \leftarrow$ Eqs. 20 and 21
    $\sigma(c) \leftarrow$ Eq. 3
  end for
  // Update source trustworthiness measures
  for all $s \in \mathscr{S}, d \in \mathscr{D}$ **do**
    $\tau_d^{pre}(s), \tau_d^{npv}(s) \leftarrow$ Eqs. 4 and 5
  end for
until *convergence*
// True values selection
for all $o \in \mathscr{O}$ **do**
  truth(o) $\leftarrow$ Eq. 25
end for

$O(aMN)$. Assigning the default values to the veracity is $O(aN)$, while doing the same for the quality measures is $O(dM)$. The loop realizing the copy-related initializations can be executed in $O(dM^2N)$. Let us now consider the iteration part, assuming there are I iterations. The copy detection requires $O(dcM^2N)$, the authority computation $O(M^2)$, and the veracity computation $O(cMN)$. Updating the source quality measures needs to consider in each domain all the pseudovalues provided by the sources, so it requires $O(dcMN)$. Finally, after the iterations, choosing the true values for the objects is $O(aN)$. To summarize, considering d, c and a as negligible, the complexity of the three parts of the algorithm can be expressed as $O(M^2N + IM^2N + N)$, i.e., $O(IM^2N)$. The algorithm is linear in the number of objects but, as in [38], leveraging copy detection comes at the cost of making the methodology quadratic in the number of sources.

6 Experiments

STORM was implemented in Python and evaluated through an extensive set of experiments. Specifically, Sect. 6.1 describes the datasets we employed, while Sect. 6.2 presents the results of a comparison in terms of effectiveness with competitor algorithms from the recent literature. Then, Sect. 6.3 analyzes parameter sensitivity, Sect. 6.4 studies STORM's scalability, Sect. 6.5 examines the computed authority scores for one of the datasets, Sect. 6.6 proposes a detailed appraisal of our similarity measure and, finally, Sect. 6.7 draws the conclusions of the evaluation.

6.1 Datasets

We carried out our experiments using one synthetic dataset and three real-world datasets belonging to different scenarios: books, movies, and daily flights.

Books The Books dataset, kindly provided by the authors of [26], contains information about books supplied by online bookstores (i.e., data sources). Each store specifies the authors of a subset of the books, and the aim is to discover the correct set of authors for each book. Every category of books is a domain of interest. Coherently with the literature on multi-truth data fusion [11,26,38,39,51], we excluded the books for which all the sources agree on the same set of authors; actually, no data fusion is needed for these objects. We also applied to the author lists the basic cleaning operations described in Sect. 4.1. The final preprocessed dataset contains 58,093 books and 6,461 sources. On average, a book is provided by 24.04 sources, a source specifies 1.28 authors for a book, and a book is associated with 3.87 distinct authors provided by at least one source. Moreover, there are 18 book categories (i.e., domains of interest); on average each category is associated with 3,544.7 books, and 8.92% of the books belong to multiple categories. To assess the effectiveness of the algorithms, we manually built a golden truth by randomly choosing 872 books and looking for their real authors on the

original book cover. We included in the golden truth only books for which all the authors specified on the cover are provided by at least one source.

Movies The Movies dataset, also provided by the authors of [26], contains information about movies supplied by some websites (i.e., data sources). Each website indicates the directors of a subset of the movies, and the aim is to discover the correct set of directors for each movie. In this case, the domain of interest is represented by the genre of the movie. As in the case of the Books dataset, the movies for which all the websites specify the same set of directors are excluded. Again, we applied to the director lists the basic cleaning operations defined in Sect. 4.1. The final, preprocessed dataset contains 13,437 movies and 15 sources. On average, a movie is provided by 3.26 sources, a source specifies 1.46 directors for a movie, and a movie is associated with 2.31 distinct directors provided by at least one source. Moreover, there are 21 movie genres (i.e., domains of interest); on average each genre is associated with 1,497.7 movies, and 70.20% of movies belong to multiple genres. We manually built a golden truth, by inspecting the movie posters of 400 randomly chosen movies. Again, only movies for which all the directors specified in the poster are provided by at least one source were considered.

DailyFlights The DailyFlights dataset is a multi-truth dataset we constructed starting from the Flight dataset used in [22][5], which contains information – like scheduled and actual departure and arrival times – about flights during December 2011. In our version of the dataset, the objects are the triples ⟨airline, route, date⟩, e.g., ⟨*American Airlines, JFK-LAX, 2011-12-01*⟩. The values are all the actual arrival times of the flights operated by the airline on the specified route in the given date. The sources are websites providing information about flights, and we use the airline as domain of interest. The dataset includes 21,206 objects and 38 sources. On average, an object is provided by 22.19 sources, a source specifies 1.52 values for an object, and an object is associated with 6.64 distinct values provided by at least one source. Moreover, there are three airlines (i.e., domains of interest), and on average an airline is associated with 7,068.7 objects; an object by construction is associated with exactly one airline. We built a golden truth by exploiting the database of the US Bureau of Transportation Statistics (BTS)[6]. We included the triples from the BTS database present also in our dataset, and only the actual arrival times of the flights whose scheduled departure time is specified by at least one source in the dataset. The resulting golden truth contains 7,778 objects. In this case we did not remove from the golden truth the objects having at least one true value not specified by any source, because this would have excluded a large portion of the objects associated with multiple true values.

Synthetic dataset We built a synthetic dataset adapting the *Pareto universe* paradigm, defined in [7], to the multi-truth scenario. This dataset contains 1,000 objects of the same domain and 20 sources. The 20% of the sources are independent, with error rates set to 0.1, 0.2, 0.6, and 0.9, while the other 80% are copiers. Among the copiers, 50% provide 20% of the objects independently, with error rate 0.8; 25% of the copiers provide random values for 20% of the objects; finally, the remaining 25% of the copiers copy all the objects. We assumed that sources copy values from one of the two more accurate sources. Each source provides values for every objects, and a source can claim three values for each object, among six possible values, half of which are correct.

6.2 Effectiveness evaluation

6.2.1 Baselines and metrics

We compared STORM with several data-fusion techniques among those presented in Sect. 2. First, we considered multi-truth algorithms: SmartVote [11], DART [26], and LTM [51]. Note that among the algorithms by the research group of Wang et al. [11,38,39], we considered SmartVote, which is the most recent and best-performing one. We added to the set of competitors Majority Voting, a basic method that selects a value as true if the proportion of sources providing that value is the biggest one. We also compared STORM against a set of traditional single-truth methods: TruthFinder [44], HubAuthority [17], Investment [31], and PooledInvestment [31]. We adapted the single-truth algorithms to the multi-truth scenario by making them return as true all the values with scores greater than a certain threshold, whose value is optimized in order to achieve the best performance. When needed, we also normalized the scores generated by the algorithms, in order to be consistent in the application of the thresholds. Finally, we performed an ablation study testing STORMNoRec – a variant of STORM that excludes the value reconciliation phase – in order to assess the contribution of this part of the methodology.

The competitor algorithms were implemented in Python as well, except LTM for which we used a publicly available Java version[7]. The parameters for the competitor algorithms were chosen according to the optimal settings recommended by their authors, and applying minor modifications when these led to better performance.

For our methods, we need to set the values of four parameters: the initial precision (τ_0^{pre}), the initial negative predictive value (τ_0^{npv}), the exponent h in the denominator of the confidence in Eq. 19, and the exponent k for the authority in

[5] Available at http://lunadong.com/fusionDataSets.htm.

[6] https://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236.

[7] https://github.com/daqcri/DAFNA-EA.

Eqs. 20, 21. In addition, there are two further parameters regarding the source expertise (mentioned in Sect. 5.2.3). Please notice that STORM is an unsupervised algorithm, so we cannot fine-tune the parameters through a validation set. We set the parameters by intuition, based on their meaning with respect to the input data. Moreover, we initially performed trial runs of the algorithms and inspected the results: if the computed truth was clearly not appropriate, e.g., because *true* (or *false*) was assigned to almost all values, we experimented alternative parameter configurations.

Table 5 shows the parameter settings we employed for the different datasets and algorithms, as explained below:

- h: the Movies dataset contains few values per object and few negative claims, therefore for this dataset we chose $h < 1$ in Eq. 19 in order to give high weight to the few negative claims and preserve their information; instead, for Books, DailyFlights and Synthetic datasets we selected $h > 1$.
- τ_0^{pre}, τ_0^{npv}: In the Books case, where many sources are available and many values are associated with lots of negative claims, we further supported the positive claims by choosing $\tau_0^{pre} > \tau_0^{npv}$, while for the other datasets we set $\tau_0^{npv} > \tau_0^{pre}$. Note also that with the Movies dataset we imposed a larger gap between τ_0^{npv} and τ_0^{pre} for STORM than we did for STORMNoRec, because the value reconciliation further reduces the number of negative claims.
- k: it is the exponent for the authority in Eqs. 20 and 21, so we chose $k < 1$ in order to assign an important role to the authority with respect to the expertise in these equations. In particular, the values employed for Movies are slightly higher than the others not to disperse the expertise information excessively; indeed, in this case we deem the expertise to have an important role since the number of sources is comparatively very low. We used a lower value for the Synthetic dataset because, since each object is present in every source, we expect the authority to have a particularly higher contribution, compared to the expertise of the sources.
- ρ, α: they are the parameters used for the expertise computation, and were set according to the guidelines provided in [26]. In particular, exactly as in [26], ρ was set to 0.2 for Books and to 0.3 for Movies, because the latter contains greater domain overlapping; this parameter is not relevant for the DailyFlights and Synthetic datasets because their domains do not overlap. Moreover, paper [26] sets $\alpha = 1.5$ for Books because objects are unevenly distributed between sources, while retains $\alpha = 1$ for Movies; we used the same values for these datasets, and chose $\alpha = 1.5$ also for DailyFlights because its objects distribution is less balanced than in Movies. Since in the Synthetic dataset each object is present in all sources, we chose $\alpha = 0.5$.

In Sect. 6.3 we perform an analysis of the impact of parameter settings on STORM's performance.

Coherently with recent data-fusion literature [26,33,51], we evaluated the algorithms in terms of precision, recall and F1 score. In particular, we consider the F1 score as very relevant, because it identifies the algorithms showing the best compromise between precision and recall.

Validation of the Differences in Effectiveness To validate the significance of the differences in terms of F1 measure between STORM and the other algorithms, we used a statistical test, checking that the performance differences between two approaches are significant and not just due to chance. The aim is to reject the null hypothesis stating that *the compared methods have the same performance*.

We compared STORM with its competitors using McNemar's statistical test, which is particularly appropriate in this situation since, despite our golden truths having good sizes with respect to the data-fusion scenario, they are too small to be split into multiple parts as required by many statistical techniques [6].

McNemar's test defines a test statistic on the basis of the numbers of test items that are classified correctly or not correctly by the two compared methods. In this case the test items to be checked are all the values provided by at least one source for the objects contained in the golden truth: The algorithms have to judge whether these values are true or false. The greater the value of the McNemar test statistic, the lower the p-value of the test and therefore more evidence is provided to reject the null hypothesis of no difference. In practice, the null hypothesis is usually rejected with p-values lower than 0.05 or 0.01 [15].

6.2.2 Results

Table 6 shows the effectiveness results for different data-fusion techniques. In the F1 column, double asterisks (**) indicate that the performance difference with respect to STORM is statistically significant with p-value lower than 0.01, according to the McNemar's test. Note that the values in the DailyFlights and Synthetic datasets are not textual strings, so the value reconciliation step of the methodology is not applicable; therefore, STORMNoRec in this case coincides with STORM.

Table 7 highlights the importance of the STORM's value reconciliation phase, showing the percentage of objects with conflicts – i.e., for which not all the sources provide the same set of values/pseudovalues – before and after value reconciliation. Note that our datasets contain only objects with conflicting values, therefore, before the application of the value reconciliation, 100% of the objects have conflicts.

Table 5 Parameter settings for the experiments

Dataset	Algorithm	τ_0^{pre}	τ_0^{npv}	h	k	ρ	α
Books	STORM	0.9	0.8	1.2	0.1	0.2	1.5
	STORMNoRec	0.9	0.8	1.2	0.1	0.2	1.5
Movies	STORM	0.7	0.9	0.85	0.25	0.3	1
	STORMNoRec	0.9	0.95	0.85	0.5	0.3	1
DailyFlights	STORM	0.9	0.95	1.2	0.1	/	1.5
Synthetic	STORM	0.75	0.95	1.2	0.01	/	0.5

Table 6 Effectiveness results per dataset and method. In the F1 column, double asterisks (**) indicate that the performance difference with respect to STORM is statistically significant with p-value lower than 0.01, according to the McNemar's test

Method	Books dataset			Movies dataset			DailyFlights dataset			Synthetic dataset		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
STORM	0.8610	0.8872	**0.8739**	0.8382	0.8750	**0.8562**	0.8581	0.7401	**0.7948**	0.7797	0.9767	**0.8671**
STORMNoRec	0.8400	0.8836	0.8613**	0.8000	0.9054	**0.8494**	/	/	/	/	/	/
SmartVote [11]	0.8890	0.8675	**0.8781**	0.8012	0.9189	**0.8560**	0.6856	0.7371	0.7104**	0.7546	0.9673	0.8478**
DART [26]	0.8377	0.8684	0.8527**	0.7292	0.9629	0.8299**	0.7665	0.7836	0.7749**	0.7404	0.9793	0.8433**
LTM [51]	0.7167	0.9624	0.8216**	0.7544	0.9392	0.8367**	0.4993	0.8006	0.6151**	0.6884	0.9793	0.8085**
TruthFinder [44]	0.7214	0.8517	0.7811**	0.6693	0.9848	0.7970**	0.8065	0.7371	0.7702**	0.5273	0.9863	0.6872**
HubAuthority [17]	0.8002	0.8102	0.8052**	0.7090	0.9781	0.8221**	0.8389	0.6581	0.7376**	0.7893	0.9127	0.8465**
Investment [31]	0.8506	0.7189	0.7792**	0.7179	0.9512	0.8182**	0.7858	0.7813	0.7835**	0.7786	0.9200	0.8434**
PooledInvestment [31]	0.8247	0.7826	0.8031**	0.6854	0.9646	0.8014**	0.7976	0.7765	0.7869**	0.7905	0.9133	0.8475**
MajorityVoting	0.9059	0.7323	0.8099**	0.7965	0.6216	0.6983**	0.7277	0.4593	0.5632**	0.9411	0.4260	0.5865**

Table 7 Percentage of objects with conflicting values/pseudovalues, before and after STORM's value reconciliation phase

Dataset	Before value reconciliation	After value reconciliation
Books	100%	68.0%
Movies	100%	74.7%

6.2.3 Result analysis

Comparison: STORM vs. STORMNoRec Comparing STORM with STORMNoRec highlights the importance of the value reconciliation phase. A first preliminary evidence of this is given by the reduction of the percentage of objects with conflicting values/pseudovalues (Table 7): Only 68% of the books and 74.7% of the movies still have conflicts after the value reconciliation. This clearly eases the task of the subsequent Bayesian inference step.

Concerning the F1 scores (Table 6), STORM outperforms STORMNoRec on the Books dataset, achieving a remarkable +0.0126, again demonstrating the usefulness of the value reconciliation step. On the contrary, on the Movies dataset the gap between the two algorithms is not statistically significant. Indeed in Books, because of the great number of sources, there are many (possibly wrong) values per object, and the value reconciliation phase seems to handle well this situation, as testified by a much better performance of STORM over STORMNoRec. In Movies the number of values per object is lower, leading to a less noticeable gain.

Comparison with Literature Competitors The first observation that emerges from Table 6 is that STORM is the algorithm that globally performs better in terms of F1 score across the four datasets, thus exhibiting the best precision-recall compromise. Indeed, on the Books and Movies datasets STORM outperforms all the competitors except SmartVote, which obtains comparable results with no statistically significant difference. Moreover, on DailyFlights and Synthetic datasets even SmartVote is significantly outperformed.

SmartVote is the competitor that overall provides the best results. In particular, it performs very well on Books and Movies but fails to obtain a satisfying effectiveness on DailyFlights and Synthetic datasets, where its F1 measure is lower with respect to STORM. When running the algorithm we noticed that SmartVote is prone to bring the source quality measures employed in the computation very close to their maximum values, making the algorithm behave quite similarly to Majority Voting, though with substantial enrichment. It seems that, when the performance of Majority Voting is extremely poor – as on DailyFlights, SmartVote also shows limited effectiveness. STORM, which considers the source authorities in a Bayesian framework, does not seem to suffer from this problem and it adapts easily to all the scenarios.

Let us now consider the comparison with DART. This comparison is especially interesting because also DART relies on a Bayesian framework, the basic method to which

we added copy detection, source authority and value reconciliation. STORM attains a relevant performance gain in terms of F1 on all the experimented datasets: +0.0212 on Books, +0.0232 on Movies, +0.0199 on DailyFlights, and +0.0238 on Synthetic.

Notice that it seems that in STORM the two phases of value reconciliation and authority-based Bayesian inference somehow compensate each other. Indeed, consider that on DailyFlights and Synthetic STORM is composed only of the authority-based Bayesian inference, and that for the other datasets we can distinguish the contribution of the two phases of the algorithm by looking at the results of STORMNoRec. Therefore, apparently the use of authority is more effective when the number of sources is limited, i.e., with Movies, DailyFlights, and Synthetic. The reason is that with fewer sources the algorithm is able to better discriminate between those that are authoritative and those that are not. On the contrary, on the Books dataset, containing a large number of sources, the value-reconciliation component of STORM appears to play the most important role.

LTM, the oldest multi-truth algorithm, shows good performance on Books, Movies and Synthetic; in particular, on Movies it also outperforms DART. However, STORM exhibits a remarkably better F1 in those cases: +0.0523 on Books, +0.0164 on Movies and +0.0586 on Synthetic. In addition, LTM does not seem to properly manage the high number of values available in DailyFlights, achieving one of the worst F1 scores measured on this dataset.

The single-truth methodologies we tested are outperformed by STORM as well. Note that, to adapt these techniques to the multi-truth scenario, we performed an unrealistic fine-tuning of the truth threshold, which would not be feasible in practice in an unsupervised scenario. In spite of this, their performance is comparable with that of STORM only on DailyFlights, where the smallest recorded loss in terms of F1 is 0.0079 by PooledInvestment.

6.3 Parameter sensitivity

In this section we investigate the impact of different parameter settings on the effectiveness of STORM. This analysis is conducted on the Movies, DailyFlights and Synthetic datasets, because they are small enough to perform a large number of experiments changing the parameter configuration. The study involves τ_0^{pre}, τ_0^{npv}, h and k; we varied the value of one parameter at a time, over an appropriate range, while the other parameters remain fixed to the settings specified in Table 5. The parameters α and ρ are omitted, because we borrowed them from paper [26] and we set them according to the heuristics that it suggests; for an analysis of their impact on the performance of algorithms we refer the reader to [26], where they were originally introduced.

Figure 3 shows the results obtained varying the parameter values for the Movies dataset. As is it clear from Fig. 3(b), the value of τ_0^{npv} does not substantially impact the performance, provided that it is not greater than 0.9. On the contrary, more care must be taken in choosing τ_0^{pre}, which should belong to [0.7, 0.95] (Fig. 3a). The exponent h in the denominator of the confidence formula should assume a value greater than 0.8 (Fig. 3c) while, for the exponent k of the authority, values lower than 1 look advisable (Fig. 3d).

The DailyFlights dataset, analyzed in Fig. 4, exhibits a different behavior. Regarding parameters τ_0^{pre}, τ_0^{npv} and h (Fig. 4a, b and c), there is actually just one specific value that needs to be identified in order for STORM to obtain good quality results, while the setting of k (Fig. 4d) does not affect effectiveness. Nevertheless, note that the non-optimal values for τ_0^{pre}, τ_0^{npv} and h lead to extremely poor performances, which can be easily detected by inspecting the veracity values computed by the algorithm.

On the Synthetic dataset, analyzed in Fig. 5, the algorithm achieves the best results if τ_0^{pre} belongs to [0.65, 0.85], while increasing τ_0^{npv} results in a better performance. The best values for h appear to be in [1.0, 1.75], and the choice of k does not have a significant impact in terms of effectiveness.

In summary, the results described in this section highlight that on the Movies dataset STORM converges to good quality results independently of the values of the parameters, if these remain within a (rather wide) range. On the contrary, the setting requires a deeper analysis on the DailyFlights and Synthetic datasets, but the non-optimal configurations seem to be easily recognizable. As explained above, parameters can be assigned by choosing an initial setting relying on their meaning, and then improving this configuration through trial executions of the algorithms and inspections of excerpts of the results: If the computed truth is evidently not correct, for instance because almost all the values are assigned to *true* or to *false*, alternative parameter settings should be experimented. Moreover, for the sake of improving the tuning of parameters, practitioners might also easily build a small golden truth including some tens or hundreds of items; this would allow to further analyze and understand the suitability of the tested configurations.

6.4 Scalability

We now address STORM's scalability. The execution times were measured using a 16-core 4.2 GHz IBM Power7 machine with 128 GB of RAM, running Fedora Linux.

First, we examine the execution times of STORM on the Books dataset (which is the biggest one) by changing the number of objects and sources. To evaluate the execution time by varying the number of objects, we randomly selected 1,000 sources, and incrementally added random objects provided by these sources from 5,000 to 40,000 with

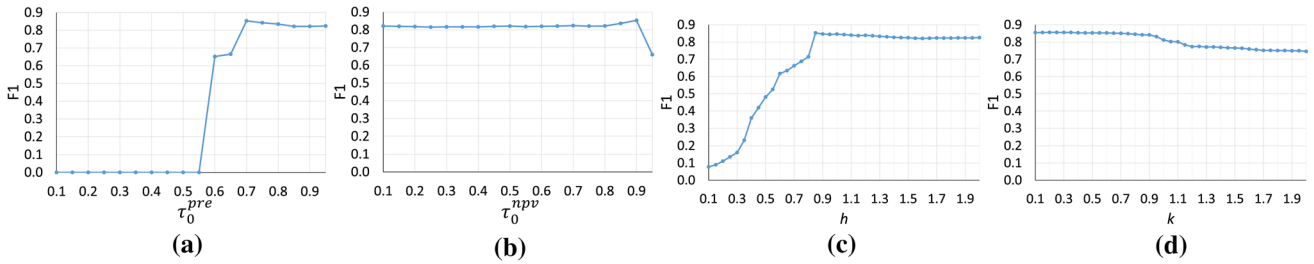

Fig. 3 F1 score varying parameter settings on the Movies dataset. When studying a parameter, the other ones are set as specified in Table 5

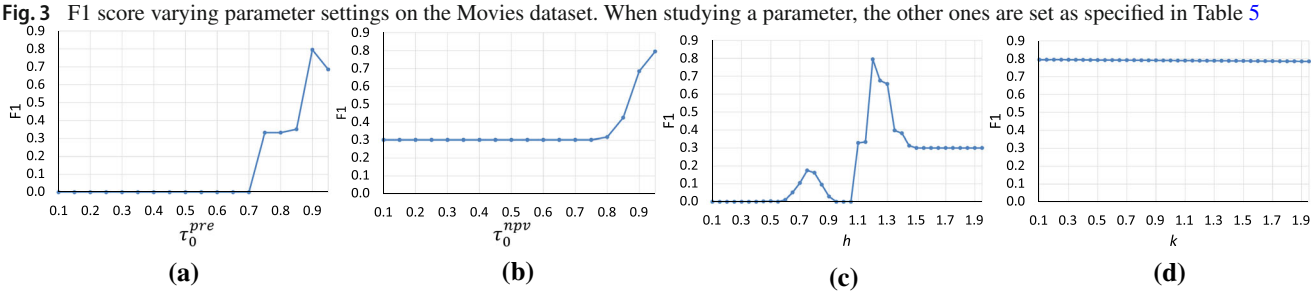

Fig. 4 F1 score varying parameter settings on the Flights dataset. When studying a parameter, the other ones are set as specified in Table 5

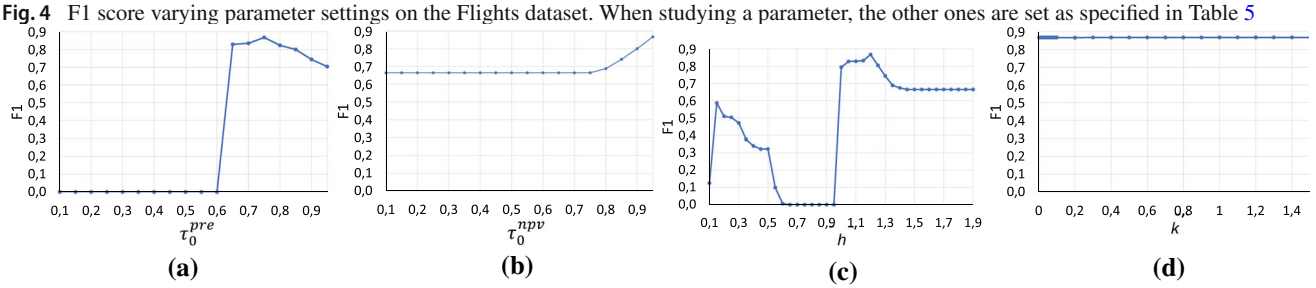

Fig. 5 F1 score varying parameter settings on the Synthetic dataset. When studying a parameter, the other ones are set as specified in Table 5

step 5,000. For varying the number of sources, on the contrary, we randomly selected 10,000 objects and incrementally added random sources providing those objects from 500 to 5,000 with a step of 500. We also conducted an experiment increasing simultaneously and exponentially the numbers of objects and sources. In more detail, the numbers of objects and sources started, respectively, from 2^8 and 2^5, and were doubled at each step. In the evaluation, the authority-based Bayesian inference phase is executed for ten iterations. Note that, in order to highlight the time required by the different components of STORM, the measurements are reported by splitting the algorithm into phases. First, we consider the value reconciliation phase described in Algorithm 2. Then, we also divide the Bayesian step detailed in Algorithm 5 into three sub-phases: preliminary initializations, copy detection, and the remaining computations (including authority, veracity, trustworthiness measures, and true values selection).

The charts are in Figs. 6 (objects), 7 (sources), and 8 (exponential growth of both objects and sources, in logarithmic scale on the x-axis). The plots related to sources and exponential growth report the copy detection separately, otherwise the other curves would not be clearly distinguishable. The first observation that can be drawn from the objects and sources plots is that the copy detection phase is largely the most time-consuming one. Moreover, the trends of the curves reflect those expected from the time complexities of Algorithms 2 and 3. In more detail, all the phases are linear with respect to the number of objects, while copy detection and initialization are quadratic in the number of sources. The curve related to the other computations is quadratic too in the number of sources because it includes the authority computation which is quadratic. The complexity of the reconciliation phase does not depend directly on the number of sources, but it is quadratic in the number of values per object, which in turn is expected to grow with the number of sources; however, on our dataset the trend looks rather linear. Regarding the charts showing the execution time when both objects and sources grow exponentially, with x-axis in logarithmic scale, as expected they turn linear and quadratic trends into exponential ones.

Then, since the Bayesian step of STORM is iterative, we also study how many iterations it requires to reach a good accuracy. Figure 9 shows the measured F1 score after each iteration from 1 to 150, for the three datasets. We note that, on the Books, DailyFlights, and Synthetic datasets, STORM converges extremely fast, reaching 99% of the F1 score in just one, four and eight iterations, respectively. The convergence is not immediate but still reasonably fast on Movies, where,

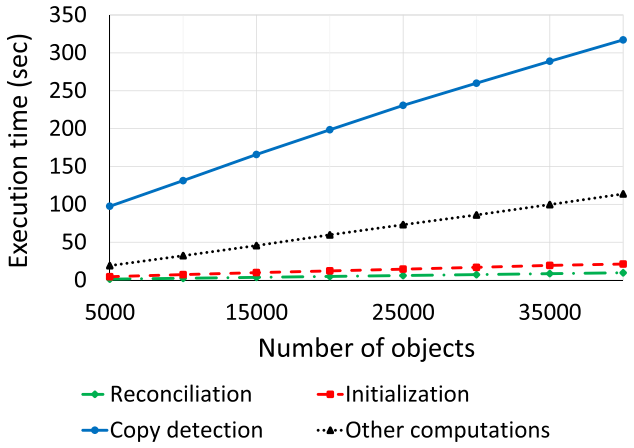

Fig. 6 Execution times of the different phases of the STORM algorithm varying the number of objects for 1,000 sources, on the Books dataset

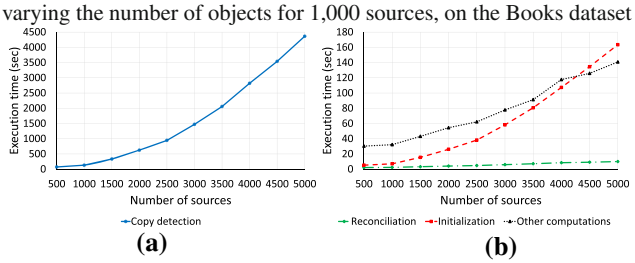

Fig. 7 Execution times of the different phases of the STORM algorithm varying the number of sources for 10,000 objects, on the Books dataset; Fig. 7(a) depicts the copy detection, while Fig. 7(b) includes the remaining phases

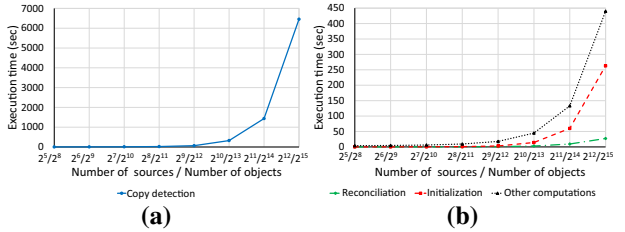

Fig. 8 Execution times of the different phases of the STORM algorithm when both the number of sources and the number of objects grow exponentially, on the Books dataset; Fig 8(a) depicts the copy detection, while Fig. 8(b) includes the remaining phases. The x-axis is in logarithmic scale

to obtain 99% of the F1 score, 47 iterations are needed. The slower convergence on Movies is probably due to the fact that the dataset is smaller and contains fewer source/value associations; as a consequence, the quantities that are updated at each iteration, i.e., value veracities, source trustworthiness and domain copying probabilities, require more iterations to consolidate.

6.5 Case study: estimated authority scores

Let us explore whether the source authorities computed by STORM are reasonable. To this aim we consider the Movies dataset, which has few sources, and analyze the authority scores.

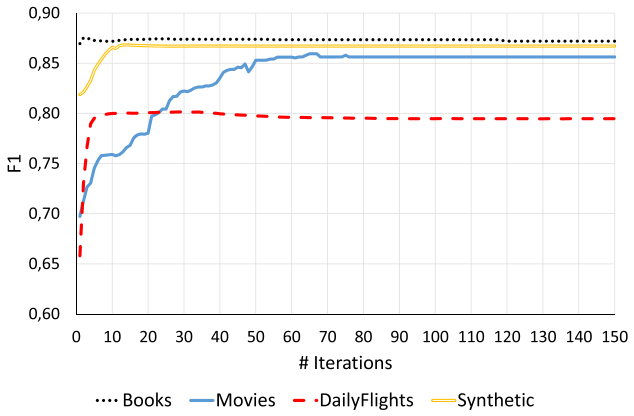

Fig. 9 F1 score obtained by STORM after each iteration of the authority-based Bayesian inference, for the four datasets

Table 8 Authorities in the "comedy" domain and F1 on our golden truth for the Movies sources providing at least 20 comedy movies of the golden truth

Source	Authority	F1
imdb	1.0000	0.8864
metacritic	0.8754	0.8214
top250tv	0.7182	0.8777
1moviesonline	0.6715	0.7826
goodfilms	0.6023	0.7664
flimcrave	0.4141	0.7246
letterboxd	0.0716	0.5920

Table 8 shows, for the popular genre "comedy," the authorities derived by STORM along with the F1 score associated with the values provided by the sources measured on our golden truth, for the sources providing at least 20 comedy movies of the golden truth. Intuitively, the authorities look sound. For instance, the IMDB website obtains the maximum authority, and this is indeed an expected result: it is a very famous source of information about movies, and many other sources copy from it. Moreover, the three sources having the greatest authority are also the three exhibiting the highest F1.

We continued the analysis by plotting F1 scores and authorities for the pairs source/domain of interest associated with at least 20 movies in the golden truth. The plot is in Fig. 10, and also shows the linear regression line (dashed). Observing the plot it is possible to appreciate the growth trend of the F1 score when the authority increases. The close relationship between the two quantities is also confirmed by their high correlation coefficient, which is 0.6999.

6.6 Detailed evaluation of our novel similarity measure

We now compare the performance of our novel similarity measure (Algorithm 1) – employed in our reconciliation algorithm – against the most commonly used string similarity

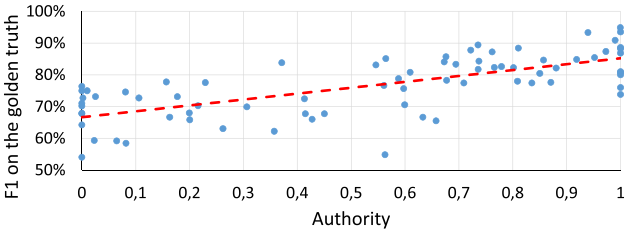

Fig. 10 F1 score measured on the golden truth with respect to the source authority on the Movies dataset, for the source/domain of interest pairs associated with at least 20 movies of the golden truth. Each point corresponds to a source/domain of interest pair, and the dashed red line represents the linear regression line

Table 9 Effectiveness results of string similarity metrics

Method	Threshold	Prec.	Rec.	F1
STORM's similarity	-	0.9222	0.9006	**0.9112**
Hamming [14]	0.22	0.8771	0.5847	0.7017
Levenshtein [19]	0.36	0.8031	0.9064	0.8516
Jaro-Winkler [40]	0.73	0.8726	0.8011	0.8353
Jaccard [16]	0.49	0.8918	0.7719	0.8275
Sørensen [37]	0.66	0.8918	0.7719	0.8275
Ratcliff-Obershelp [35]	0.46	0.8019	0.9707	0.8783
LCS [2]	0.27	0.7962	0.9825	0.8796

metrics. The tests were performed on a subset of the Book dataset, obtained by randomly selecting 100 objects from the Books dataset, for each object we extracted all the possible pairs of values for the attribute "authors" and manually labeled each pair with 1 in case the two values actually represented the same author, and with 0 otherwise. In order to have a balanced dataset, we only selected the objects with up to five different values provided by the sources; this ensures that the number of non-matching pairs is limited. The result is a dataset with 625 tuples, 28% of which labeled as 1 and 72% labeled as 0.

The results in terms of precision, recall and F1 score are presented in Table 9. For what regards the traditional string similarity metrics, we selected the best performances over 100 parametrizations of the threshold used to decide whether two values actually corresponded to the same concept. On the contrary, the threshold for STORM's similarity measure is fixed to 0.5.

To summarize, we can highlight three clear advantages of our algorithm:

- It exploits the concept of *partial token matching* to recognize abbreviations and typos in values; challenges normally encountered by data fusion algorithms that operate on attributes whose values represent person names, addresses, and similar types of data.
- It obtains the best performances, improving the best traditional string similarity metric by 0.0316.
- It does not need to be optimized. This actually is a great advantage, since, as can be seen from the thresholds of the competitors in Table 9, some are very far from 0.5.

6.7 Summary of the evaluation

As a first result, our experimental campaign showed the effectiveness of STORM, comparing it with recent techniques from the literature on three real-world datasets. STORM exhibited the best F1 score jointly with SmartVote [11] on two of the four datasets, and the absolute best F1 score on the others.

Our experiments also highlighted the importance of the value-reconciliation phase in determining the effectiveness of STORM.

In addition, STORM largely outperformed DART [26], which represents the basic Bayesian infrastructure on which we added copy detection, source authority and value reconciliation.

After concentrating on effectiveness, we carried out further experiments to assess specific issues. In particular, STORM showed a good scalability, being linear in the number of objects and quadratic in the number of sources. Moreover, we studied parameter sensitivity, highlighting how suitable parameter settings can be easily identified. Finally, we evaluated the novel similarity measure on which the value reconciliation is founded, emphasizing that it is advantageous with respect to traditional string similarity measures.

7 Conclusion and future work

This paper has presented STORM, a novel multi-truth data-fusion algorithm. STORM is a domain-aware Bayesian algorithm that determines the source trustworthiness relying on the concept of source authority, where a source is regarded as more authoritative if it is copied by many other sources; the copy-detection mechanism we employed considers both directions of copying. Moreover, our technique incorporates a component to identify and group together variant values, in the very common case in which values are represented as textual strings, exploiting a novel token-based similarity measure; to the best of our knowledge, this is the first time that variant values are taken into account in multi-truth data fusion. Our approach has been thoroughly evaluated on one synthetic and three real-world datasets, showing the best performances in comparison with the existing techniques. In more detail, it achieves the absolute best F1 score on two of the four datasets, and the best F1 score jointly with SmartVote [11] on the other two.

We believe that STORM constitutes a valuable alternative in the hands of the practitioners having to build data-fusion systems, since, as already noted, it seems to be able to adapt to different scenarios better than its competitors.

An interesting development of this work might deal with extending STORM to case the objects are associated with time series [24]. For instance, in our DailyFlights dataset, one could think that a pair ⟨route,airline⟩ is connected to a time series with a set of values (i.e., arrival times) for each day. It would be interesting to examine whether the truth of values in a day is connected to that in the previous days, and understand the possible evolution of the authority scores through time. Moreover, a possible research direction to be explored consists in providing the Bayesian inference step of STORM with veracities initialized with the results of an alternative truth discovery technique, instead of using the same default initialization (0.5) for all values. Other developments would be studying similarity measures for non-textual data, and investigating data-fusion techniques dealing with values represented by long textual descriptions such as movie plots or product reviews. Finally, a natural extension of this paper regards the possibility to consider objects with multiple attributes, e.g., to discover the true values for both authors and title of a book. Such a scenario requires to examine the interactions between authority and trustworthiness on different attributes, possibly resorting to joint estimation.

Declaration

Competing Interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Batini, C., Scannapieco, M.: Data and Information Quality - Dimensions,Principles and Techniques. Data-Centric Systems and Applications. Springer (2016)
2. Bergroth, L., Hakonen, H., Raita, T.: A survey of longest common subsequence algorithms. In: Proceedings Seventh International Symposium on String Processing and Information Retrieval. SPIRE 2000, pp. 39–48. IEEE (2000)
3. Bleiholder, J., Naumann, F.: Data fusion. ACM Comput. Surv. **41**(1), 1:1-1:41 (2008)
4. Canalle, G.K., Salgado, A.C., Lóscio, B.F.: A survey on data fusion: what for? in what form? what is next? J. Intell. Inform. Syst. **57**(1), 25–50 (2021)
5. Das Sarma, A., Dong, X.L., Halevy, A.Y.: Data integration with dependent sources. In: Proc. of EDBT 2011, 14th International Conference on Extending Database Technology, pp. 401–412. ACM (2011)
6. Dietterich, T.G.: Approximate statistical test for comparing supervised classification learning algorithms. Neural Comput. **10**(7), 1895–1923 (1998)
7. Dong, X.L., Berti-Équille, L., Srivastava, D.: Integrating conflicting data: The role of source dependence. Proc. VLDB Endowment **2**(1), 550–561 (2009)
8. Dong, X.L., Gabrilovich, E., Murphy, K., Dang, V., Horn, W., Lugaresi, C., Sun, S., Zhang, W.: Knowledge-based trust: Estimating the trustworthiness of web sources. Proc. VLDB Endowment **8**(9), 938–949 (2015)
9. Dong, X.L., Saha, B., Srivastava, D.: Less is more: Selecting sources wisely for integration. Proc. VLDB Endowment **6**(2), 37–48 (2012)
10. Dong, X.L., Srivastava, D.: Big Data Integration. Synthesis Lectures on Data Management. Morgan & Claypool Publishers (2015)
11. Fang, X.S., Sheng, Q.Z., Wang, X., Chu, D., Ngu, A.H.H.: SmartVote: A full-fledged graph-based model for multi-valued truth discovery. World Wide Web **22**(4), 1855–1885 (2019)
12. Ferreira, A.A., Gonçalves, M.A., Laender, A.H.: A brief survey of automatic methods for author name disambiguation. ACM SIGMOD Rec. **41**(2), 15–26 (2012)
13. Galland, A., Abiteboul, S., Marian, A., Senellart, P.: Corroborating information from disagreeing views. In: Proc. of WSDM 2010, 3rd International Conference on Web Search and Web Data Mining, pp. 131–140. ACM (2010)
14. Hamming, R.W.: Error detecting and error correcting codes. Bell Syst. Techn. J. **29**(2), 147–160 (1950)
15. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edition. Morgan Kaufmann (2011)
16. Jaccard, P.: The distribution of the flora in the alpine zone. New Phytol. **11**(2), 37–50 (1912)
17. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proc. of SODA 1998, 9th Symposium on Discrete Algorithms, pp. 668–677. ACM/SIAM (1998)
18. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logist. Q. **2**(1–2), 83–97 (1955)
19. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Phys. Doklady **10**(8), 707–710 (1966)
20. Li, Q., Li, Y., Gao, J., Su, L., Zhao, B., Demirbas, M., Fan, W., Han, J.: A confidence-aware approach for truth discovery on long-tail data. Proc. VLDB Endowment **8**(4), 425–436 (2014)
21. Li, Q., Li, Y., Gao, J., Zhao, B., Fan, W., Han, J.: Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In: Proc. of SIGMOD 2014, International Conference on Management of Data, pp. 1187–1198. ACM (2014)
22. Li, X., Dong, X.L., Lyons, K., Meng, W., Srivastava, D.: Truth finding on the deep web: Is the problem solved? Proc. VLDB Endowment **6**(2), 97–108 (2012)
23. Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., Han, J.: A survey on truth discovery. SIGKDD Explorat. **17**(2), 1–16 (2015)
24. Li, Y., Li, Q., Gao, J., Su, L., Zhao, B., Fan, W., Han, J.: On the discovery of evolving truth. In: Proc. of KDD 2015, 21th International Conference on Knowledge Discovery and Data Mining, pp. 675–684. ACM (2015)
25. Li, Y., Rubinstein, B.I.P., Cohn, T.: Truth inference at scale: A Bayesian model for adjudicating highly redundant crowd annotations. In: Proc. of WWW 2019, 28th International World Wide Web Conference, pp. 1028–1038. ACM (2019)
26. Lin, X., Chen, L.: Domain-aware multi-truth discovery from conflicting sources. Proc. VLDB Endowment **11**(5), 635–647 (2018)
27. Liu, W., Liu, J., Wei, B., Duan, H., Hu, W.: A new truth discovery method for resolving object conflicts over Linked Data with scale-free property. Knowl. Inf. Syst. **59**(2), 465–495 (2019)
28. Liu, X., Dong, X.L., Ooi, B.C., Srivastava, D.: Online data fusion. Proc. VLDB Endowment **4**(11), 932–943 (2011)
29. Lyu, S., Ouyang, W., Wang, Y., Shen, H., Cheng, X.: Truth discovery by claim and source embedding. IEEE Trans. Knowl. Data Eng. **33**(3), 1264–1275 (2021)
30. Ma, F., Li, Y., Li, Q., Qiu, M., Gao, J., Zhi, S., Su, L., Zhao, B., Ji, H., Han, J.: FaitCrowd: Fine grained truth discovery for crowdsourced

data aggregation. In: Proc. of KDD 2015, 21th International Conference on Knowledge Discovery and Data Mining, pp. 745–754. ACM (2015)
31. Pasternack, J., Roth, D.: Knowing what to believe (when you already know something). In: Proc. of COLING 2010, 23rd International Conference on Computational Linguistics, pp. 877–885. Tsinghua University Press (2010)
32. Pasternack, J., Roth, D.: Latent credibility analysis. In: Proc. of WWW 2013, 22nd International World Wide Web Conference, pp. 1009–1020. ACM (2013)
33. Pochampally, R., Das Sarma, A., Dong, X.L., Meliou, A., Srivastava, D.: Fusing data with correlations. In: Proc. of SIGMOD 2014, International Conference on Management of Data, pp. 433–444. ACM (2014)
34. Ramshaw, L., Tarjan, R.E.: On minimum-cost assignments in unbalanced bipartite graphs. HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1 (2012)
35. Ratcliff, J.W., Metzener, D.E.: Pattern matching: The Gestalt approach. Dr Dobbs J. **13**(141), 46–51 (1988)
36. Rekatsinas, T., Joglekar, M., Garcia-Molina, H., Parameswaran, A.G., Ré, C.: Slimfast: Guaranteed results for data fusion and source reliability. In: Proc. of SIGMOD 2017, International Conference on Management of Data, pp. 1399–1414. ACM (2017)
37. Sørensen, T.A.: A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. Kongelige Danske Videnskabernes Selskab **5**(4), 1–34 (1948)
38. Wang, X., Sheng, Q.Z., Fang, X.S., Yao, L., Xu, X., Li, X.: An integrated Bayesian approach for effective multi-truth discovery. In: Proc. of CIKM 2015, 24th International Conference on Information and Knowledge Management, pp. 493–502. ACM (2015)
39. Wang, X., Sheng, Q.Z., Yao, L., Li, X., Fang, X.S., Xu, X., Benatallah, B.: Truth discovery via exploiting implications from multi-source data. In: Proc. of CIKM 2016, 25th International Conference on Information and Knowledge Management, pp. 861–870. ACM (2016)
40. Winkler, W.E.: String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In: Proc. of the Section on Survey Research Methods, American Statistical Association (1990)
41. Xiao, H., Gao, J., Li, Q., Ma, F., Su, L., Feng, Y., Zhang, A.: Towards confidence interval estimation in truth discovery. IEEE Trans. Knowledge Data Eng. **31**(3), 575–588 (2019)
42. Yang, J., Tay, W.P.: An unsupervised Bayesian neural network for truth discovery in social networks. IEEE Trans. Knowledge Data Eng. (2021)
43. Ye, C., Wang, H., Zheng, K., Kong, Y., Zhu, R., Gao, J., Li, J.: Constrained truth discovery. IEEE Trans. Knowledge and Data Eng. (2020)
44. Yin, X., Han, J., Yu, P.S.: Truth discovery with multiple conflicting information providers on the web. IEEE Trans. Knowl. Data Eng. **20**(6), 796–808 (2008)
45. Yin, X., Tan, W.: Semi-supervised truth discovery. In: Proc. of WWW 2011, 20th International Conference on World Wide Web, pp. 217–226. ACM (2011)
46. Zhang, D., Wang, D., Vance, N., Zhang, Y., Mike, S.: On scalable and robust truth discovery in big data social media sensing applications. IEEE Trans. Big Data **5**(2), 195–208 (2019)
47. Zhang, H., Li, Q., Ma, F., Xiao, H., Li, Y., Gao, J., Su, L.: Influence-aware truth discovery. In: Proc. of CIKM 2016, 25th International Conference on Information and Knowledge Management, pp. 851–860. ACM (2016)
48. Zhang, J., Wu, X.: Multi-label truth inference for crowdsourcing using mixture models. IEEE Trans. Knowledge and Data Eng. **33**(5), 2083–2095 (2021)
49. Zhang, L., Qi, G., Zhang, D., Tang, J.: Latent dirichlet truth discovery: Separating trustworthy and untrustworthy components in data sources. IEEE Access **6**, 1741–1752 (2018)
50. Zhao, B., Han, J.: A probabilistic model for estimating real-valued truth from conflicting sources. In: Proc. of QDB 2012, 10th International Workshop on Quality in Databases (2012)
51. Zhao, B., Rubinstein, B.I.P., Gemmell, J., Han, J.: A Bayesian approach to discovering truth from conflicting sources for data integration. Proc. VLDB Endowment **5**(6), 550–561 (2012)