Knowledge-Based Systems 176 (2019) 97–109

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

PatternFinder: Pattern discovery for truth discovery

Chen Ye [a], Hongzhi Wang [a,*], Tingting Ma [a], Jing Gao [b], Hengtong Zhang [b], Jianzhong Li [a]

[a] Harbin Institute of Technology, PR China
[b] SUNY Buffalo, USA

HIGHLIGHTS

- First study to discover patterns for truth discovery.
- An optimization problem is formulated to discover the patterns.
- An iterative algorithm is proposed to jointly and iteratively learn the variables.
- Experimental results show the effectiveness and efficiency of the proposed algorithm.

ARTICLE INFO

Article history:
Received 22 August 2018
Received in revised form 21 March 2019
Accepted 23 March 2019
Available online 28 March 2019

Keywords:
Truth discovery
Pattern discovery
Unsupervised learning

ABSTRACT

Truth discovery methods infer truths from multiple sources. These methods usually resolve conflicts based on the information on the entity level. However, due to the existence of incompleteness and the difficulty in entity matching, the information on the individual entity is often insufficient. This motivates pattern discovery, which aims to mine useful patterns across entities from a global perspective. In this paper, we introduce pattern discovery for truth discovery and formulate it as an optimization problem. To solve such a problem, we propose an algorithm called PatternFinder that jointly and iteratively learns the variables. Additionally, we also propose an optimized grouping strategy to enhance its efficiency. Experimental results on simulated and real-world datasets demonstrate the advantage of the proposed methods, which outperform the state-of-the-art baselines in terms of both effectiveness and efficiency.

1. Introduction

It is critical to identify correct information from multi-source conflicting data. Such a task is called truth discovery [1,2]. A straightforward truth discovery approach is to conduct majority voting or averaging. However, the most significant shortcoming of such approaches is that they assume all the sources are equally reliable. As information quality usually varies a lot among different sources [2], such approaches may not achieve correct results.

To improve the performance, various truth discovery methods [1,3–8] are proposed. In these methods, a common principle is applied. That is, if an entity's information provided by a source is often supported by other sources, the source is regarded as a reliable one, and in turn, its information is more likely to be true. It can be inferred that for one entity, its correct values are found by resolving the conflicts among multiple sources. Regardless of the duplicate situation among different sources [3], for a set of entities, the greater the number of sources that provide information for each entity, the more likely we can identify the reliable sources and find the truths.

Unfortunately, when the evidence is insufficient on the entity level, the picture is different. The insufficient evidence makes it difficult for existing methods to identify the reliable sources and find the truths, especially for entities covered mostly by the unreliable sources. For the circumstance where there is insufficient evidence on the entity level [9,10], we first analyze its causes in three aspects, i.e., *long-tail phenomenon*, *mismatching*, and *incompleteness*.

- *Long-tail phenomenon*. The phenomenon where the entities' information is provided by very few sources is common in applications [4,9]. For one source, it may contain the information about a large number of entities. However, most of the entities in this source may not have the corresponding information in the other sources.
- *Mismatching*. In many applications, it is common that each source has its own entity identifier and becomes an isolated island of information [10]. To identify each entity's information from multiple sources, a natural way is to first conduct entity matching. However, due to erroneous values

* Corresponding author.
E-mail addresses: yech@hit.edu.cn (C. Ye), wangzh@hit.edu.cn (H. Wang), hitmtt@stu.hit.edu.cn (T. Ma), jing@buffalo.edu (J. Gao), hengtong@buffalo.edu (H. Zhang), lijzh@hit.edu.cn (J. Li).

https://doi.org/10.1016/j.knosys.2019.03.027
0950-7051/

Table 1
Patient information from three hospitals.

		Name	Age	Condition	Measure
$\mathcal{X}_1$	$\boldsymbol{o}_1$	Mike	23	Fever	Febrifuge
	$\boldsymbol{o}_2$	–	35	Stroke	Thrombolytic
$\mathcal{X}_2$	$\boldsymbol{o}_3$	Angela	30	*Feven* (fever)	Febrifuge
	$\boldsymbol{o}_4$	Bob	20	Stroke	*Warfarin* (thrombolytic)
$\mathcal{X}_3$	$\boldsymbol{o}_5$	Angela	–	Fever	Febrifuge
	$\boldsymbol{o}_6$	Jim	41	*Stoke* (stroke)	Thrombolytic

Table 2
Example patterns.

Applied set	Condition	Measure
$\{\boldsymbol{o}_1, \boldsymbol{o}_3, \boldsymbol{o}_5\}$	Fever	Febrifuge
$\{\boldsymbol{o}_2, \boldsymbol{o}_4, \boldsymbol{o}_6\}$	Stroke	Thrombolytic

in the multi-source noisy data, it is hard to correctly link the records [11–14], which also results in insufficient evidence for the entities.

- *Incompleteness*. Due to the incomplete entry, inaccurate extraction or heterogeneous schemas, it is very prevalent that sources only provide information for a subset of attributes about a given entity [15]. Thus, even if entity matching is feasible and effective, enough information for each attribute of the entities cannot be guaranteed.

Due to the existence of insufficient evidence on the entity level in these three aspects, it is challenging to find the truths about the multi-source unaligned data. We use an example to illustrate how existing methods work that motivates our approach.

Example 1. Table 1 contains six records collected from three hospitals $\{\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3\}$. Each record $\boldsymbol{o}$ specifies a patient described by four attributes: name, age, condition, and measure, among which the condition denotes the clinical symptom of the patient and measure denotes the therapeutic drug for the patient. All erroneous values are marked in italics and their correct values are given in the following brackets. Note that we do not know whether the records provided by the different hospitals refer to the same patient, and missing values are represented as "-".

Based on this example, we can see that $\boldsymbol{o}_1, \boldsymbol{o}_2, \boldsymbol{o}_4, \boldsymbol{o}_6$ tend to refer to different patients, as they have dissimilar attribute values. For $\boldsymbol{o_3}$ and $\boldsymbol{o_5}$, though they may refer to the same person (same name and measure), due to the wrong value "feven" of the condition from o_3 and the missing value of age from o_5, there is not enough information to link o_3 and o_5. As a result, $\boldsymbol{o}_1, \boldsymbol{o}_2, \boldsymbol{o}_3, \boldsymbol{o}_4, \boldsymbol{o}_5, \boldsymbol{o}_6$ will be all treated as a record for a separate entity. However, only one piece of information for each patient is insufficient for the existing methods [3–8,16–19] to infer the truths and identify the reliable sources. Given $\boldsymbol{o}_1$ provided by $\mathcal{X}_1$, there are two circumstances: (1) $\mathcal{X}_2$ and $\mathcal{X}_3$ may provide the same information as $\boldsymbol{o}_1$ to support $\boldsymbol{o}_1$ to be true; (2) $\mathcal{X}_2$ and $\mathcal{X}_3$ may provide different information from $\boldsymbol{o}_1$, then the information provided by the most reliable source is true. Hence, more evidence is needed from $\mathcal{X}_2$ and $\mathcal{X}_3$. Without more evidence about these patients, existing methods will consider $\boldsymbol{o}_1, \boldsymbol{o}_2, \boldsymbol{o}_3, \boldsymbol{o}_4, \boldsymbol{o}_5, \boldsymbol{o}_6$ all to be true, and fail to find the true value of $\boldsymbol{o}_3$, $\boldsymbol{o}_4$, and $\boldsymbol{o}_6$, e.g., "fever" for "feven" for the condition of $\boldsymbol{o}_3$. Moreover, considering these records all to be true will draw a conclusion that all the sources are reliable, while the fact is that $\mathcal{X}_2$ and $\mathcal{X}_3$ are not very reliable as they contain several errors.

Observations. The above example indicates that truth discovery methods will become less effective when faced with *long-tail phenomena*, *mismatching*, and *incompleteness* issues on the entity level. Fortunately, such entities may still find counterparts that share similar patterns. Consider hospitals and social forums as examples. Patients from different hospitals may be different, but the properties (e.g., symptoms, medical history, demographics) of patients with the same disease could be quite comparable; multiple online social forums may attract overlapping but different sets of users, in which user communities and community patterns may be shared across platforms. Therefore, when the evidence is not sufficient on the entity level, the latent patterns shared among different entities would be helpful to discover the truths of the entities.

Pattern Discovery. Motivated by these observations, in this study, we propose to leverage pattern discovery for truth discovery on multi-source unaligned data. A pattern is a triple variable that contains an applied set, an attribute set, and a value combination towards the attribute set. For each pattern, the applied set precisely describes the scope that the pattern is suitable for. If a record is in the applied set of a pattern, its values on the attribute set match the value combination of the pattern.

Example 2. Table 2 shows two patterns whose attribute set contains the condition and measure. For the first pattern, its applied set is $\{\boldsymbol{o}_1, \boldsymbol{o}_3, \boldsymbol{o}_5\}$, and the value combination is (fever, febrifuge). It states that for $\{\boldsymbol{o}_1, \boldsymbol{o}_3, \boldsymbol{o}_5\}$, their values for the condition and measure should be "fever" and "febrifuge", respectively. Considering $\boldsymbol{o}_3$, the value "feven" for the condition will be corrected to "fever". Similarly, the second pattern states that for $\{\boldsymbol{o}_2, \boldsymbol{o}_4, \boldsymbol{o}_6\}$, their values for the condition and measure should be "stroke" and "thrombolytic", respectively. The errors in $\boldsymbol{o}_4$ and $\boldsymbol{o}_6$ will then also be corrected.

Based on this example, we can infer that, when the evidence is insufficient on the entity level, matching the corresponding patterns can help to improve the performance of truth discovery. Therefore, it is crucial to design algorithms for all the entities across sources so that the patterns shared among them can be automatically discovered. However, discovering proper patterns raises several challenges.

- As no oracle tells which attribute can make up the attribute set of the patterns, the question is how to infer the attribute set so that it can accurately cover all and only the significant attributes.
- With errors in the multi-source records, the concern is how to generate the value combinations concisely enough to be close to the true ones.
- We need to accurately apply the patterns to each record, and efficiently find the applied set for each pattern.

In this paper, we jointly address these issues. First, to obtain the attribute set, we assign an attribute weight to each attribute. The higher the weight of an attribute, the higher the possibility that it belongs to the attribute set. Second, to ensure the accuracy of the value combinations, a source weight is assigned to each source, which indicates that the information provided by the sources with higher weights are more reliable. Third, to find the applied set for each pattern, we aim to infer the latent groups which share the same pattern. The patterns can then be discovered by inferring the group-level representatives and applied to the group members. In summary, we model the pattern discovery problem by an optimization framework, where the latent groups, the group-level representatives, the source weights, and the attribute weights are defined as four sets of variables. The objective is to minimize the overall weighted deviation between the group-level representatives and the multi-group records. We propose an algorithm to solve the optimization problem by iteratively updating the four sets of variables. Benefiting from the iterative

procedure, we can achieve high-quality patterns on significant attributes provided by reliable sources. By using a global analysis of all the entities, the proposed algorithm can take advantage of more evidence from the entities. The generated patterns will be useful in detecting the high-level behavior of entities observed from multiple perspectives, such as user community patterns of multiple social networks [20], patient symptoms recorded and diagnosed by multiple hospitals [21], and traffic features captured by multiple sensors [22].

Contributions. We summarize our contributions as follows.

- We study pattern discovery for truth discovery. We formally define patterns and then formulate an optimization problem to discover the patterns.
- We propose an iterative algorithm called PatternFinder to solve the problem by jointly inferring the latent groups, the group-level representatives, the source weights, and the attribute weights.
- To improve the efficiency, we enhance PatternFinder by an optimized grouping strategy.
- We present extensive experiments with simulated and real-world datasets. The experimental results clearly demonstrate the advantages of PatternFinder compared to the baselines.

Organization. We analyze related work in Section 2 and define the problem of pattern discovery in Section 3. Section 4 describes the overall solution and the main component PatternFinder, followed by the experimental results in Section 5. We conclude the paper with final remarks in Section 6.

2. Related work

The traditional problem of truth discovery has been studied for years to resolve conflicts among multiple sources [1,4]. Existing approaches [3–8,16,18,19,23–29] adopt a common principle that if the information provided by a source is often supported by other sources, the source is regarded as a reliable one, and in turn, its information is more likely to be true. Thus, for one entity, the value provided by reliable sources will be regarded as correct. To enlarge the scope of applications, these approaches consider various scenarios. In [3,18], the authors analyze the relationship among the sources, in which the sources are not independent, and they may copy from each other. The proposed approaches in [16,24,26,27] take into account the correlations among the entities, such as temporal factors [16,26], common senses [24] and functional dependency constraints [27]. The proposed approaches in [4,7] deal with the long-tail phenomenon, in which most sources only provide a few claims, and only a few sources make plenty of claims. The existence of multiple truths for a single entity is considered in [6,8,19] in which the source reliability is modeled as two-sided, i.e., sensitivity and specificity. The difficulty of obtaining the truths is considered in [25]. Recently, Beretta et al. considered the problem of leveraging the partial order among the claims [23]. Zhang et al. considered the problem of truth discovery on textual data [28]. However, all these approaches rely on the information provided on the entity level. When the information on the entity level is incomplete, they will fail to obtain the correct result. Different from these approaches, such a case can be handled by our proposed framework, since we can correct the errors by inferring the latent patterns existing in the data from a global view.

Another line of related work is known as multi-view clustering [30–35], which aims to obtain an accurate clustering result by taking advantage of information from multiple views. The key of multi-view clustering is to explore diverse information from multiple feature sets, and simultaneously uncover the consistent cluster structure of the dataset. Our pattern discovery method shares a similar goal, which aims to find a common pattern across various sources. Comparing these multi-view clustering methods, our methods do not assume that observations from different sources are aligned, and we also take source reliability into account when deriving the group-level truths.

There has also been a large body of work on entity resolution [11,12,36], which aims to identify the records in one or multiple data sources that correspond to the same real-world entity. In contrast to that line of work, we do not identify multi-source information for one specific entity. Instead, we focus on the discovery of latent patterns shared among the entities. As we take the source reliability into account in the discovery procedure, the latent groups achieved by our methods can also be treated as the meaningful blocks in the entity matching, which will improve the performance of entity matching across multiple sources of heterogeneous data.

3. Problem definition

We first define useful terms for the multi-source data and then give the problem definition.

An *entity* is a person or thing of interest. An *attribute* is a feature used to describe the entity. A *data item* is a paired entity-attribute. A *data source* describes the place where information about data items can be collected. A *claim* is a value of a data item provided by a data source. A *record* contains all the claims about an entity provided by a data source. Suppose there are K data sources, each of which contains several records for a set of entities. The entities being observed by different sources could be different. Even if some entities are overlapping, we do not know the link across their sources. Suppose that each record has M attributes $\{A_1, \ldots, A_M\}$. The ith record is denoted as $\boldsymbol{o}_i = \{v_{i1}, v_{i2}, \ldots, v_{iM}\}$, where v_{im} is the value of record $\boldsymbol{o}_i$ subject to A_m. Let $\mathcal{X}_k = \{\boldsymbol{o}_i\}_{i=1}^{N_k}$ denote the record collection of the kth source, where N_k is the total number of records in the kth source. A record collection $\mathcal{D} = \bigcup_{k \in K} \mathcal{X}_k$ made up of $n = \sum_{k=1}^{K} N_k$ records from K sources is then generated.

Definition 1 (*Patterns*)**.** A pattern φ_l defined on $\mathcal{D}$ is a triple variable $(R_l, X, \boldsymbol{t}_{lX})$ where

1. R_l is an applied set which is made up of records $\boldsymbol{o}_i \in \mathcal{D}$;
2. X is a set of attributes in $\{A_1, \ldots, A_M\}$;
3. $\boldsymbol{t}_{lX}$ is a value combination on X. For each attribute $A_m \in X$, t_{lm} is a constant value in the domain of A_m specified in $\mathcal{D}$.

We can infer that, to avoid one record appearing in the applied set of two patterns, each pair of applied sets of the patterns must be disjoint. Thus, the problem of pattern discovery can be treated as a task of inferring the latent groups, which is defined as follows.

Definition 2 (*Latent Groups*)**.** Given the number L of the latent groups, $\mathcal{G}$ is a $n \times L$ partition matrix whose element g_{il} denotes the group indicators for $\boldsymbol{o}_i \in \mathcal{D}$, i.e., $g_{il} = 1$, if $\boldsymbol{o}_i$ belongs to group l, otherwise $g_{il} = 0$. The latent groups $\{C_1, C_2, \ldots, C_L\}$ are then formed, where the lth group C_l is made up of records $\boldsymbol{o}_i$ whose group indicators $g_{il} = 1$.

Example 3. As shown in Table 2, given $L = 2$, $C_1 = \{\boldsymbol{o}_1, \boldsymbol{o}_3, \boldsymbol{o}_5\}$ and $C_2 = \{\boldsymbol{o}_2, \boldsymbol{o}_4, \boldsymbol{o}_6\}$ are two latent groups formed by the records in Table 1.

Given the latent groups, the patterns are found by inferring the group-level representatives and applied to the group members.

Definition 3 (*Group-level Representatives*)**.** For group C_l, the group-level representative is denoted as $\boldsymbol{c}_l = \{c_{l1}, c_{l2}, \ldots, c_{lM}\}$, where c_{lm} is the most representative value, i.e., group-level truth, for A_m among C_l. In total, the collection of the group-level representatives is $\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_L\}$.

Example 4. Consider Example 3. $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are the group-level representatives for the latent group C_1 and C_2, respectively.

Benefiting from the inference of the latent groups and the group-level representatives, for each pattern φ_l, R_l corresponds to the group C_l, and $\boldsymbol{t}_{lX}$ is achieved from the group-level representative $\boldsymbol{c}_l$ specific to the attribute set X. To find the proper X and $\mathcal{C}$, an attribute weight and a source weight are assigned to each attribute and each source, respectively.

Definition 4 (*Attribute Weights*)**.** Attribute weights are denoted as $\mathcal{P} = \{p_1, p_2, \ldots, p_M\}$ in which p_m is the significance score of A_m. A higher p_m indicates that A_m is more significant and more likely to be a part of X.

Example 5. Consider Table 1. Suppose that the attribute weights of four attributes are $p_1 = p_2 = 0.06$, $p_3 = 0.23$, and $p_4 = 0.65$, we consider that the condition and measure with higher weights compose the attribute set X with a higher probability.

Remark. With the existence of patterns, a direct observation is that, in each latent group, records may share similar values on several attributes and these attributes form X. Therefore, to infer X, we estimate p_m for A_m by evaluating the differences between the group members' values and the corresponding group-level truths. If the differences are small, p_m should be high, and A_m is more likely to a part of X.

Definition 5 (*Source Weights*)**.** Source weights are denoted as $\mathcal{W} = \{w_1, w_2, \ldots, w_K\}$ in which w_k is the reliability score of the kth source. A higher w_k indicates that the kth source is more reliable, and claims provided by this source are more likely to be the group-level truths.

Example 6. Consider Table 1. Suppose that the source weights of three sources are $w_1 = 0.6$, $w_2 = 0.3$, and $w_3 = 0.1$, we consider that the records provided by source 1 are more reliable than the records provided by source 2 and source 3, and the claims provided by source 1 are regarded as the group-level truths.

Remark. Different sources usually have different reliability degrees. The existing truth discovery methods [2,3] are proposed to estimate the source reliability. The main idea is that the sources providing a larger number of true claims will be assigned higher reliability degrees (a.k.a, source weights), and the claims provided by the reliable sources will be regarded as the correct values. For the proposed method, the difference from existing methods is that we estimate the source weights by evaluating the true values subject to a set X of significant attributes with high attribute weights. Therefore, when errors are involved in X, we can obtain the correct $\boldsymbol{t}_{lX}$ by treating the claims provided by the reliable sources as true values.

Problem definition. Given a collection $\mathcal{D}$ of unaligned records from K sources and L latent groups, we attempt to accurately infer the group indicators $\mathcal{G}$, the group-level representatives $\mathcal{C}$, the attribute weights $\mathcal{P}$, as well as the source weights $\mathcal{W}$, such that the patterns with maximum precision can be achieved.

4. Methodology

In this section, we formally introduce the approach of pattern discovery for truth discovery. We first provide the whole solution overview in Section 4.1. To achieve accurate patterns, we propose an optimization framework in Section 4.2 and solve it through the iterative algorithm PatternFinder in Section 4.3. To improve the efficiency, we then develop a scalable strategy for PatternFinder in Section 4.4. Finally, we discuss how to generate patterns and truths according to the output of PatternFinder in Section 4.5.

4.1. Solution overview

For the multi-source unaligned data, facing the insufficient information for one entity, it is difficult to infer its true value. To tackle this issue, we propose to discover the patterns existing in the multi-source data. Therefore, the true values of different entities sharing similar properties could be inferred from one pattern. To achieve the accurate patterns, we attempt to learn four variables, i.e., the latent groups $\mathcal{G}$, the group-level representatives $\mathcal{C}$, the attribute weights $\mathcal{P}$, and the source weights $\mathcal{W}$. The applied sets and the attribute set of the patterns are then representatively inferred from the latent groups $\mathcal{G}$ and the attribute weights $\mathcal{P}$. Meanwhile, the value combinations of the patterns are determined according to the group-level representatives $\mathcal{C}$ specific to the attribute set.

Fig. 1 is an illustration of the entire solution. The core component is the PatternFinder algorithm, which jointly learns the four variables $\mathcal{G}$, $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$ from the multi-source unaligned data. To improve the efficiency, we developed an optimization grouping strategy. The patterns are then achieved by the pattern generation module, and the truths are found by the truth generation module.

4.2. Optimization framework

We next propose an optimization framework to jointly learn the group indicators, the group-level representatives, the attribute weights, and the source weights. The basic idea is that for each latent group, reliable sources provide trustworthy claims and significant attributes form the attribute set of patterns. The group-level representatives should then be close to the claims from reliable sources on significant attributes. Thus, we should minimize the overall weighted deviation from the group-level representatives to the multi-group records, where each source is weighted by its reliability, and each attribute is weighted by its importance. Based on this principle, we propose the following optimization framework.

$$\begin{aligned}\min_{\mathcal{G},\mathcal{C},\mathcal{P},\mathcal{W}} f(\mathcal{G},\mathcal{C},\mathcal{P},\mathcal{W}) = &\sum_{l=1}^{L}\sum_{k=1}^{K}\sum_{o_i\in\mathcal{X}_k} g_{il} w_k \sum_{m=1}^{M} p_m d_m(v_{im}, c_{lm}) \\ &+ \alpha \sum_{m=1}^{M} p_m \log(p_m)\end{aligned} \tag{1}$$

subject to

$$\begin{cases} \sum_{l=1}^{L} g_{il} = 1, g_{il} \in \{0,1\}, 1 \le i \le n, \\ \sum_{m=1}^{M} p_m = 1, 0 \le p_m \le 1, \\ \sum_{k=1}^{K} \exp(-w_k) = 1. \end{cases} \tag{2}$$

We are trying to search for the values for the following four sets of unknown variables: group indicators $\mathcal{G}$, group-level representatives $\mathcal{C}$, attribute weights $\mathcal{P}$, and source weights $\mathcal{W}$, by minimizing the objective function $f(\mathcal{G},\mathcal{C},\mathcal{P},\mathcal{W})$. There are three types of functions that need to be plugged into this framework.

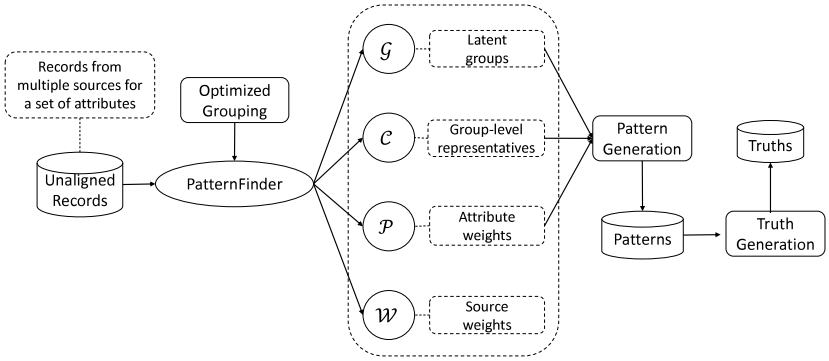

Fig. 1. The whole solution overview.

- *Loss function.* d_m refers to a loss function defined based on the data type of A_m. This function measures the distance between v_{im} and c_{lm}. If the attribute is numerical, then

$$d_m(v_{im}, c_{lm}) = (c_{lm} - v_{im})^2. \tag{3}$$

If the attribute is categorical, then

$$d_m(v_{im}, c_{lm}) = \begin{cases} 1 & \text{if } v_{im} \neq c_{lm}, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

- *Balance function.* A negative attribute weight entropy term is added to the framework as a balance function, where a given positive parameter α is used to control the attribute weight distribution. A large α means that more attributes contribute to the grouping, while a small α allows only significant attributes to contribute to the grouping;
- *Regularization function.* To constrain the group indicators $\mathcal{G}$, the attribute weights $\mathcal{P}$, and the source weights $\mathcal{W}$ into a certain range, we specify the regularization functions in Eq. (2). We use the exponential function to constrain $\mathcal{W}$, as it is a reasonable constraint function that leads to meaningful source weights [4];

Intuitively, if a source is more reliable (i.e., w_k is high) and an attribute A_m is more significant (i.e., p_m is high), we trust the source's information on A_m more in determining the group-level truths. That is, we give a higher penalty when the group-level truth c_{lm} deviates from the value v_{im} provided by the kth source. On the other hand, the penalty is lower when v_{im} is either from unreliable sources with a smaller w_k or towards insignificant attributes with a smaller p_m.

4.3. Pattern-Finder Algorithm

According to the optimization framework, the group indicators $\mathcal{G}$, the group-level representatives $\mathcal{C}$, the attribute weights $\mathcal{P}$, and the source weights $\mathcal{W}$ are learned together by optimizing the objective function through a joint procedure. However, it is difficult to directly calculate the four sets of variables. Therefore, we iteratively updated the values of one set to minimize the objective function while keeping the values of the other sets unchanged until convergence. This iterative four-step procedure, referred as the block coordinate approach [37], will keep reducing the value of the objective function. To minimize the objective function in Eq. (1), we iteratively conducted the following steps.

Step 1: Attribute weights update. With the initial estimates of $\mathcal{G}$, $\mathcal{C}$, and $\mathcal{W}$, we first weigh each attribute based on the differences between the group members' values and the corresponding group-level truths. We then obtain an initial estimate of the attribute weights by assigning higher weights to the attributes with smaller differences. At this step, we fix the values for $\mathcal{G}$, $\mathcal{C}$, and $\mathcal{W}$, and compute the attribute weights that jointly minimize the objective function in Eq. (1) subject to the regularization constraint in Eq. (2). Through Lagrange multipliers, we derive:

$$p_m = \frac{\exp\left\{\frac{-D_m - \alpha}{\alpha}\right\}}{\sum_{m'=1}^{M} \exp\left\{\frac{-D_{m'} - \alpha}{\alpha}\right\}}, \tag{5}$$

where

$$D_m = \sum_{l=1}^{L} \sum_{k=1}^{K} \sum_{i \in \mathcal{X}_k} g_{il} w_k d_m(v_{im}, c_{lm}). \tag{6}$$

Step 2: Group indicators update. After the update of the attribute weights, we update the group indicators so that records sharing similar values on the attributes with higher weights are clustered into the same group. At this step, the values for $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, and each record $\boldsymbol{o}_i \in \mathcal{D}$ is assigned to a group which minimizes the weighted distance between the group members' values and the corresponding group-level truths according to Theorem 1.

Theorem 1. *Suppose that $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, the group indicator g_{il} towards the record $\boldsymbol{o}_i \in \mathcal{D}$ is achieved by*

$$\begin{cases} g_{il} = 1, & \text{if } S_{il} \leq S_{il'} \text{ for } 1 \leq l' \leq L, \\ & \text{where } S_{il'} = \sum_{m=1}^{M} p_m d_m(v_{im}, c_{l'm}), \\ g_{il'} = 0, & \text{for } l' \neq l. \end{cases} \tag{7}$$

Proof. Since $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, the optimization problem Eq. (1) has only one set $\mathcal{G}$ with variables. With the regularization constraint in Eq. (2), it is obvious that for a record $\boldsymbol{o}_i$, when we assign g_{il} according to Eq. (7), the item $\sum_{l=1}^{L} \sum_{m=1}^{M} g_{il} w_k p_m d_m(v_{im}, c_{lm})$ is minimal. As the records from multiple sources are independent of each other, the objective function in Eq. (1) is minimal.

Remark. Under the assumption that each pair of applied sets of the patterns to be disjoint, each record belongs to at most one group (i.e., one pattern), where g_{il} is assigned to 0 or 1. However, there does exist the situation that a single record may be associated with multiple patterns. To handle this case, similar to extending K-means to Fuzzy-C-means [38], we adjust the scope of the group indicator g_{il} to be [0, 1] instead of {0, 1}. g_{il} can then be updated. Through this adjustment, for each record $\boldsymbol{o}_i$, it will be associated with multiple groups (i.e., multiple patterns) whose group indicators $g_{il} > 0$.

Step 3: Group-level representatives update. After the update of the group indicators, we update the group-level representatives.

When $\mathcal{G}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, the group-level truths are updated based on the members in the groups. Each member is weighted by the source that provides it. Therefore, the more accurate group-level truths are discovered by trusting the information provided by the sources with higher weights.

For the numerical data, when $\mathcal{G}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, based on the loss function defined in Eq. (3), the group-level truth c_{lm} should be:

$$c_{lm} = \frac{\sum_{k=1}^{K} \sum_{i \in \mathcal{X}_k} g_{il} w_k v_{im}}{\sum_{k=1}^{K} \sum_{i \in \mathcal{X}_k} g_{il} w_k}. \tag{8}$$

For the categorical data, when $\mathcal{G}$, $\mathcal{P}$, and $\mathcal{W}$ are fixed, based on the loss function defined in Eq. (4), the group-level truth c_{lm} should be the value that receives the highest weighted votes among all possible values (mode value):

$$c_{lm} \leftarrow \arg\max_{v} \sum_{k=1}^{K} \sum_{i \in \mathcal{X}_k} g_{il} w_k \cdot h(v, v_{im}), \tag{9}$$

where $h(x, y) = 1$ if $x = y$, and 0 otherwise.

Step 4: Source weights update. After the updates of $\mathcal{G}$, $\mathcal{C}$, and $\mathcal{P}$, we update the weight of each source according to the differences between the values it provided and the group-level truths, where each attribute is weighted by $\mathcal{P}$. That is to say, when more consideration is given to the differences towards significant attributes, the source, which provides more correct values on the significant attributes, will be assigned to a higher weight. Therefore, by trusting the information provided by the sources with higher weights, we can more correctly achieve the group-level truths on significant attributes. At this step, $\mathcal{G}$, $\mathcal{C}$, and $\mathcal{P}$ are fixed. To minimize the objective function in Eq. (1) with the regularization constraint in Eq. (2), we derive the following equation using Lagrange multipliers.

$$w_k = -\log\left(\frac{\sum_{l=1}^{L} \sum_{o_i \in \mathcal{X}_k} g_{il} \sum_{m=1}^{M} p_m d_m(v_{im}, c_{lm})}{\sum_{l=1}^{L} \sum_{k'=1}^{K} \sum_{o_i \in \mathcal{X}_{k'}} g_{il} \sum_{m=1}^{M} p_m d_m(v_{im}, c_{lm})}\right). \tag{10}$$

The pseudo code of this framework is summarized in Algorithm 1. We start with initial estimates on the group indicators $\mathcal{G}$, the group-level representatives $\mathcal{C}$, and the source weights $\mathcal{W}$, then iteratively conduct the above four steps until convergence.

Algorithm 1: The PatternFinder algorithm

Input: A collection $\mathcal{D}$ of unaligned records, L latent groups, and the balance parameter α.
Output: The group indicators $\mathcal{G}$, the group-level representatives $\mathcal{C}$, the attribute weights $\mathcal{P}$, and the source weights $\mathcal{W}$.

```
Initialize G and C
Initialize W
repeat
    Update P according to Eq. (5)
    Update G according to Eq. (7)
    for each group C_l do
        for each attribute A_m do
            Update c_lm according to Eq. (8) and Eq. (9)
    Update W according to Eq. (10)
until the convergence criterion is satisfied
return G, C, P, and W.
```

In the following parts, we first discuss two important issues that make PatternFinder more practical including *initialization* and *convergence*. We then analyze the time complexity of PatternFinder.

Initialization. For the initialization of the source weights, we simply assign the same weight for each source. The initialization of the latent groups and the group-level representatives can be obtained using the existing clustering approaches. Clusters are treated as the latent groups, and the centers of the clusters are the group-level representatives. In our experiments, we find that the result from K − means is typically a good start. The attribute set is regarded as the feature matrix for each entity, and the number of clusters is set according to the elbow method [39], which is experimentally validated in Section 5.2.

Convergence. We prove the convergence of PatternFinder as follows.

Theorem 2. *PatternFinder algorithm converges.*

Proof. For the optimization problem in Eq. (1), it can be inferred that the unique minimum with respect to one set of variables is achieved when the other three sets of variables are fixed. Thus, for the four steps generated by PatternFinder, the objective value is minimal at each step. According to the proposition on the convergence of the block coordinate descent [37], the proposed iterative procedure will converge to a stationary point.

Time complexity. In each iteration, PatternFinder computes $\mathcal{G}$ for each record according to the current estimates of $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$. Thus, the running time is linear with $\mathcal{O}(LMn)$, where L is the number of latent groups, M is the number of attributes and n is the total number of records. In total, the time complexity of PatternFinder is $\mathcal{O}(rLMn)$, where r is the number of iterations. When L and M are fixed, the time complexity is linear with respect to n, which is experimentally validated in Section 5.2.

4.4. Optimized grouping strategy

In this section, we develop an optimized grouping strategy to improve the efficiency of PatternFinder. We first provide the observations to PatternFinder and then propose this strategy.

Observations. In Algorithm 1, according to Eq. (7), during the process of updating $\mathcal{G}$, we need to update the group indicators of each record $\boldsymbol{o}_i$ by calculating an item $S_{il'}$ $(1 \le l' \le L)$ for every group-level representative $\boldsymbol{c}_{l'}$. o_i can then be clustered to group C_l $(g_{il} = 1)$ which has the minimum value S_{il}. To simplify the discussion, we denote S_{il} as B_i. It can be inferred that, when L is large, the process will be too costly to scale well for a large amount of data. Therefore, we present a scalable strategy to reduce the calculations in this process by making use of the information achieved in the previous iterations.

Let j be the current iteration and $j-1, j-2$ be the previous iterations. For a record $\boldsymbol{o}_i$, $B_i^{(j)}$ is the product of $\mathcal{P}^{(j)}$ and the distance between $\boldsymbol{o}_i$ and $\boldsymbol{c}_l^{(j-1)}$. As $\mathcal{P}$ always slightly changes after the first several iterations, the change will not significantly influence the update of the group indicators. Therefore, we set a threshold ξ to evaluate the change $\Delta(\mathcal{P}^{(j)} - \mathcal{P}^{(j-1)})$. If $\Delta(\mathcal{P}^{(j)} - \mathcal{P}^{(j-1)}) < \xi$, we consider that the update of $\boldsymbol{c}_l$ is the only factor affecting the change from $B_i^{(j-1)}$ to $B_i^{(j)}$.

We then analyze the update of $\boldsymbol{c}_l$, which involves two cases. (1) The current distance between $\boldsymbol{o}_i$ and $\boldsymbol{c}_l^{(j-1)}$ is no greater than the previous distance between $\boldsymbol{o}_i$ and $\boldsymbol{c}_l^{(j-2)}$. It is obvious that if $\boldsymbol{o}_i$ is closer to $\boldsymbol{c}_l^{(j-1)}$, it will be far apart from the other group-level representatives. Therefore, $\boldsymbol{o}_i$ will stay in group l, and there is no need to calculate the distances between $\boldsymbol{o}_i$ and other group-level representatives. (2) The current distance between $\boldsymbol{o}_i$ and $\boldsymbol{c}_l^{(j-1)}$ is greater than the previous distance between $\boldsymbol{o}_i$ and $\boldsymbol{c}_l^{(j-2)}$. In this case, as $\boldsymbol{o}_i$ is far apart from $\boldsymbol{c}_l^{(j-1)}$, there may exist a group l' with a smaller distance. Thus, we need to calculate the distances between $\boldsymbol{o}_i$ and the other group-level representatives, and assign $\boldsymbol{o}_i$ to the nearest group by Eq. (7).

Based on the above analysis, the process of updating $\mathcal{G}$ is shown in Algorithm 2. It reduces the running time of updating

Algorithm 2: Optimized grouping strategy

Input: A collection $\mathcal{D}$ of unaligned records, the given parameter ξ, $\mathcal{B}^{(j-1)} = \{B_1^{(j-1)}, B_2^{(j-1)}, \cdots, B_n^{(j-1)}\}$, $\mathcal{G}^{(j-1)}$, $\mathcal{C}^{(j-1)}$, and $\mathcal{P}^{(j-1)}$ in the $(j-1)$-th iteration, and $\mathcal{P}^{(j)}$ in the (j)-th iteration.
Output: $\mathcal{B}^{(j)} = \{B_1^{(j)}, B_2^{(j)}, \cdots, B_n^{(j)}\}$, $\mathcal{G}^{(j)}$.

if $\Delta(\mathcal{P}^{(j)} - \mathcal{P}^{(j-1)}) < \xi$ **then**
  for each $\boldsymbol{o}_i \in \mathcal{D}$ **do**
    Compute $S_{il}^{(j)} \leftarrow \sum_{m=1}^{M} p_m^{(j)} d_m(v_{im}, c_{lm}^{(j-1)})$, where $l \leftarrow \{l | g_{il}^{(j-1)} = 1, 1 \le l \le L\}$
    if $S_{il}^{(j)} < B_i^{(j-1)}$ **then**
      $g_{il}^{(j)} \leftarrow 1$
      $B_i^{(j)} \leftarrow S_{il}^{(j)}$
      for each $1 \le l' \le L \cap l' \ne l$ **do**
        $g_{il'}^{(j)} \leftarrow 0$
    else
      Update the group indicator $g_{il}^{(j)}$ according to Eq. (7)
      $B_i^{(j)} \leftarrow S_{il}^{(j)}$, where $l \leftarrow \{l | g_{il'}^{(j)} = 1, 1 \le l' \le L\}$
else
  Update the group indicator $\mathcal{G}^{(j)}$ according to Eq. (7)
  Update $\mathcal{B}^{(j)}$
return $\mathcal{B}^{(j)}, \mathcal{G}^{(j)}$.

the group indicators with respect to each $\boldsymbol{o}_i$. In each iteration, if $\boldsymbol{o}_i$ stays in the current group, it requires $\mathcal{O}(1)$ time, otherwise, it requires $\mathcal{O}(L)$. Suppose that half of the records update their group indicators, this then requires $\mathcal{O}(MnL/2)$. Since PatternFinder converges to a stationary point, the number of records updating their group indicators decreases in each iteration. Thus, the total cost is at most $LMn\sum_{i=1}^{r} 1/r$, where r is the number of iterations. For a large number of iterations, $LMn\sum_{i=1}^{r} 1/r$ is much less than $rLMn$.

4.5. Pattern and truth generation

Given the output of PatternFinder, we discuss the pattern and truth generation approaches as follows.

Pattern generation. Recall that a pattern φ_l is a triple variable that contains an applied set R_l, an attribute set X and a value combination $\boldsymbol{t}_{lX}$. Given the output $\mathcal{G}$, $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{W}$ of PatternFinder, the pattern φ_l is generated as follows.

1. R_l corresponds to C_l achieved from $\mathcal{G}$;
2. X is made up of attributes A_m with a higher p_m, where the number of attributes in X can be specified by the users. In our experiments, we consider $A_m \in X$ when $p_m > \frac{1}{|M|}$, which implies that if p_m is larger than the average weight $\frac{1}{|M|}$, A_m is added to X;
3. $\boldsymbol{t}_{lX}$ corresponds to the value combination of $\boldsymbol{c}_l$ on X.

Pattern application. It is convenient to discover truths by patterns. Given $\mathcal{D}$ and a set of patterns, for each o_i in R_l, if $A_m \in X$, the truth v_{im}^* of v_{im} is t_{lm}. We then achieve the truth set $\mathcal{D}_v$ made up of the truths v_{im}^*, where $o_i \in \mathcal{D}$ and $A_m \in X$.

5. Experiments

In this section, we evaluate the proposed methods using both simulated and real-world datasets. The experimental results clearly demonstrate the advantages of the proposed methods in pattern discovery and truth discovery in terms of both effectiveness and efficiency. We first discuss the experimental setup in Section 5.1, and then present experimental results for the simulated and real-world datasets in Sections 5.2 and 5.3, respectively.

5.1. Experimental setup

5.1.1. Algorithms

For the proposed methods, we evaluate the basic version PatternFinder and the scalable version PatternFinder + OP with the optimized grouping strategy proposed in Section 4.4.

For the baseline methods, as our methods find the truths for multi-source unaligned data, the goal can also be achieved orderly or jointly by performing entity resolution (ER) and truth discovery (TD) methods. For the orderly baselines, we separately implement the following ER and TD approaches, and first record the performance of ER approaches, then the TD approaches.

ER baselines:

- Link: This is a naive entity resolution method. For each pair of records, it computes a value similarity for each attribute and takes the average. It links two records if the average similarity is more than the given threshold of 0.9 and then considers all linked records as representing one entity.
- R-Swoosh [40]: This approach uses a boolean pairwise match function to compare records and uses a pairwise merge function to merge two records that match into a composite record. It returns a partition of records by merging records which have a similarity of at least 0.9.
- Lego [41]: It formalizes an iterative model where blocks are processed in an iterative fashion until no block contains any matching records. We use R-Swoosh as the core entity resolution algorithm, which follows the same setting described in [41].
- Magellan [36]: It develops an open-world ER system, which relies on many other systems to provide the fullest amount of support to the ER user. As it is designed for ER between two sources, we run it several times to achieve the result of record linking over multiple sources.

TD baselines:

- Vote: We use voting as a baseline method. Voting chooses the values for each attribute for each entity that is provided by the largest number of sources as the final results.
- CRH [5]: This approach models the conflicts resolution problem for data of the heterogeneous types. It derives a two-step iterative procedure including the computation of truths and source weights as a solution to an optimization problem.

Joint baseline:

- Cluster [42]: It models the entity resolution problem as a k-partite graph clustering problem. By making each cluster including at most one single value for each attribute, the clustering process can jointly conduct entity resolution and truth discovery.[1]

All the experimental results are conducted using a Linux machine with 8G RAM and Inter Core i5 processor. We implemented all the methods including our methods and the baselines in Matlab. For PatternFinder and PatternFinder+OP, we ran them 10 times and report the average results.

[1] Due to its efficiency issues, we failed to achieve any result for both the simulated and real-world datasets in 48 h. Thus, we omit the performance of this method in the result presentation.

5.1.2. Performance measures

As the proposed methods study pattern discovery for truth discovery, we evaluate the accuracy of the pattern discovery as well as the truth discovery.

Pattern discovery accuracy. For each pattern, it is crucial to discover the correct value combination, as this value combination is used to match all the records in the applied set. Thus, we evaluate the accuracy of the value combinations of patterns as the accuracy of the pattern discovery. The accuracy is measured by **precision** and **recall**. We denote the set of value combinations existing in the dataset by G_p for the golden standard, and denote the set of value combinations of patterns by D_p for an approach. *Precision* is calculated by $precision = \frac{|G_p \cap D_p|}{|D_p|}$, denoting the proportion of the corrected value combinations to the number of all the value combinations found by the approach. *Recall* is calculated by $recall = \frac{|G_p \cap D_p|}{|G_p|}$, representing the proportion of the corrected value combinations to the number of all the value combinations existing in the dataset.

Truth discovery accuracy. To evaluate the performance of pattern application in Section 4.5, we also measure the accuracy of truth discovery. More specifically, the truths of the entities in the attribute set X can be directly obtained from the patterns. We denote the truth set by D_v for an approach which contains the truths in the attribute set X for each record, and G_v for the golden standard. We measure the truth discovery accuracy by $\textbf{ErrorRate} = 1 - \frac{|G_v \cap D_v|}{|D_v|}$, which denotes the percentage of the estimated truths achieved by the approach that are different from the ground truths.

5.2. Experiments using simulated datasets

To show the performance of the proposed methods and the baseline methods using the data of different characteristics, we first experiment using the simulated data generated from two real-world datasets.

Hospital.[2] Hospital is a real-word dataset used in [43,44]. It contains 105 K records with 6 categorical attributes[3]: Patient Number (PN), Hospital Name (HN), Condition, Measure Code (MC), Measure Name (MN), and Sample.

Bus.[4] Bus is a real-word dataset used in [43]. It contains 108K records with 7 categorical attributes: Bus Number (BN), Station Name (SN), Locality Code (LC), Locality Name (LN), Parent Locality Name (PN), Easting, and Northing.

5.2.1. Dirty data simulation

The original dataset is regarded as the ground truth, and we generate five conflicting sources by injecting different levels of noise into the ground truth. To simulate the real-world problem setting, for an entity, each source has a 50% probability to provide its information. For a given attribute value of an entity, we flip its value to another value based on the source's noise rate. After the simulation, a collection $\mathcal{D}$ of records is then formed, which contains the records from five sources with various degrees of the noise rate $\beta = \{0.1, 0.5, 0.9, 1.3, 1.7\}$. A lower β indicates a lower chance that the ground truth is altered to generate records.

[2] http://www.hospitalcompare.hhs.gov/.

[3] Note that we use categorical attributes in our experiments, as most of our baselines [40–42] can only be applied to categorical data. However, our methods can also deal with numerical attributes.

[4] http://data.gov.uk/data.

5.2.2. Effectiveness of parameters

In this part, we first test the performance of our methods subject to different settings of the parameters, then compare it with the baseline methods of different data sizes. Due to the efficiency issues of the baselines, we randomly select a part of the original records (min $2.5k$, max $20k$) to test them.

The effect of the number L of the latent groups. Integrating the tuning of L into the entire optimization framework may increase the computational complexity which leads to bad results. As the number L of the latent groups is relatively isolated from the rest of the variables in Eq. (1), we set L as a hyper input parameter of PatternFinder. Recall that we use the output of K-means as the initialization of PatternFinder, where the number L of the latent groups is set as the number of clusters for K-means. To choose the optimal number of clusters for K-means, we apply the elbow method [39]. That is, we run K-means for a range of L, and compute the sum of squared errors (SSE) for each L. SSE is computed as the sum of the squared distance between each record member of a group and its group-level representatives. Fig. 2 shows the SSE for each L on both datasets. It can be inferred that each line chart looks like an arm, and then we choose the "elbow" on the arm as the number of clusters (i.e., the number of the latent groups). For the hospital dataset, L is set as 30, while for the bus dataset, L is set as 200.

The effect of balance parameter α. As the accurate estimation for the attribute weights and the source weights is the key to obtain the correct patterns, we further show the attribute weights and the source weights of PatternFinder by various α values. The balance parameter α is set as $\{1k, 2k, 4k, 8k\}$. For each α, we report the result of the attribute weights, source weights, and the overall performance in Fig. 3.

From Fig. 3(a)(d), we can infer that with the decrease in α, the variance of $\mathcal{P}$ rapidly rises in both datasets. The experimental results can be explained from Eq. (5): As α decreases, only a few attributes play important roles in the process of pattern discovery. However, we can see that the weights of the significant attributes stand out in every situation: For the Hospital dataset, the significant attributes are Condition (ID 3), MC (ID 4), and MN (ID 5); For the Bus dataset, the significant attributes are LC (ID 3), LN (ID 4) and PN (ID 5). It can be inferred that PatternFinder is not too sensitive to the setting of parameter α. As long as the pattern exists among the attributes, PatternFinder is able to discover it.

We also study the distribution of the source weights according to different settings of α, which are shown in Fig. 3(b)(e). The x-axis is the noise rate $\beta = \{0.1, 0.5, 0.9, 1.3, 1.7\}$ set for each source and the y-axis is the source weights $\mathcal{W}$ achieved by PatternFinder. We use a logscale to perform $\mathcal{W}$ and calculate the Pearson correlation coefficient ρ between β and $\log \mathcal{W}$. We can see that the Pearson correlation coefficient ρ tends to -1 for different α values, which implies that $\mathcal{W}$ estimated by PatternFinder is precise enough to remain consistent with the sources' noise rate we generally set.

Finally, Fig. 3(c) and (f) show the precision, recall, and ErrorRate of PatternFinder for different settings of α. We can see that PatternFinder performs the best when $\alpha = 2k$ for the Hospital dataset and $\alpha = 4k$ for the Bus dataset. The reason is that in these settings, ρ reaches -0.9902 and -0.9850, respectively. The experimental results indicate that our method can successfully distinguish good sources from bad ones, and accordingly, derive the correct patterns based on the good sources.

The effect of threshold ξ. To show the performance of the scalable version PatternFinder + OP, we varied the threshold ξ. We set the balance parameter α as $2k$ and $4k$ on the two datasets, and varied the threshold ξ from 0 to 0.2 by a step of 0.05. The recall, precision, ErrorRate, and runtime compared for the

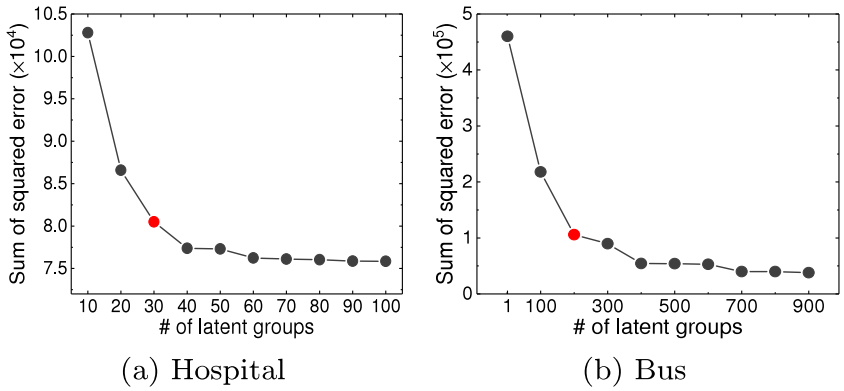

Fig. 2. *SSE* w.r.t. the number L of the latent groups.

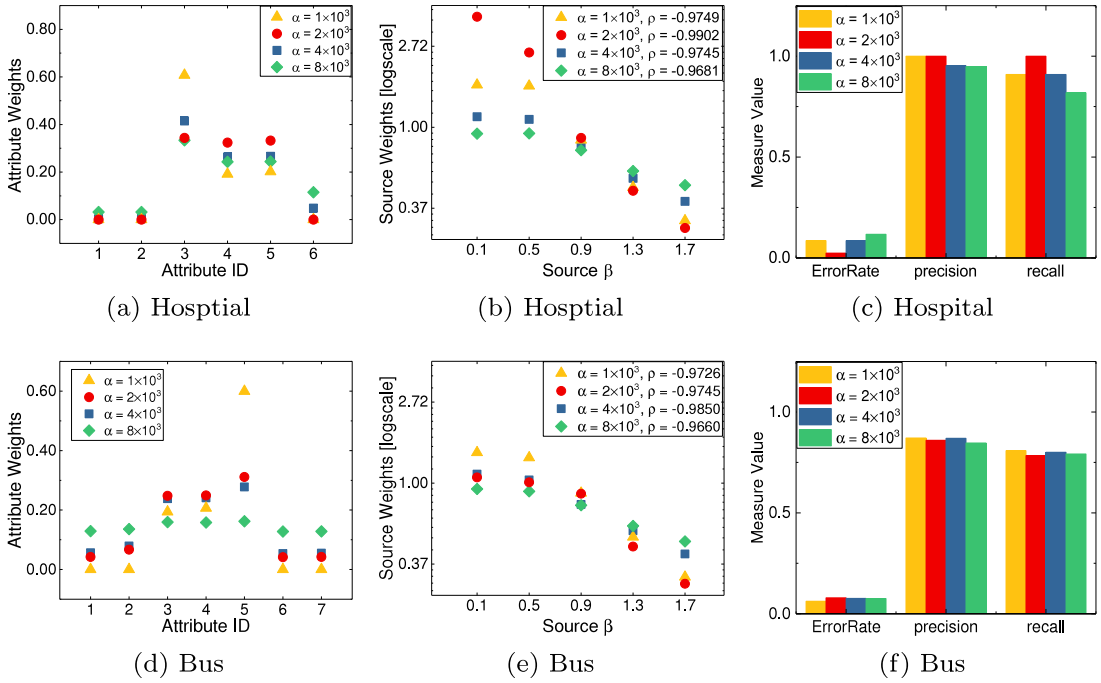

Fig. 3. Performance w.r.t. threshold α.

basic version PatternFinder are shown in Fig. 4. From Fig. 4(a)(c), we can see that with the increase in threshold ξ, the ErrorRate slightly rises for both datasets. On the other hand, the recall and precision slightly decrease. However, the rise of the ErrorRate and drop of the precision and recall are acceptable in contrast for the improvement in the efficiency. As shown in Fig. 4(b)(d), the optimization strategy leads to a runtime improvement up to 35% and 33% for the two datasets, respectively. The reason is that we reduce the calculations of updating $\mathcal{G}$ by making use of the information in the previous iterations.

The effect of data sizes. We varied the data sizes from 2.5k to 20k to test the performance of our methods and the baselines, and the results are shown in Fig. 5. For PatternFinder, we set α as 2k and 4k on the two datasets, respectively. Note that we omit the performance of PatternFinder + OP for its similar performance to PatternFinder, which can be inferred from Fig. 4. Based on Fig. 5(a)(d), we can see that PatternFinder achieves a significantly lower ErrorRate than the baselines. This is because the accurate estimation of the source weights gives a better guidance to find the truths for the attributes of the patterns. Moreover, with the increase in the data sizes, the ErrorRate of PatternFinder decreases while that of the baselines methods rises. The reason is that PatternFinder finds the correct patterns in a global view. Thus, a larger data size results in a lower ErrorRate. Magellan+CRH and Magellan+CRH have relatively low ErrorRates compared for other baseline methods. This is because Magellan selects the best learning-based matcher by a cross validation strategy. Lego + CRH and Lego + Vote perform better than R − Swoosh + CRH and R − Swoosh + Vote. This is because Lego processes in an iterative fashion, and it can generate new record matches based on the blocking results produced by R-Swoosh. However, the result of Lego is still not good enough, resulting in the bad performance of CRH when compared for Vote. Link+CRH and Link+Vote also perform not well enough, as Link fails to accurately match most of the entities by simply linking records according to their similarity.

In terms of efficiency, the result is shown in Fig. 5(b)(e). We can see that PatternFinder achieves a great improvement compared for the baseline methods. This is because PatternFinder treats the pattern discovery problem as a high-level clustering problem and aims to find the common patterns existing in the data rather than conducting both the entity resolution and truth

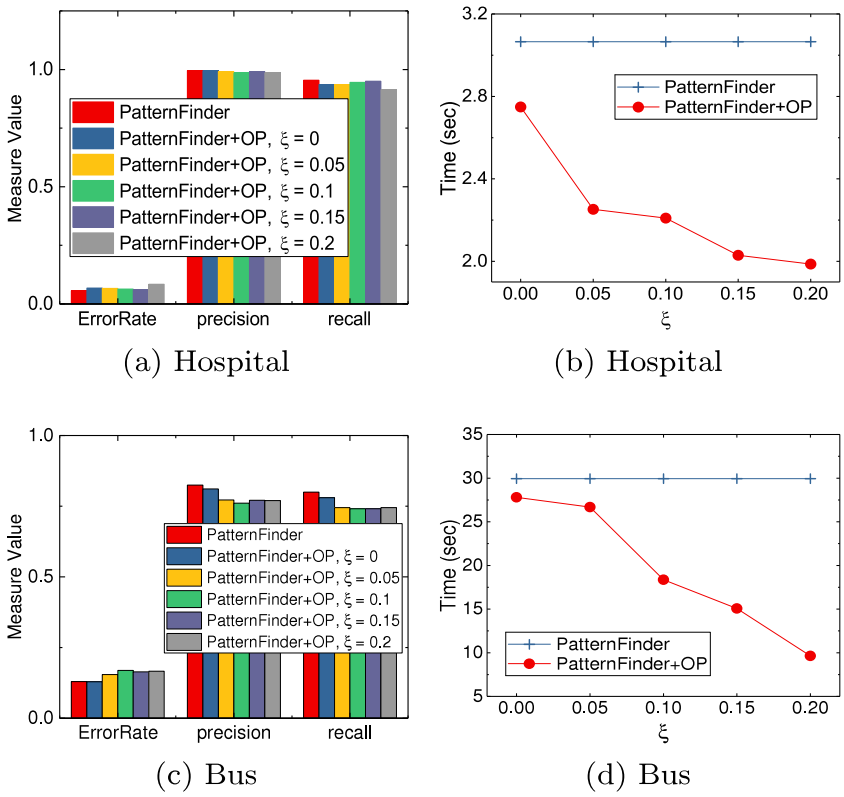

Fig. 4. Performance w.r.t. threshold ξ.

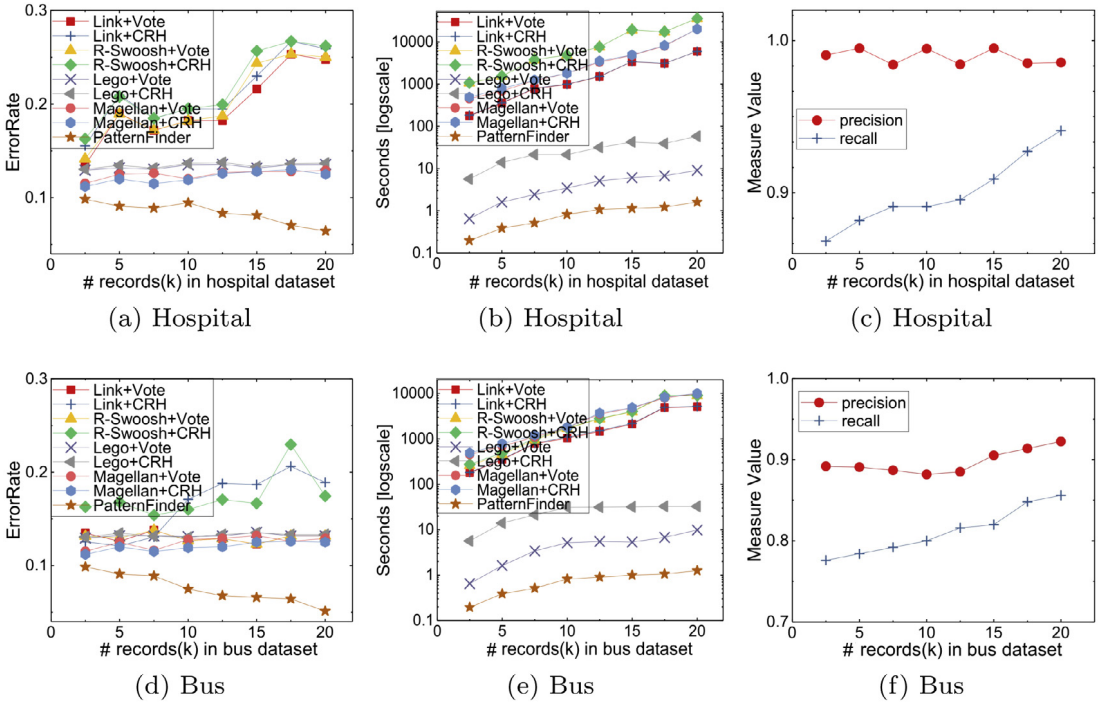

Fig. 5. Performance w.r.t. data sizes.

discovery steps. Among the baseline methods, Lego + Vote performs the best. The reason is that Lego converges fast by making use of the result of R-Swoosh. For CRH, it needs more time to iteratively estimate the source reliability, which makes Lego + CRH run slower than Lego + Vote.

Additionally, the precision and the recall of PatternFinder are shown in Fig. 5(c)(f). We can see that with the increase in the data sizes, the precision remains stable and the recall significantly rises. As PatternFinder can discover the patterns based on the grouping of similar records, it can find more patterns with an

Table 3
Effectiveness comparison on real-world datasets.

Methods	ErrorRate	
	Restaurant	Flight
Link+Vote	0.2397	0.6661
Link+CRH	0.2034	0.6717
R-Swoosh+Vote	0.2502	0.6659
R-Swoosh+CRH	0.2139	0.6717
Lego+Vote	0.2490	0.3561
Lego+CRH	0.2119	0.3574
Magellan+Vote	0.2123	0.2392
Magellan+CRH	0.1915	0.2104
PatternFinder	**0.1362**	**0.1298**

Table 4
Efficiency comparison on real-world datasets.

Methods	Runtime (s)	
	Restaurant	Flight
Link+Vote	11542.83	40759.88
Link+CRH	11585.62	40780.50
R-Swoosh+Vote	21259.04	105136.71
R-Swoosh+CRH	21301.03	105166.56
Lego+Vote	40580.64	57183.65
Lego+CRH	40624.87	57206.08
Magellan+Vote	50383.56	171908.09
Magellan+CRH	50451.37	171935.98
PatternFinder	**151.63**	**9.2329**

increase in the relevant data size. With a stable precision and higher recall, the ErrorRate tends to reasonably decrease, which also matches the ErrorRate of PatternFinder in Fig. 5(a)(d).

5.3. Experiments using real-world datasets

Restaurant dataset: We collected the restaurants' information located in NYC from NYC Open Data,[5] Yelp,[6] YellowPage,[7] NYC health,[8] and SuperTour[9] and obtained a collection of a total of 20,546 records. Note that the records provided by different sources may refer to the same restaurant, but we do not know the linkage beforehand. The restaurant dataset has the following attributes: Name, Street, Building, Zip, and Phone. In order to test the accuracy of the proposed methods and the baseline methods, we randomly selected 1281 restaurants and manually labeled their information, which contain 272 different value combinations of street and zip. We looked up the official websites of these restaurants and regarded the information on these websites as the gold standard. In our experiments, we set $L = 3k$ and $\alpha = 2k$. Under this setting, the attributes Street and Zip form the pattern.[10]

Flight dataset: The flight data [1], collected over a one-month period starting from December 2011, consists of 1200 flights and 6 attributes from 38 sources. We conducted pre-processing on the data to convert the gate information into the same format and the time information into minutes. Note that we consider the flight information provided by different sources as different entities, which is a different task setting compared for that in [1,5]. The ground truth of 100 flights is also available from a total of 51,114 records. In order to compare the results with the baseline methods, which take a long time to produce, we reduced the size of the original dataset by only using the 51,114 records which have ground truth as our test dataset. In the experiments, we set $L = 100$ and $\alpha = 8k$. Under this setting, the attributes FN (Flight number), SDT (Scheduled departure time) and SAT (Scheduled arrival time) make up the pattern.[10]

5.3.1. Effectiveness evaluation

Table 3 summarizes the ErrorRate for all the methods of the two real-world datasets. We can see that the proposed method achieves the best performance on every dataset, and the improvement is promising. For the Restaurant dataset, compared for the best baseline Link+CRH, the proposed method's ErrorRate decreases by 6.72% and for the Flight dataset, compared for the best baseline Lego+Vote, the proposed method's ErrorRate decreases by 22.6%. All the baseline methods first conduct the entity resolution step, then make use of different similarity functions to block the records. Due to large number of errors existing in every attribute, these methods fail to divide the records referred to the same entity into the same block. Thus, they all poorly perform. With such bad entity resolution results, the advantage of truth discovery (CRH) cannot be seen (Restaurant) and even performs worse (Flight).

We also validated PatternFinder+OP with different settings of the threshold ξ for both real-world datasets, which is shown in Fig. 6. From Fig. 6(a)(c) we can see that, with the increase in ξ, the ErrorRate slightly rises to 0.01%, and the precision and recall drop slightly to 0.03% for both real-world datasets. In contrast, the runtime has a significant improvement compared for PatternFinder, which is shown in Fig. 6(b)(d). For the Restaurant dataset, the runtime decreases to 34% and for the Flight dataset, the runtime decreases to 30%.

5.3.2. Efficiency evaluation

In this section, we evaluated the efficiency of PatternFinder. We first explored its convergence, then showed its runtime as well as the baselines.

Convergence speed. As PatternFinder uses an iterate process to discover patterns, we first test its convergence for both real-world datasets. Fig. 7 shows the change in the objective value with respect to each iteration. We can see that the objective value decreases fast for the first five iterations, then reaches a stable stage. This is because the proposed optimization problem Eq. (1) is biconvex, thus we need to alternatively optimize each variable. Due to the large gradients in the first five iterations, the variables dramatically change, resulting in fast decrease of the objective value. The experimental result indicates that PatternFinder quickly converges in practice.

Runtime. Table 4 summarizes the runtime for all the methods using both real-world datasets. For the Restaurant dataset, PatternFinder is significantly two orders of magnitude faster than the other baselines. For the Flight dataset, PatternFinder achieves four orders of magnitude faster time than the other baselines. Magellan runs the slowest, as it needs time for reading the documentation and labeling samples when matching. Lego and R-Swoosh are also time-consuming, both of which require the process of blocking and matching. Lego performs relatively faster than R-Swoosh, if the runtime saved by using the blocking result of R-Swoosh is greater than the additional time needed to iteratively process the blocks (Flight), while Lego performs worse than R-Swoosh if it is not the above case (Restaurant). Link requires less runtime, as it only uses a similarity score to decide whether two records represent one entity. For the truth discovery baselines, as CRH needs some iterations to converge, the methods conducted with CRH need more time than the methods conducted with Vote for both real-world datasets.

[5] https://opendata.cityofnewyork.us/.
[6] https://www.yelp.com/nyc.
[7] https://www.yellowpages.com/.
[8] http://www.nychealthratings.com/.
[9] http://test.supertour.com/newyork-ny.us.aspx.
[10] We omit the performance of PatternFinder for the different settings of L and α, which is similar to the results shown in Section 5.2.2 (Fig. 2 and Fig. 3).

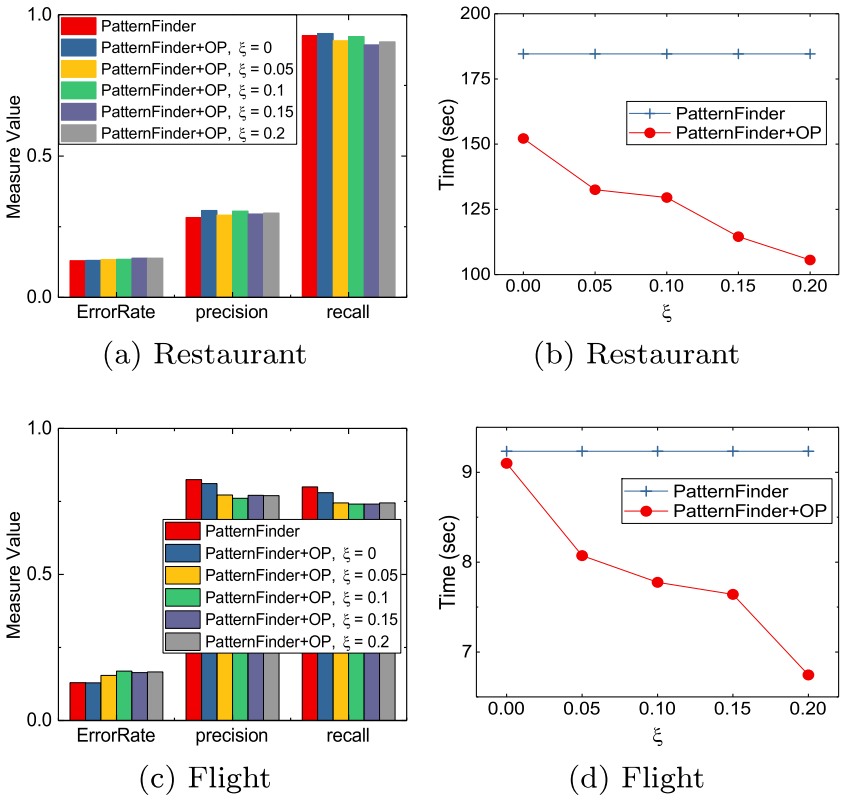

Fig. 6. Performance w.r.t. Threshold ξ.

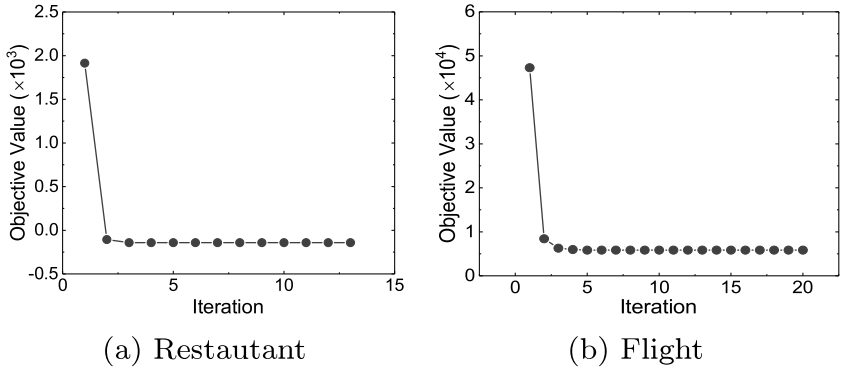

Fig. 7. Convergence speed.

6. Conclusion

In this paper, we introduce pattern discovery for truth discovery of multi-source unaligned data. We model this pattern discovery problem as a task of inferring latent groups using a general optimization framework. In this model, the objective is to minimize the overall weighted deviation between the group-level representatives and the multi-group records where each source is weighted by its reliability and each attribute is weighted by its significance. We developed a four-step iterative algorithm called PatternFinder to solve the optimization problem. To improve its efficiency, we also proposed a scalable version by an optimized grouping strategy. We conducted experiments using both real-word and simulated datasets. The results demonstrated the efficiency and the effectiveness of PatternFinder. In this study, the source weights and group indicators output by PatternFinder are meaningful to some other applications. In the future, we plan to adapt the framework to more application scenarios, such as the analysis of information trustworthiness.

Acknowledgments

This paper was partially supported by NSF IIS-1747614, IIS-1553411, NSFC grant U1509216, U1866602, 61602129 and MOE-Microsoft Key Laboratory of Natural Language Processing and Speech, Harbin Institute of Technology.

References

[1] X. Li, X.L. Dong, K. Lyons, W. Meng, D. Srivastava, Truth finding on the deep web: Is the problem solved?, PVLDB 6 (2) (2013) 97–108.
[2] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, J. Han, A survey on truth discovery, in: Proc. of SIGKDD, vol. 17(2), 2015 pp. 1–16.
[3] X.L. Dong, L. Berti-Equille, D. Srivastava, Truth discovery and copying detection in a dynamic world, PVLDB 2 (1) (2009) 562–573.
[4] Q. Li, Y. Li, J. Gao, L. Su, B. Zhao, W. Fan, W. Fan, J. Han, A confidence-aware approach for truth discovery on long-tail data, PVLDB 8 (4) (2014) 425–436.
[5] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, J. Han, Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation, in: Proc. of SIGMOD, 2014, pp. 1187–1198.

[6] R. Pochampally, A. Das Sarma, X.L. Dong, A. Meliou, D. Srivastava, Fusing data with correlations, in: Proc. of SIGMOD, 2014, pp. 433–444.
[7] H. Xiao, J. Gao, Q. Li, F. Ma, L. Su, Y. Feng, A. Zhang, Towards confidence in the truth: A bootstrapping based truth discovery approach, in: Proc. of SIGKDD, 2016, pp. 1935–1944.
[8] X. Yin, J. Han, S.Y. Philip, Truth discovery with multiple conflicting information providers on the web, IEEE Trans. Knowl. Data Eng. 20 (6) (2008) 796–808.
[9] X.S. Fang, Q.Z. Sheng, X. Wang, A.H.H. Ngu, Value veracity estimation for multi-truth objects via a graph-based approach, in: International Conference on World Wide Web Companion, 2017, pp. 777–778.
[10] S.E. Whang, D. Marmaros, H. Garcia-Molina, Pay-as-you-go entity resolution, IEEE Trans. Knowl. Data Eng. 25 (5) (2013) 1111–1124.
[11] P. Christen, A survey of indexing techniques for scalable record linkage and deduplication, IEEE Trans. Knowl. Data Eng. 24 (9) (2012) 1537–1555.
[12] A.K. Elmagarmid, P.G. Ipeirotis, V.S. Verykios, Duplicate record detection: A survey, IEEE Trans. Knowl. Data Eng. 19 (1) (2007) 1–16.
[13] H. Köpcke, A. Thor, E. Rahm, Evaluation of entity resolution approaches on real-world match problems, PVLDB 3 (1–2) (2010) 484–493.
[14] N. Koudas, S. Sarawagi, D. Srivastava, Record linkage:similarity measures and algorithms, in: Proc. of SIGMOD, 2006, pp. 802–803.
[15] C. Ye, H. Wang, J. Li, H. Gao, S. Cheng, Crowdsourcing-enhanced missing values imputation based on Bayesian network, in: Proc. of DASFAA, 2016, pp. 67–81.
[16] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, J. Han, On the discovery of evolving truth, in: Proc. of SIGKDD, 2015, pp. 675–684.
[17] X. Wang, Q.Z. Sheng, L. Yao, X. Li, X.S. Fang, X. Xu, B. Benatallah, Truth discovery via exploiting implications from multi-source data, in: Proc. of CIKM, 2016, pp. 861–870.
[18] H. Zhang, Q. Li, F. Ma, H. Xiao, Y. Li, J. Gao, L. Su, Influence-aware truth discovery, in: Proc. of CIKM, 2016, pp. 851–860.
[19] B. Zhao, B.I. Rubinstein, J. Gemmell, J. Han, A bayesian approach to discovering truth from conflicting sources for data integration, PVLDB 5 (6) (2012) 550–561.
[20] D. Wang, L.M. Kaplan, H.K. Le, T.F. Abdelzaher, On truth discovery in social sensing: a maximum likelihood estimation approach, in: Proc. of IPSN, 2012, pp. 233–244.
[21] S. Mukherjee, G. Weikum, C. Danescu-Niculescu-Mizil, People on drugs: credibility of user statements in health communities, in: Proc. of KDD, 2014, pp. 65–74.
[22] L. Su, Q. Li, S. Hu, S. Wang, J. Gao, H. Liu, T.F. Abdelzaher, J. Han, X. Liu, Y. Gao, L.M. Kaplan, Generalized decision aggregation in distributed sensing systems, in: Proc. of RTSS, 2014, pp. 1–10.
[23] V. Beretta, S. Harispe, S. Ranwez, I. Mougenot, Truth selection for truth discovery models exploiting ordering relationship among values, Knowl.-Based Syst. 159 (2018) 298–308.
[24] J. Pasternack, D. Roth, Making better informed trust decisions with generalized fact-finding, in: Proc. of IJCAI, 2011, pp. 2324–2329.
[25] T. Rekatsinas, M. Joglekar, H. Garcia-Molina, A. Parameswaran, C. Ré, Slimfast: Guaranteed results for data fusion and source reliability, in: Proc. of SIGMOD, 2017, pp. 1399–1414.
[26] Y. Yang, Q. Bai, Q. Liu, A probabilistic model for truth discovery with object correlations, Knowl.-Based Syst. 165 (2019) 360–373.
[27] C. Ye, Q. Li, H. Zhang, H. Wang, J. Gao, J. Li, Autorepair: an automatic repairing approach over multi-source data, Knowl. Inf. Syst. (2018) 1–31.
[28] H. Zhang, Y. Li, F. Ma, J. Gao, L. Su, Texttruth: An unsupervised approach to discover trustworthy information from multi-sourced text data, in: Proc. of SIGKDD, 2018, pp. 2729–2737.
[29] B. Zhao, J. Han, A probabilistic model for estimating real-valued truth from conflicting sources, in: Proc. of QDB, 2012.
[30] G. Cleuziou, M. Exbrayat, L. Martin, J.-H. Sublemontier, Cofkm: A centralized method for multiple-view clustering, in: Proc. of ICDM, 2009, pp. 752–757.
[31] A. Kumar, H. Daumé, A co-training approach for multi-view spectral clustering, in: Proc. of ICML, 2011, pp. 393–400.
[32] A. Kumar, P. Rai, H. Daume, Co-regularized multi-view spectral clustering, in: Proc. of NIPS, 2011, pp. 1413–1421.
[33] J. Liu, C. Wang, J. Gao, J. Han, Multi-view clustering via joint nonnegative matrix factorization, in: Proc. of SIAM, 2013, pp. 252–260.
[34] C. Xu, D. Tao, C. Xu, Multi-view intact space learning, IEEE Trans. Pattern Anal. Mach. Intell. 37 (12) (2015) 2531–2544.
[35] G.-Y. Zhang, C.-D. Wang, D. Huang, W.-S. Zheng, Y.-R. Zhou, Tw-co-k-means: Two-level weighted collaborative k-means for multi-view clustering, Knowl.-Based Syst. 150 (2018) 127–138.
[36] P. Konda, S. Das, P. Suganthan GC, A. Doan, A. Ardalan, J.R. Ballard, H. Li, F. Panahi, H. Zhang, J. Naughton, et al., Magellan: Toward building entity matching management systems, PVLDB 9 (12) (2016) 1197–1208.
[37] Y. Xu, W. Yin, A block coordinate descent method for regularized multi-convex optimization with applications to nonnegative tensor factorization and completion, SIAM J. Imaging Sci. 6 (3) (2013) 1758–1789.
[38] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, J. Cybern. 3 (3) (1973) 32–57.
[39] D.J. Ketchen, C.L. Shook, The application of cluster analysis in strategic management research: an analysis and critique, Strateg. Manage. J. 17 (6) (1996) 441–458.
[40] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S.E. Whang, J. Widom, Swoosh: a generic approach to entity resolution, VLDB J. 18 (1) (2009) 255–276.
[41] S.E. Whang, D. Menestrina, G. Koutrika, M. Theobald, H. Garcia-Molina, Entity resolution with iterative blocking, in: Proc. of SIGMOD, 2009, pp. 219–232.
[42] S. Guo, X.L. Dong, D. Srivastava, R. Zajac, Record linkage with uniqueness constraints and erroneous values, PVLDB 3 (1–2) (2010) 417–428.
[43] M. Dallachiesa, A. Ebaid, A. Eldawy, A. Elmagarmid, I.F. Ilyas, M. Ouzzani, N. Tang, Nadeef: a commodity data cleaning system, in: Proc. of SIGMOD, 2013, pp. 541–552.
[44] F. Geerts, G. Mecca, P. Papotti, D. Santoro, The llunatic data-cleaning framework, PVLDB 6 (9) (2013) 625–636.