Contents lists available at ScienceDirect

Information Systems

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

Reasoning about attribute value equivalence in relational data

Fengfeng Fan[a,*], Zhanhuai Li[a], Qun Chen[a], Lei Chen[b]

[a] School of Computer Science and Engineering, Northwestern Polytechnical University, 27 West Youyi Road Xian Shaanxi, PR China
[b] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 28 August 2017
Revised 22 December 2017
Accepted 16 February 2018
Available online 21 February 2018

Keywords:
Entity resolution
Attribute value matching
Relational data
Value Correlation Analysis

ABSTRACT

In relational data, identifying the distinct attribute values that refer to the same real-world entities is an essential task for many data cleaning and mining applications (e.g., duplicate record detection and functional dependency mining). The state-of-the-art approaches for attribute value matching are mainly based on string similarity among attribute values. However, these approaches may not perform well in the cases where the specified string similarity metric is not a reliable indicator for attribute value equivalence. To alleviate such limitations, we propose a new framework for attribute value matching in relational data. Firstly, we propose a novel probabilistic approach to reason about attribute value equivalence by value correlation analysis. We also propose effective methods for probabilistic equivalence reasoning with multiple attributes. Next, we present a unified framework, which incorporates both string similarity measurement and value correlation analysis by evidential reasoning. Finally, we demonstrate the effectiveness of our framework empirically on real-world datasets. Through extensive experiments, we show that our framework outperforms the string-based approaches by considerable margins on matching accuracy and achieves the desired efficiency.

1. Introduction

The same real-world entities may have different representations within or across relational databases. Variations in representation can arise from differences in storage formats, typographical errors, aliases and abbreviations. Determining attribute value equivalence is an essential task for many relational data cleaning and mining applications [1,2]. For instance, most techniques for duplicate record detection in relational data [3,4] divide each record into fields (attributes) and identify duplicate records by comparing their values on fields. Effective attribute value matching can therefore improve the accuracy of duplicate record identification. Functional dependency and conditional functional dependency mining [5,6] also requires attribute value matching to reduce noise: non-identical but equivalent attribute values could make a valid functional dependency elusive.

As pointed out by the surveys [7,8], most existing work on attribute value matching focused on reasoning about the equivalence between string data. The state-of-the-art techniques are based on measuring string similarity. A wide variety of metrics [9–12] have been proposed for this purpose. In comparison, the methods for capturing similarity in numeric data are rather primitive. Typically, the similar numbers are located by simple range queries, or treated as strings, which are then compared using string similarity metrics. Therefore, effective matching usually requires a metric to accommodate the value representation variations specific to a domain. Even though the existing string similarity metrics have been shown to be effective in various applications, they also have the fundamental limitation: a metric tuned and tested on previous problems can perform poorly on a new problem. Even though researchers have proposed adaptive algorithms [13,14] that can learn similarity metrics automatically, the difficulty of using these methods cannot be overlooked: they require significant training data and intensive human intervention. Provided with a new problem, it remains challenging to design both string similarity metric and threshold that can effectively capture the value representation variations present in the problem.

We illustrate the limitation of the string-based approach by the example as shown in Table 1. The relational records refer to research papers and each paper has four attributes, `title`, `author`, `journal`, `year`, which describe the title, authors, publication venue and publication year of the paper respectively. It can be observed that the `journal` values “Computers” and “Computer” look very similar but actually represent different publication venues. In contrast, the `journal` values “Journal on Very Large Data Bases” and “VLDB J” appear much less similar but actually refer to the same research journal. To alleviate the limitation of the string-based approach, we propose to reason about attribute value equiv-

* Corresponding author.
E-mail address: fanfengfeng@mail.nwpu.edu.cn (F. Fan).

https://doi.org/10.1016/j.is.2018.02.008
0306-4379/© 2018 Elsevier Ltd. All rights reserved.

Table 1
A relational table on research publications.

title	author	journal	year
Energy management in industrial plants	D. Bruneo, A. Cucinotta, A.L. Minnolo, A. Puliafito, M. Scarpa	Computers	2012
Beyond bits: the future of quantum information processing	A.M. Steane, E.G. Rieffel	Computer	2000
Priority assignment in real-time active databases	R.M. Sivasankaran, J.A. Stankovic, D. Towsley, B. Purimetla, K. Ramamritham	Journal on Very Large Data Bases	2003
A taxonomy of correctness criteria in database applications	K. Ramamritham, P.K. Chrysanthis	VLDB J	2002

alence by value correlation analysis. Note that in Table 1, the two `title` values, which are correlated with “Journal on Very Large Data Bases” and “VLDB J” respectively, have a common keyword “database”, and similarly, the `author` values correlated with them share a common author “K. Ramamritham”. Generally, we observe that the papers published in the same journal have a higher probability to be in the same research area than those published in different journals. Accordingly, they usually share some author names and their titles share some common keywords with higher probabilities. As a result, correlation analysis between the `journal` values and their corresponding `author` and `title` values can provide with useful clues for equivalence reasoning. More specifically, if two `journal` values are correlated with many common `author` values and many highly similar `title` values, it can be reasoned that they refer to the same journal entity with a high probability.

Note that a simple type of correlation among attribute values can be described by *functional dependency*, which specifies that the value of one attribute uniquely determines the value at another attribute. Obviously, a functional dependency can be exploited to match two attribute values. In the example shown in Table 1, suppose that each paper has a unique title. Accordingly, we have the functional dependency,

$$\mathtt{fd}_1 : \mathtt{title} \rightarrow \mathtt{journal} \tag{1}$$

As a result, two attribute values on `journal` can be determined to be equivalent if their corresponding records have the same value at the attribute `title`. Unfortunately, in practice, it is challenging to detect a clear-cut functional dependency in the presence of non-identical but equivalent attribute values, and even if it can be successfully detected, it may be of limited use in determining equivalence due to lack of matching data. Again, in the example shown in Table 1, if each record refers to a unique paper, the functional dependency, $\mathtt{fd}_1$, is then powerless in determining attribute value equivalence at `journal` because there do not exist two papers sharing a common title.

As illustrated by the motivating example, besides string similarity measurement, value correlation analysis can also be useful in reasoning about attribute value equivalence in relational data. Therefore, in this paper, we aim for a formal probabilistic model for value correlation analysis, and also a unified framework that can incorporate both string similarity measurement and value correlation analysis. Our major contributions can be summarized as follows:

1. We present a novel probabilistic approach to estimate the probability of attribute value equivalence by value correlation analysis, which reasons about the equivalence between two attribute values by analysing their correlation with other attribute values.
2. We propose a unified framework for attribute value matching in relational data. Based on both string similarity measurement and value correlation analysis, it provides a unified equivalence estimation by evidential reasoning. The proposed framework is a unified one in the sense that it can be simplified into a pure string similarity metric by setting the evidence weight of value correlation analysis to be 0.
3. We experimentally evaluate the performance of the proposed framework on real-world publicly-available datasets. Our extensive experiments show the effectiveness of the unified framework, demonstrating that it outperforms the string-based approaches by considerable margins on matching accuracy and achieves the desired efficiency.

Table 2
Summary of symbols.

Denotation	Description
R	A relational table.
$r \in R$	A tuple from R.
X, Y	Correlated and target attribute in R.
$\mathbf{X}$, $\mathbf{Y}$	Domains of attribute X and Y in R.
$X[R]$	Attribute values, $\{r[X] \mid r \in R\}$, at X in R.
$Y[R]$	Attribute values, $\{r[Y] \mid r \in R\}$, at Y in R.
$x_i \in \mathbf{X}$ ($y_i \in \mathbf{Y}$)	Values at attribute X (Y).
$R[x_i]$	Tuples $\{r \mid r[X] = x_i\}$ in R who have values x_i at attribute X.
$X[y_j]$	Attribute values $\{r[X] \mid r[Y] = y_j\}$, at X in $R[y_j]$.
$Y[x_i]$	Attribute values $\{r[Y] \mid r[X] = x_i\}$, at Y in $R[x_i]$.

2. Problem statement

For simplicity, those notations and corresponding meanings used in this paper are shown in Table 2.

Note that in R, each non-null value y_i at target attribute Y refers to a real-world entities. Two attribute values are deemed to be *equivalent*, denoted by $y_i \simeq y_j$, if and only if they refer to the same real-world entity. For instance, in Table 1, the values "VLDB J" and "Journal on Very Large Data Bases" are equivalent because they refer to the same publication venue.

The problem of attribute value matching in relational data is to find the equivalences existing between values at target attribute. It can be formally defined as follows:

Definition 1. Given two domains of distinct relations, $\mathbf{Y}_1 \in R_1$ and $\mathbf{Y}_2 \in R_2$ and pairs of attribute values, $\langle y_i, y_j \rangle \in \mathbf{Y}_1 \times \mathbf{Y}_2$, the problem of attribute value matching is to identify all the pairs $\langle y_i, y_j \rangle$ that referring to the same real-world entity.

Above definition has already subsumed the special case of finding pairs of equivalent attribute values within a single relation ($Y_1 = Y_2 \wedge R_1 = R_2$).

Attribute value matching is usually performed by pairwise comparisons. It ranks the pairs of attribute values by their equivalence probabilities. Therefore, the core challenge is to compute the equivalence probability between two given values at target attribute.

3. Proposed framework

In this section, we present the concept of *Value Correlation Analysis* (VCA), and then introduce the unified framework to incorporate the String Similarity Measurement (SSM) and Value Correlation Analysis (VCA).

3.1. Value correlation Analysis

We first present a basic model in Section 3.1.1 which exploits only identical values and reasons about value equivalence on a target attribute based on another attribute. Next, in Section 3.1.2, we propose an extended model that incorporates string similarity metrics into the estimation of equivalence reasoning. Finally, in Section 3.1.3, we discuss how to reason about value equivalence on a target attribute by multiple correlated attributes.

We first present a basic model in Section 3.1.1 which exploits only identical values and reasons about value equivalence on a target attribute based on another attribute. Next, in Section 3.1.2, we propose an extended model that incorporates string similarity metrics into the process of equivalence reasoning. Finally, in Section 3.1.3, we discuss how to reason about value equivalence on a target attribute based on multiple attributes.

3.1.1. Basic model

Given two non-identical values y_1 and y_2 at the target attribute Y, the basic model uses their correlated values at another attribute X to estimate their equivalence probability. As shown in Table 2, $X[y_i]$ denotes the attribute values at X for the tuples in $R[y_i]$. Therefore, the attribute values (e.g. $X[y_i]$) may contain duplicate attribute values.

Based on the Bayes' theorem, the probability of y_1 and y_2 being equivalent can be computed by

$$P(y_1 \simeq y_2) = \frac{P(y_1 \simeq y_2 | x_1 \simeq x_2)}{P(x_1 \simeq x_2 | y_1 \simeq y_2)} \cdot P(x_1 \simeq x_2) \tag{2}$$

in which $y_1, y_2 \in \mathbf{Y}$, $P(x_1 \simeq x_2)$ represents the probability that a value in $X[y_1]$ is equivalent to a value in $X[y_2]$, and $P(y_1 \simeq y_2 | x_1 \simeq x_2)$ and $P(x_1 \simeq x_2 | y_1 \simeq y_2)$ are two conditional probabilities representing the correlation between two attributes X and Y with respect to y_1 and y_2, respectively.

For reasoning about value equivalence between y_1 and y_2, the two conditional probabilities at the right-hand side of Eq. (2) cannot be computed directly from data. Therefore, we approximate them by general correlation analysis, which considers all the attribute values at Y and X existing in R, as follows:

$$\frac{P(y_1 \simeq y_2 | x_1 \simeq x_2)}{P(x_1 \simeq x_2 | y_1 \simeq y_2)} \approx \frac{P(y_i \simeq y_j | x_i \simeq x_j)}{P(x_i \simeq x_j | y_i \simeq y_j)} \tag{3}$$

in which $P(y_i \simeq y_j | x_i \simeq x_j)$ represents the probability that two random tuples' attribute values at Y are equivalent given that their attribute values at X are equivalent, and analogously $P(x_i \simeq x_j | y_i \simeq y_j)$ represents the probability that two tuples' attribute values at X are equivalent given that their attribute values at Y are equivalent.

Note that Eq. (3) essentially estimates the conditional probabilities between the individual values, $\{y_1, y_2\}$ and $\{x_1, x_2\}$, by the collective conditional probability relationship existing between Y and X in R. It can be observed that while the left-hand side of Eq. (3) can not be computed directly, the right-hand side can usually be easily computed in practice. For instance, in the running example shown in Table 1, suppose that Y and X correspond to the `journal` and `title` attributes respectively, and two attribute values are equivalent if and only if they are identical. Then, $P(y_i \simeq y_j | x_i \simeq x_j)$ corresponds to the probability that two paper records have the same `journal` value given that they have the same title. Similarly, $P(x_i \simeq x_j | y_i \simeq y_j)$ corresponds to the probability that two paper records have the same title given that they have the same `journal` value.

Denoting the right-hand side of Eq. (3) by the general correlation factor between Y and X, $f(Y, X)$, we have

$$f(Y, X) = \frac{P(y_i \simeq y_j | x_i \simeq x_j)}{P(x_i \simeq x_j | y_i \simeq y_j)} \tag{4}$$

Accordingly, the equivalence probability estimation for pair $\langle y_1, y_2 \rangle$ can be represented by

$$P(y_1 \simeq y_2) \approx f(Y, X) \cdot P(x_1 \simeq x_2) \tag{5}$$

Note that in Eq. (4), the two conditional probabilities can be represented by

$$\begin{cases} P(y_i \simeq y_j | x_i \simeq x_j) & = \frac{P(y_i \simeq y_j \wedge x_i \simeq x_j)}{P(x_i \simeq x_j)} \\ P(x_i \simeq x_j | y_i \simeq y_j) & = \frac{P(y_i \simeq y_j \wedge x_i \simeq x_j)}{P(y_i \simeq y_j)} \end{cases} \tag{6}$$

Therefore, by combining Eq. (4) with Eq. (6), we can represent the correlation factor between Y and X, $f(Y, X)$, as

$$f(Y, X) = \frac{P(y_i \simeq y_j)}{P(x_i \simeq x_j)} \tag{7}$$

where $P(y_i \simeq y_j)$ represents the equivalence probability of two random values in $Y(R)$, and $P(x_i \simeq x_j)$ represents the equivalence probability of two random values in $X(R)$.

In the basic model, we assume that two attribute values are equivalent iff they are identical. It estimates $P(x_1 \simeq x_2)$ in Eq. (5) by $P(x_1 = x_2)$. Similarly, it estimates the correlation factor, *f*(*Y, X*), by

$$f(Y,X) = \frac{P(y_i = y_j)}{P(x_i = x_j)} \tag{8}$$

in which $P(y_i = y_j)$ represents the probability of two random values in *Y*(*R*) being identical, and $P(x_i = x_j)$ represents the probability of two random values in *X*(*R*) being identical. Therefore, the equivalence probability estimation of y_1 and y_2 can be represented by

$$P(y_1 \simeq y_2) \approx f(Y,X) \cdot P(x_1 = x_2) \tag{9}$$

in which $P(x_1 = x_2)$ represents the probability of a random value in $X[y_1]$ being identical to a random value in $X[y_2]$, and the value of *f*(*Y, X*) is estimated by Eq. (8).

We observe that the computation of *f*(*Y, X*) as shown in Eq. (8) requires to retrieve all the *Y* and *X* attribute values in *R*. If the table *R* has a big size, full retrieval of its attribute values may take long time even if feasible. This observation motivates us to propose a scheme that can save equivalence probability estimation from the computation of *f*(*Y, X*). The proposed scheme is called *estimation by reference*, which only involves the attribute values at *X* correlated with y_1 and y_2 in *R*, $X[y_1]$ and $X[y_2]$.

Estimation by Reference Given two random tuples r_{1i}, $r_{1j} \in R[y_1]$, the probability of their attribute values at *Y* being equivalent can be estimated by Eq. (5) as follows:

$$P(y_{1i} \simeq y_{1j}) \approx f(Y,X) \cdot P(x_{1i} = x_{1j}) \tag{10}$$

where y_{1i} and y_{1j} represent r_{1i} and r_{1j}'s attribute values at *Y*, and x_{1i} and x_{1j} represent their attribute values at *X*. Since $y_{1i} = y_{1j} = y_1$, $p(y_{1i} = y_{1j}) = 1$. Therefore, by combining Eqs. (5) and 10, we can estimate the equivalence probability between y_1 and y_2 by

$$P(y_1 \simeq y_2) \approx \frac{P(x_1 = x_2)}{P(x_{1i} = x_{1j})} \tag{11}$$

in which $P(x_1 = x_2)$ represents the probability of a random value in $X[y_1]$ being identical to a random value in $X[y_2]$, and $P(x_{1i} = x_{1j})$ represents the probability of two random values in $X[y_1]$ being identical.

In Eq. (11), the denominator serves as the reference to determine the equivalence between y_1 and y_2. The closer the value of $P(x_1 = x_2)$ is to the value of $P(x_{1i} = x_{1j})$, the more probably y_1 and y_2 are equivalent. Suppose that in the running example, y_1="VLDB J" and y_2="Journal on Very Large Data Bases". Then, $P(x_1 = x_2)$ measures the title similarity between the papers published at "VLDB J" and those published at "Journal on Very Large Data Bases", and $P(x_{1i} = x_{1j})$ captures the title similarity among the papers published at "VLDB J". If these two title similarity measurements are very close in value, it can be expected that y_1 and y_2 refer to the same journal entity with a high probability.

We observe that both $R[y_1]$ and $R[y_2]$ can serve as the reference. In a practical implementation, we suggest to choose the one with the highest value of $P(x_{1i} = x_{1j})$. We can represent the equivalence probability of y_1 and y_2 by

$$P(y_1 \simeq y_2) \approx \frac{P(x_1 = x_2)}{\max_{a=1,2} P(x_{ai} = x_{aj})} \tag{12}$$

The probabilistic metric obtained by Eq. (12) is called as *Value Correlation Analysis* or *VCA*(y_1, y_2), which can be taken as the ratio of matching probability across clusters and the maximal counterpart within clusters.

3.1.2. An extension of basic model

The basic model as presented in Eq. (5) has limited applicability because it only considers identical attribute values. Consider the running example shown in Table 1. To determine the equivalence between two non-identical *journal* names, we compare their corresponding paper titles. In case that their papers do not share common titles, the basic model for *VCA*(y_1, y_2) would approximate to be 0. However, it can be observed that if two papers are published in the same `journal`, they may be in the same research area with a high probability; accordingly, their titles may be also similar. Therefore, it is important for *VCA* to capture the correlation between `journal` and `title` as shown in Table 1. To this end, we extend the basic model by incorporating string similarity measurement into the estimation of VCA.

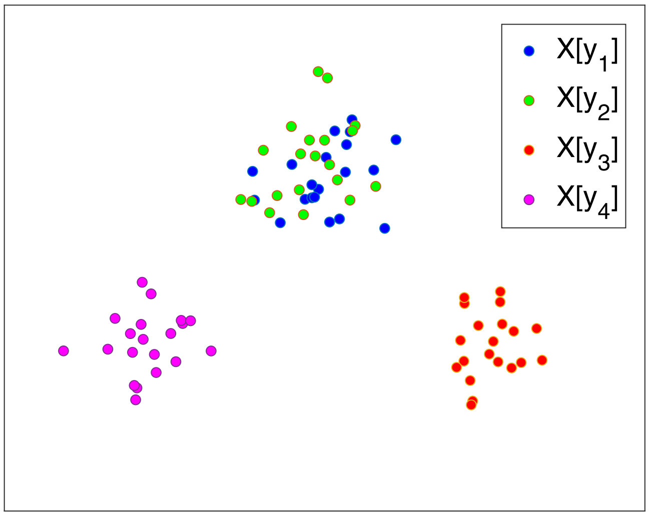


Fig. 1. An illustrative examples for *VCA*.

The extended model considers not only identical values, but also similar values in computing the VCA in Eq. (5). Previous work on attribute value matching assumes that the probability of two attribute values being equivalent equals to their string similarity. Under such assumption, $P(x_1 = x_2)$ in Eq. (5) can be substituted by $P(x_1 \approx x_2)$, which represents the average of string similarity of pairs across $X[y_1]$ and $X[y_2]$, and can be represented by

$$P(x_{1i} \approx x_{2j}) = \frac{\sum_{i,j} sim(x_{1i}, x_{2j})}{||X[y_1]|| \cdot ||X[y_2]||} \tag{13}$$

in which $x_{1i} \in X[y_1]$, $x_{2j} \in X[y_2]$ and $sim(x_{1i}, x_{2j})$ specifies the string similarity between x_{1i} and x_{2j}. Note that in Eq. (13), the numerator exhaustively aggregates the similarity of every value pair across $X[y_1]$ and $X[y_2]$, and the denominator represents the cardinality of Cartesian product $X[y_1] \times X[y_2]$.

Similarly, $P(x_{ai} \approx x_{aj})$ can be expressed by

$$P(x_{ai} \approx x_{aj}) = \frac{\sum_{i,j} sim(x_{ai}, x_{aj})}{||X[y_a]|| \cdot (||X[y_a]|| - 1)/2} \tag{14}$$

where x_{ai} and x_{aj} $(i \neq j)$ represent two random values from $X[y_a]$, and the denominator denotes the cardinality of value pairs.

By extending Eq. (12) with Eq. (13) and Eq. (14), we obtain Eq. (15).

$$VCA(y_1 \simeq y_2) = \frac{P(x_{1i} \approx x_{2j})}{\max_{a=1,2}(P(x_{ai} \approx x_{aj}))} \tag{15}$$

Analogously, *VCA*(y_1, y_2) in extended model, can be regarded as the ratio of the average similarity across clusters and the maximal counterpart within clusters.

Example 1. As shown in Fig. 1, assume that there are many research papers included in publication venue "*VLDB*" (denoted by y_1), and "*Very Large Database*" (denoted by y_2) respectively. Since y_1 and y_2 refer to the same entity of Journal, $X[y_1]$, $X[y_2]$, $X[y_1 \cup y_2]$ were deemed to be drawn from the same underlying distribution. Should sufficient samples be available, VCA for pairs that are equivalent, e.g., *VCA*(y_1, y_2), tends to approximate to 1. while *VCA* for pairs that are *not* equivalent, e.g., *VCA*(y_1, y_3) or *VCA*(y_3, y_4), are far less than 1.

On the complexity of *VCA*, we have following lemmas:

Lemma 1. *VCA*(y_1, y_2) *takes the complexity of* $O((N_1 + N_2)^2)$, *where* N_1 *and* N_2 *denote the cardinalities of* $X[y_1]$ *and* $X[y_2]$ *respectively.*

Lemma 2. *VCA*(y_i, y_j) *over all value pairs takes the complexity of* $O(N^2)$, *where* N *denotes the total number of tuples and pair* $\langle y_i, y_j\rangle \in \langle \mathbf{Y} \times \mathbf{Y}\rangle$.

3.1.3. Reasoning by multiple attributes

To reason about the equivalence between attribute values by multiple correlated attributes, a straightforward solution is to combine multiple attribute values into a long field and then treat the resulting string as a single attribute in correlation analysis. However, this approach does not differentiate the uneven importance of different attributes. Using a single string similarity metric, it cannot incorporate specific domain knowledge on individual attributes into the reasoning process either.

To address the limitation of the straightforward solution, we model the problem of probabilistic equivalence reasoning as a classification problem. It classifies each value pair, y_i and y_j, at a target attribute Y into one of two labels, *equivalent* and *inequivalent*. Other attributes in R (except Y) are considered as the feature sets of the objects (value pairs). Probabilistic reasoning by an individual attribute works as a classifier, which computes for its corresponding feature. Probabilistic reasoning by multiple attributes then corresponds to the problem of building an ensemble of classifiers and combining their classification results.

According to the ensemble theory [15], the performance of an ensemble depends on the performance of its component classifiers and their performance diversity. Therefore, the process of attribute selection for probabilistic reasoning consists of two steps. The first step of *attribute filtering* chooses the candidate attributes whose corresponding classifiers can achieve good performance. The following step of *combination generation* selects a combination of attributes from the candidate set with the aim to boost the performance diversity among the chosen attributes.

Attribute Filtering Informative attributes should have two properties:

1. high similarities for pairs drawn from the same clusters;
2. low similarities for pairs across clusters.

Informative attributes tend to provide strong evidence for matching attribute values. We represent such informativeness by *conditional correlation factor* (ccf).

We first define the conditional correlation factor between attribute X and Y_1 by

$$ccf(X, y_1) = \frac{P(x_{1i} \approx x_{1j})}{P(x_i \approx x_j)} \tag{16}$$

where x_{1i}, $x_{1j} \in X[y_1]$ and x_i, $x_j \in X[R]$.

Given a target attribute Y and a candidate attribute X, we measure their conditional correlation by a weighted sum of *ccf*(X, y_a) as follows:

$$ccf(X, Y) = \sum_{y_a \in \mathbf{Y}} (w_a \cdot ccf(X, y_a)) \tag{17}$$

in which y_a denotes a distinct value in domain $\mathbf{Y}$, w_a denotes its weight, which is proportional to the occurrence frequency of y_a in $Y[R]$. Intuitively, high value of *ccf* implies that high similarities for pairs drawn from the same clusters and low similarities for pairs drawn across clusters.

The process of *attribute selection* sets a threshold θ_c (e.g., θ_c = 5) on the value of measured conditional correlation. An attribute X will be included into the candidate set if and only if its conditional correlation with regard to Y is no smaller than θ_c, *ccf*(X, Y) $\geq \theta_c$. Intuitively, if the Y attribute values are independent of their corresponding X values, the value of *ccf*(X, Y) tend to achieve to 1.

Combination Generation

We first describe how to measure performance diversity among candidate attributes, and then present an algorithm to generate the attribute combination.

According to Eq. (5), given a candidate attribute, X_1, the equivalence probability of a pair of target Y attribute values, (y_i, y_j), as reasoned by X_1, can be represented by

$$P(y_i \simeq y_j) = f(Y, X_1) \cdot P(x_{1i} = x_{1j}) \tag{18}$$

in which $x_{1i} \in X_1[y_i]$, $x_{1j} \in X_1[y_j]$ and $P(x_{1i} = x_{1j})$ represents the equivalence probability between a random value in $X_1[y_i]$ and a random value in $X_1[y_j]$. Therefore, given two candidate attributes, X_1 and X_2, their performance difference on reasoning about the equivalence between y_i and y_j, $D_{(y_i, y_j)}(X_1, X_2)$ can be represented by

$$|f(Y, X_1) \cdot P(x_{1i} = x_{1j}) - f(Y, X_2) \cdot P(x_{2i} = x_{2j})| \tag{19}$$

Additionally, denoting $P(y_i = y_j)$ and $P(x_i = x_j)$ by $P(Y(R))$ and $P(X(R))$ respectively, we have

$$f(Y, X_1) = \frac{P(Y[R])}{P(X_1[R])} \tag{20}$$

$$f(Y, X_2) = \frac{P(Y[R])}{P(X_2[R])} \tag{21}$$

$$f(X_1, X_2) = \frac{P(X_1[R])}{P(X_2[R])} \tag{22}$$

Therefore, we have

$$f(Y, X_2) = f(Y, X_1) \cdot f(X_1, X_2) \tag{23}$$

Normalized by the estimation result by X_1, the performance difference between X_1 and X_2 can be rewritten as

$$D_{(y_i, y_j)}(X_1, X_2) = |1 - f(X_1, X_2) \cdot \frac{P(x_{2i} = x_{2j})}{P(x_{1i} = x_{1j})}| \tag{24}$$

Denoting the correlation between $X_1(y_i)$ and $X_2(y_j)$ by

$$f(X_1[y_i], X_2[y_j]) = \frac{P(x_{1i} = x_{1j})}{P(x_{2i} = x_{2j})} \tag{25}$$

we can therefore rewrite Eq. (24) as

$$D_{(y_i, y_j)}(X_1, X_2) = |1 - \frac{f(X_1, X_2)}{f(X_1[y_i], X_2[y_j])}| \tag{26}$$

Since probabilistic reasoning aims to identify equivalent attribute values, the classifier ensemble should be designed to boost the performance diversity of the classifiers on these value pairs. Therefore, we estimate the performance diversity between X_1 and X_2 by considering all the identical value pairs at Y, (y_i, y_i). Denoting $D_{(y_i, y_i)}(X_1, X_2)$ by $D_{y_i}(X_1, X_2)$, we have

$$D_{y_i}(X_1, X_2) = |1 - \frac{f(X_1, X_2)}{f(X_1[y_i], X_2[y_i])}| \tag{27}$$

Therefore, we can estimate the performance diversity between X_1 and X_2 on all the identical Y value pairs by

$$D_Y(X_1, X_2) \approx \sum_{y_i \in \mathbf{Y}} w_i \cdot D_{y_i}(X_1, X_2) \tag{28}$$

in which y_i represents a distinct value in **Y**, and w_i represents its weight, whose value is proportional to the occurrence frequency of y_i in **Y**.

The algorithm for generating attribute combination is sketched in Algorithm 1 . It iteratively selects the attribute X_j with the highest conditional correlation factor in the candidate attribute set S_C. If there exists an attribute X_i in the result attribute set, S_F, such that the performance diversity between X_i and X_j with regard to Y as estimated by Eq. (28) is below a threshold θ_d (e.g., 0.3), $D_Y(X_i, X_j) < \theta_d$, then X_j is filtered out; otherwise, X_j is inserted into S_F. The algorithm stops when the candidate set S_C becomes empty.

Algorithm 1: Algorithm for Generating Attribute Combination.

```
// $S_C$ denotes the candidate attribute set;
// $S_F$ denotes the set of chosen attributes;
// $Y$ denotes the target attribute;
$S_F = \emptyset$;
while ($S_C \neq \emptyset$) do
    Select the attribute $X_j$ in $S_C$ with the highest conditional
    correlation factor with regard to $Y$, $ccf(X_j, Y)$;
    if $S_F == \emptyset$ then
        Insert $X_j$ into $S_F$;
    else
        if (there does not exist $X_i \in S_F$ s.t. $D_Y(X_i, X_j) < \theta_d$) then
            Insert $X_j$ into $S_F$;
    Remove $X_j$ from $S_C$;
```

On the complexity of Algorithm 1, we have following lemma:

Lemma 3. *Provided that **performance diversities** have been computed beforehand, Algorithm 1 runs in $O(k^2)$ time in worse case, in which k denotes the number of attributes in R.*

Combination Rule

As in other applications of the ensemble theory, the probabilistic estimations computed by the classifiers for individual attributes are combined using the voting method. It first estimates the equivalence probability of y_1 and y_2 by each chosen attribute and then computes a weighted sum of the estimation results. The weight for an attribute X_i is proportional to the log of the conditional correlation factor between X_i and Y. Suppose that the chosen k attributes are $\{X_1, \ldots, X_k\}$. The combined probability is estimated by

$$VCA(y_1 \simeq y_2) = \sum_{1 \leq i \leq k} w_i \cdot VCA_i(y_1 \simeq y_2) \tag{29}$$

$VCA_i(y_1 \simeq y_2)$ X_i w_i X_i 29w_i

$$w_i = \frac{\ln(ccf(X_i, Y)}{\sum_{1 \leq i \leq k} \ln(ccf(X_i, Y))} \tag{30}$$

$ccf(X_i, Y)$ X_i $\ln(\cdot)$

3.2. The unified framework

The unified framework consists of two analytical components: string similarity measurement and value correlation analysis. We first describe a theory of evidence, the Dempster–Shafer theory, in Section 3.2.1, and then present how to use it to combine the estimation results obtained at the two components in Section 3.2.2.

3.2.1. The Dempster–Shafer theory

The Dempster-Shafer (*D-S*) theory [16,17], which is a generalization of the Bayesian theory of subjective probability, combines evidences from different sources and arrives at a degree of belief that takes into account all the available evidence.

Formally, let Z be the universal set representing all possible states of a system under consideration. By a function of *Basic Belief Assignment* (*BBA*), the D-S theory assigns a belief mass to each element of the power set. The mass of an element E_i, $m(E_i)$, expresses the proportion of all relevant and available evidence that supports the claim that the actual state belongs to E_i but to no particular subset of E_i. The masses of elements satisfy

$$\sum_{E_i \in 2^Z} m(E_i) = 1,$$

and

$$m(\emptyset) = 0.$$

If only singleton propositions are assigned belief masses, a BBA function reduces to a classical probability function.

The kernel of *D-S* theory is Dempster's rule, which is rooted in probability theory and constitutes a conjunctive probabilistic inference process. It generalises Bayes' rule and was indeed promoted as the sole evidence combination rule to combine evidence in the *D-S* framework originally. The Dempster's rule assumes that evidence is fully reliable: a proposition will not be supported at all if it is ruled out by an evidence. It adopts the orthogonal sum operation to combine evidence, which is rooted in calculating the joint probability of independent events. With two pieces of independent evidence represented by two *BBAs* m_1 and m_2 respectively, the joint mass of a proposition E is calculated in the following manner:

$$m_{1,2}(E) = \frac{\sum_{E_i \cap E_j = E \neq \emptyset} m_1(E_i) \cdot m_2(E_j)}{1 - \sum_{E_i \cap E_j = \emptyset} m_1(E_i) \cdot m_2(E_j)},$$

and

$$m_{1,2}(\emptyset) = 0.$$

in which $\sum_{E_i \cap E_j = \emptyset} m_1(E_i) \cdot m_2(E_j)$ measures the amount of conflict between the two mass sets.

3.2.2. Evidential reasoning

The unified framework assumes that the two pieces of evidences for attribute value equivalence, based on string similarity measurement and value correlation analysis respectively, are independent. It therefore uses the Dempster's rule of combination in evidential reasoning.

Suppose that the equivalence probability of two attribute values y_1 and y_2 estimated by string similarity measurement is denoted as $P_s(y_1 \simeq y_2)$; similarly, the probability estimated by value correlation analysis is denoted as $P_c(y_1 \simeq y_2)$. The BBA function of string similarity measurement assigns probability masses to singleton propositions in the following manner:

$$m_s(y_1 \simeq y_2) = P_s(y_1 \simeq y_2),$$

and

$$m_s(y_1 ! \simeq y_2) = 1 - P_s(y_1 \simeq y_2).$$

Similarly, the BBA function of value correlation analysis assigns probability masses to singleton propositions in the following manner:

$$m_c(y_1 \simeq y_2) = P_c(y_1 \simeq y_2),$$

and

$$m_c(y_1 ! \simeq y_2) = 1 - P_c(y_1 \simeq y_2).$$

Then, according to the Dempster's rule, the combined probability of y_1 and y_2 being equivalent is computed by

$$p_{s,c}(y_1 \simeq y_2) = \frac{p_s \cdot p_c}{1 - p_s \cdot (1 - p_c) - (1 - p_s) \cdot p_c} \tag{31}$$

in which p_s denotes $p_s(y_1 \simeq y_2)$ and p_c denotes $p_c(y_1 \simeq y_2)$

For instance, suppose that the equivalence probabilities estimated by similarity measurement and correlation analysis are 0.8 and 0.6 respectively. Then, the combined probability is computed by $\frac{0.8 \cdot 0.6}{1-0.8 \cdot 0.4-0.2 \cdot 0.6} = 0.857$.

According to the above formula, the unified estimation would be 0 if any of the individual estimations is computed to be 0. To address this counter-intuitive issue, we introduce evidence weights and translate them into simple support functions. We use the Shafer's discounting method [16] to combine two weighted evidence. Suppose $p(E_i)$ is the degree of belief to which a piece of evidence points to a proposition E_i. Let w be a factor that is used to discount $p(E_i)$, where w $(0 \le w \le 1)$ is interpreted as evidence weight. The Shafer's discounting method defines the BBA function for an evidence as follows:

$$m(E_i) = \begin{cases} w \cdot p(E_i) & E_i \subset Z, E_i \neq \emptyset \\ 0 & E_i = \emptyset \\ 1 - w & E_i = Z \end{cases} \quad (32)$$

What Eq. (32) means is that the degree of support for a proposition is proportional to both the weight of the evidence and the belief degree to which the evidence points to the proposition. It allocates the residual support left by the evidence due to its weight, as measured by $(1 - w)$, to Z. We have noticed that there are other methods [17] for weighted evidence combination. The difference is on how to handle the residual support, as measured by $1 - w$. We propose to use the Shafer's discounting method because assigning the residual support to Z is most reasonable in this setting.

Generally, suppose that the evidence weights of similarity measurement and correlation analysis are w_s and w_c respectively. Then, we have

$$m_s(y_1 \simeq y_2) = w_s \cdot p_s,$$

$$m_s(y_1 ! \simeq y_2) = w_s \cdot (1 - p_s),$$

and

$$m_s(Z) = 1 - w_s.$$

Similarly, we also have

$$m_c(y_1 \simeq y_2) = w_c \cdot p_c,$$

$$m_c(y_1 ! \simeq y_2) = w_c \cdot (1 - p_c),$$

and

$$m_s(Z) = 1 - w_c.$$

Therefore, the unified equivalence probability can be computed by

$$p_{s,c}(y_1 \simeq y_2) = \frac{w_s \cdot p_s \cdot w_c \cdot p_c + w_s \cdot p_s \cdot (1-w_c) + w_c \cdot p_c \cdot (1-w_s)}{1 - w_s \cdot p_s \cdot w_c \cdot (1-p_c) - w_c \cdot p_c \cdot w_s \cdot (1-p_s)} \quad (33)$$

Consider again the case that the equivalence probabilities estimated by similarity measurement and correlation analysis are 0.8 and 0.6 respectively; but both evidence weights are 0.5. Then, we have

$$m_s(y_1 \simeq y_2) = 0.4,$$

$$m_s(y_1 ! \simeq y_2) = 0.1,$$

and

$$m_s(Z) = 0.5.$$

We also have

$$m_c(y_1 \simeq y_2) = 0.3,$$

$$m_c(y_1 ! \simeq y_2) = 0.2,$$

and

$$m_c(Z) = 0.5.$$

The combined probability is therefore computed by

$$m_{s,c}(y_1 \simeq y_2) = \frac{0.4 \cdot 0.3 + 0.4 \cdot 0.5 + 0.3 \cdot 0.5}{1 - 0.4 \cdot 0.2 - 0.3 \cdot 0.1} = 0.528$$

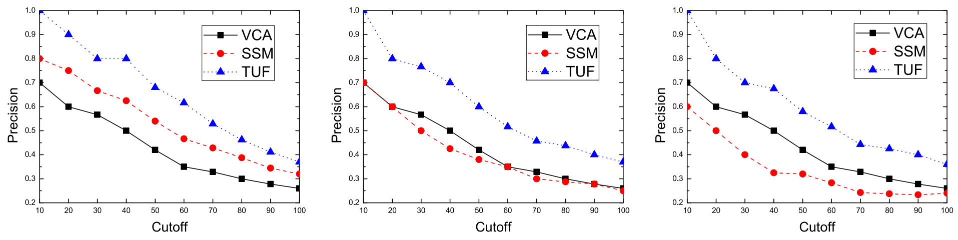
Our experimental evaluation in Section 4.3 shows that evidence weighting can affect the performance of the unified framework and it performs best when the two weights are set to be equal. Therefore, we suggest to set the weights of string similarity measurement and value correlation analysis to be equal in practical implementation. It is worthy to point out that the unified framework is reduced into string similarity measurement or value correlation analysis if the weight of the other component is set to be 0.

4. Experimental study

In this section, we empirically evaluate the performance of the unified framework on real datasets. The details of these test datasets are described as follows:

1. DBLP. The DBLP dataset[1] contains the information on the papers published in the research venues in various areas of computer science. The original dataset is clean, containing no dirty data. For experimental purpose, we introduce dirty data by manipulating the attribute values at `journal`, which specifies the venue a paper is published in. We transform the `journal` values by performing a series of edit operations, e.g. *insertion, deletion* and *substitution*, as defined in the Edit distance[18]. We specify the numbers of edit operations executed on `journal` values by a normal distribution, $\mathcal{N}(\mu, \sigma^2)$, which we refer to as the manipulation function in the rest of this section.
2. HOTEL. The dataset contains the information of more than 138,000 hotels around the world. Similar to the case of DBLP, we manually generate dirty data by manipulating the attribute values at `city` and `aircode`, which specify the city a hotel is located at and the code of its nearest airport respectively. As on DBLP, we transform the values by performing a series of edit operations as defined in the Edit distance. The numbers of executed edit operations on the `city` values are also specified by a normal distribution, $\mathcal{N}(\mu, \sigma^2)$. The `aircode` values have only three characters. Therefore, we randomly replace 1 or 2 characters at the chosen values.
3. DBLP+CiteSeer (DLCS). Similar to DBLP, the CiteSeer dataset[2] also records the published research papers in computer science. An author and a publication venue may have different representations in DBLP and CiteSeer. Our experiments aim to match the attribute values of `journal` and `author` between them. DBLP and CiteSeer contain around 960,000 and 45,000 tuples respectively. To facilitate repeated experimentation, we randomly choose 20% tuples in DBLP and match them with all the tuples in CiteSeer.
4. CORA. The CORA dataset[3] also contains the published research papers in computer science. Unlike the DBLP and CiteSeer datasets, it is pretty dirty by itself. In CORA, two records may refer to the same paper entity but their publication venues have different representations. The dataset contains totally 1295

[1] http://dblp.uni-trier.de/xml/
[2] https://www.cs.purdue.edu/commugrate/data/citeseer/
[3] https://people.cs.umass.edu/~mccallum/data.html



(a) journal($\mathcal{N}(\mu$=3,σ=2)) (b) journal($\mathcal{N}(\mu$=5,σ=2)) (c) journal($\mathcal{N}(\mu$=7,σ=2))

Fig. 2. Precision-cutoff evaluation on DBLP.

records and 474 attribute values at venue, which specifies publication venue. Our experiments aim to match the attribute values at venue. Since the venue values in CORA carry time information, two venues are considered to be different if they correspond to the same conference/journal series but have distinct publication time (e.g. “the 31th IEEE International Conference on Data Engineering” and “the 32nd IEEE International Conference on Data Engineering”).

Our experiments mainly compare the performance of four approaches, string similarity measurement (SSM), value correlation analysis (VCA) and the unified framework (TUF). Additionally, we compare TUF to the approach of collective entity resolution (CER) proposed in [19], which takes advantage of co-occurring references to reason about entity equivalence jointly rather than independently. Note that the effectiveness of the CER approach depends on the presence of value co-occurrence in a record. Compared with TUF, CER has more limited applicability in practice. For instance, CER can only work on the author names of the bibliographic datasets, but not on their journal attribute values; CER can't work on the HOTEL dataset either. In comparison, TUF can work on both the author and journal attributes of the bibliographic datasets, and it can also work on the HOTEL dataset.

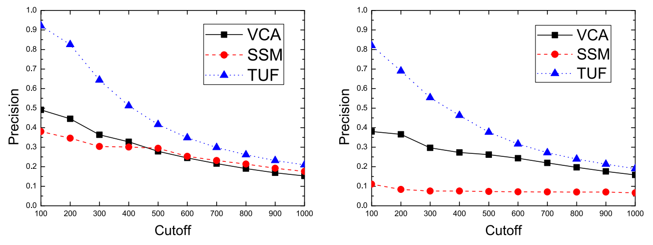
4.1. Comparative evaluation

In this subsection, we first compare the performance of SSM, VCA and TUF (Section 4.1.1 on the DBLP dataset, Section 4.1.2 on HOTEL and Section 4.1.3 on DLCS) and then compare TUF to CER (Section 4.1.4). The unified framework sets both weights of SSM and VCA to the default value of 0.5. On the DBLP dataset, VCA reasons about journal by title and author. On HOTEL, VCA reasons about city by address, and aircode by address and city. On DLCS, VCA reasons about journal by title and author, and author by title and journal. On CORA, VCA reasons about venue by title and author. Reasoning over multiple attributes are presented in Section 3.1.3.

4.1.1. DBLP

To evaluate the comparative performance in the circumstances where SSM has varying accuracies, we set the value of *standard deviation* (σ) of the manipulation function to 2 but vary its value of *mean* (μ) from 3 to 7. Note that as the value of μ increases, the equivalent attribute values would appear less similar. As a result, the performance of SSM would deteriorate.

The precisions measured over top-K (cutoff) are presented in Fig. 2. We present the precision results at the cutoff levels up to 100, where TUF achieves the maximal recalls. In the case of μ=3, it can be observed that even though SSM performs better than VCA, TUF performs clearly better than SSM, and the performance gain ranges from 8% to 30%. Close scrutiny reveals that VCA tends to match the journals in the same research area, whose papers have similar titles and share some common authors. It effectively filters out the *journal* names that appear similar but are actually in different research areas from its top ranked list.



(a) city($\mathcal{N}(\mu$=3,σ=2)) (b) aircode($\mathcal{N}(\mu$=3,σ=2))

Fig. 3. Precision-cutoff evaluation on HOTEL.

It can also be observed that as the value of μ increases, the performance of SSM deteriorates as expected while the performance of VCA remains stable. As a result, the performance improvement achieved by TUF over SSM becomes more considerable. As the value of μ increases, SSM becomes less reliable in determining value equivalence. The performance of VCA is instead independent of the representations of values on journal. Since TUF leverages both SSM and VCA estimation results, its performance improvement over SSM increases as the performance of SSM deteriorates.

4.1.2. HOTEL

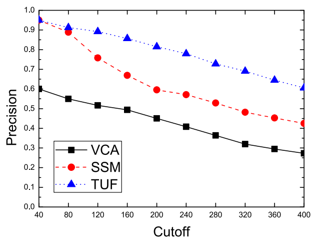
On the city values, the mean and standard deviation of the manipulation function are set to be 3 and 2 respectively, $\mu = 3$ and $\sigma = 2$. The aircode values have only three characters. Therefore, we randomly replace 1 or 2 characters at the chosen values.

The comparative precision-cutoff evaluation results are presented in Fig. 3. We report the precision results up to top-1000. After the cutoff level of 1000, every approach generates few (≤ 5) correct matches.

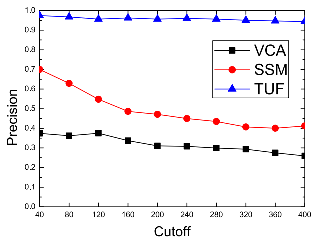
It can be observed that on city, even though neither SSM nor VCA achieves the precision above 50%, TUF achieves a precision above 90% at the cutoff level of 100. On aircode, it can be observed that SSM performs very poorly. With only three characters, many aircode value pairs have the same string similarity but most of them actually represent different airports. There are a lot of false positive matches in the top-k results. As a result, the performance of SSM remains low (around 10%) at various cutoff levels. In comparison, by analyzing the correlated city and address values, VCA achieves much higher matching accuracy. It can also be observed that TUF's performance gains over SSM range from 5% to 50% over city, and from 11% to 70% over aircode respectively.

4.1.3. DLCS & CORA

On the DLCS and CORA datasets, we evaluate the precision-cutoff performance of different approaches. On one hand, manually



(a) on `journal`



(b) on `author`

Fig. 4. Precision-cutoff evaluation on DLCS.

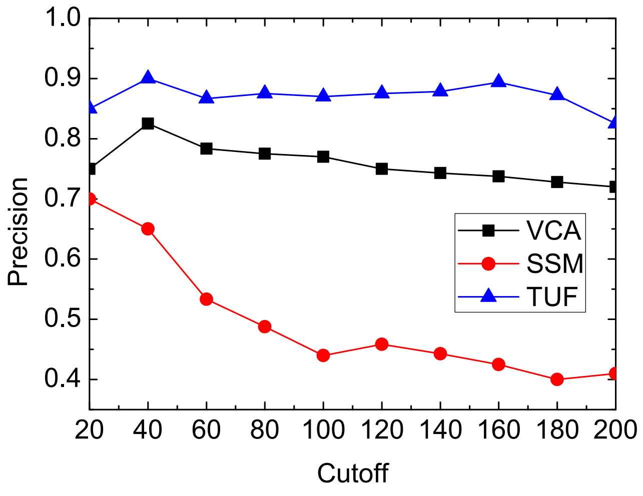


Fig. 5. Precision-cutoff evaluation on CORA.

checking the true recall values on these datasets is a daunting task. On the other hand, if the cutoff level is set to be large enough, the percentage of matching pairs among those ranked after the cutoff level could be expected to be very small; therefore, further increasing the cutoff level could only have negligible effect on the comparative performance of different approaches. In our experiments, the cutoff levels of DLCS and CORA are set to be 400 and 200 respectively. After the cutoff levels, the percentage of matching pairs becomes small enough such that the comparative performance of different approaches stabilize. The experimental results on DLCS are shown in Fig. 4 (a) and (b). Similar to what were observed on DBLP and HOTEL, TUF consistently outperforms SSM on matching precision. On `journal`, SSM performs well at the cutoff levels of 40 and 80. It achieves the precisions of 97.5% and 87.5% respectively. At these two cutoff levels, TUF achieves slightly better performance than SSM. After that, TUF outperforms SSM by considerable margins: at the cutoff levels between 160 and 400, TUF achieves around 20% improvement over SSM.

On `author`, SSM does not perform as well as on `journal`. In DLCS, there are many author names that appear similar but actually represent different researchers. As a result, SSM returns many false positive matches in its top-k results. It can be observed that even though VCA performs worse than SSM, TUF performs much better than SSM. After the cutoff level of 80, TUF achieves at least 30% improvement over SSM on precision.

The experimental results on CORA are presented in Fig. 5. Since many values on `venue` values contain the time token either explicitly or implicitly, e.g. “ICDE,1998”, “31st ICDE”. Values are taken as equivalents only if they are matching on both of publication venue and time token (either explicit or implicit) in our experiment. It can be observed that SSM performs poorly. Ranking by SSM results in many false positive matchings in the top-k pairs. In

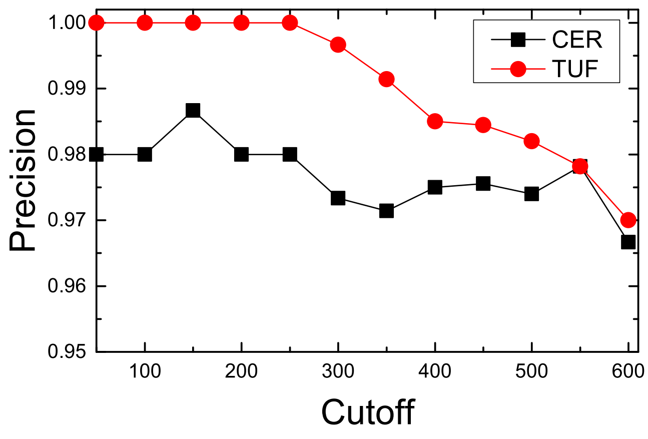


Fig. 6. TUF vs CER on Citeseer.

contrast, VCA performs well on CORA, achieving precision between 70% and 80% at various cutoff levels. By leveraging both analytical results of SSM and VCA, TUF achieves the best performance among them, and the performance gain of TUF over SSM ranges from 15% to 45%.

4.1.4. TUF vs CER

Since the approach of CER can only work on the attribute values with the co-occurrence relationship, we compare the performance of TUF and CER on the `author` attribute of the single-source Citeseer dataset. In Citeseer, very few author names are ambiguous. Therefore, we ignore name ambiguity and suppose that identical author names refer to the same person. We have implemented the CER approach as presented in [19]. In the experiment, both CER and TUF used the hybrid metric, which augments a primary TF-IDF weighting scheme for matching token sets with a secondary Jaro-Winkler scheme for matching tokens, to measure string similarity. We have actually tested different secondary metrics for CER, and observed that as reported in [19], CER performs best with the secondary metric of Jaro-Winkler. As in Section 4.1.3, we measure the performance by the precisions at various cutoff levels. The comparative results are presented in Fig. 6. It can be observed that due to limited growth space, performance gain (2%) seems small, however TUF performs consistently better than CER.

4.2. Attribute combination

This subsection evaluates the effectiveness of attribute selection and estimation combination in value correlation analysis, as presented in Section 3.1.3. On all the three datasets, the threshold of conditional correlation factor (θ_c) for candidate filtering is set to be 5; the threshold of the performance diversity between two candidate attributes (θ_d) for combination generation is set to be 0.3. The conditional correlation factor of chosen attributes and their performance diversity with regard to a target attribute are detailed in Table 3.

The evaluation results on DBLP are presented in Fig. 7 (a) and (b). On DBLP, we set the value of μ to be 5. The evaluation results for other values of μ are similar, thus omitted here. Value correlation analysis are executed on `author`, `title` and the combination of `author` and `title` respectively. In Fig. 7 (a), it can be observed that VCA on `author` performs better than VCA on `title`. Similarly, in Fig. 7 (b), it can be observed that TUF with VCA on `author` performs better than TUF with VCA on `title`. We also observe that VCA on the attribute combination clearly outperforms VCA on single attributes. The comparative performance of their corresponding TUF are similar. The evaluation results on

Table 3
The values of *ccf*(X, Y) on the Datasets.

Dataset(Y)	*ccf*(X_1, Y)	*ccf*(X_2, Y)	$D_Y(X_1, X_2)$
DBLP(`journal`) X_1=`author` X_2=`title`	24.70	14.84	0.48
DLCS(`journal`) X_1=`author` X_2=`title`	332.50	69.60	0.79
HOTEL(`aircode`) X_1=`city` X_2=`address`	440.14	93.15	0.81

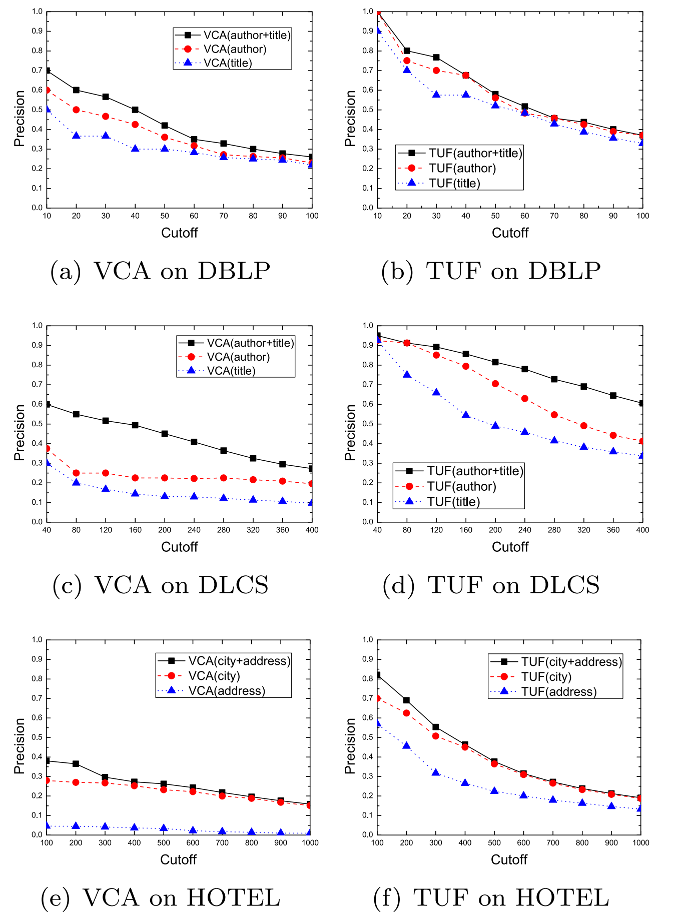


Fig. 7. Precision-cutoff evaluation results of attribute combination on DBLP and HOTEL.

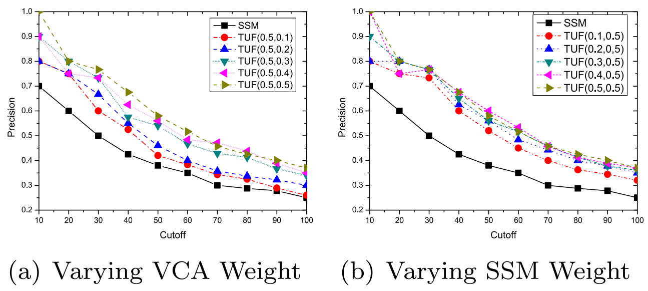


Fig. 8. Evidence weighting.

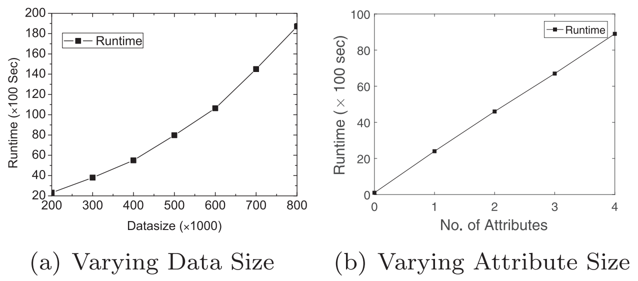


Fig. 9. Efficiency evaluation.

DLCS are also presented in Fig. 7 (c) and (d). They are similar to what were observed on DBLP.

On HOTEL, the value of *ccf*(`city`, `aircode`) is larger than the value of *ccf*(`address`,`aircode`). As a result, VCA on `city` performs better than VCA on `address`. VCA on the attribute combination outperforms both of them. TUF with VCA on the attribute combination also achieves the best performance.

Our experiments show that the higher conditional correlation factor a candidate attribute has, the better performance its corresponding VCA and TUF can achieve. Moreover, the performances of VCA and TUF are better on attribute combination than on any single attribute. They validate the effectiveness of our approach for value correlation analysis based on multiple attributes proposed in Section 3.1.3.

4.3. Evidence weighting

This subsection evaluates how evidence weighting can affect the performance of the unified framework. We present the evaluation results on matching `journal` values on the DBLP dataset. The results on other datasets are similar, thus omitted here due to space limit. The μ and σ values of the normal function are set to be 5 and 2 respectively. Our experiments showed that if the weights of SSM and VCA are set to be equal ($0 < w < 1$), the performance of TUF remains stable regardless of the value of w. Therefore, we set the weight of one component to the default value of 0.5 and vary the weight of the other component from 0.1 and 0.5.

The detailed results are presented in Fig. 8. From Fig. 8 (a), it can be observed that if the weight of SSM is set to be 0.5, TUF achieves the best performance when the weight of VCA is also set to be 0.5. TUF with any weight setting outperforms SSM and its performance consistently improves as the weight of VCA increases from 0.1 to 0.5. The experimental results for the case of varying the SSM weight, as presented in Fig. 8 (b), are similar. TUF with both weights set to be the default value of 0.5 achieves the best performance.

4.4. Scalability evaluation

The algorithms were evaluated on DBLP. VCA reasons about `journal` by `author`. In the first case, we vary data size (measured by the total number of tuples) from 200,000 to 800,000. In the second case, the number of tuples is set to 200, to better measure the scalability with the number of attributes, `author` attribute is duplicated multiple times and appended to the relation *R*, and the horizon-axis of Fig. 9 (b) denote number of duplicates.

The experimental results are presented in Fig. 9. From Fig. 9 (a), we observe that the consumed time of VCA increases nearly quadratically with data size (consistent with Lemma 2). From Fig. 9 (b), we observe that our approach scales linear with the number of attributes.

5. Related work

Attribute Values Matching is an important research problem in data quality community, and the state-of-the-art approaches for attribute value matching are mainly based on string similarity among attribute values. In general, string similarity metrics can be broadly classified into three categories[8]: *character-based, token-based* and *hybrid*. The *character-based* metrics treat an attribute value as a long string. The most typical ones include *edit distance*, affine gap cost [20] or the *Jaro* metric [10]. The *token-based* metrics, e.g., Jaccard Similarity, convert the strings to token sets and consider similarity metrics on these sets. The token-based metrics are usually more effective than the character-based ones in the circumstances where word order is unimportant. Similar to the token-based metrics, the *hybrid* metrics [21,22] convert the strings to token multisets. It also considers token similarity in measuring the similarity between these multisets. However, as mentioned in [8], different string similarity metrics usually have distinct suitable application domains.

Beside the string similarity measurement, we also take into consideration the evidential support from *value correlation analysis*, which is distinct from existing approaches: (I) Constraint-based approaches, e.g., [23–25], have been widely used in data cleaning. However, to match attribute values, the effectiveness of such deterministic approaches may heavily rely on the quality of constraints (either specified by experts or discovered by automatic algorithms) and the number of pairs of tuples captured by matching pattern. In comparison, *value correlation analysis* or *VCA*, takes a probabilistic approach and has much wider application scenario. Those matching candidates failing to be captured as violations, e.g., “Journal on Very Large Databases” vs “VLDB J” in Table 1, can still be further resolved by *VCA*. (II) Bayesian theory [26] is also widely used in pattern recognition and statistical inference. A significant distinction lies in that the labels are predefined and distinct in *Bayesian classification*, whereas in the task of *attribute value matching*, the equivalence between labels need to be reasoned about. Since Bayesian-based approaches can not be used directly for the task of matching attribute values, we propose the novel concept of *value correlation analysis*, which is rooted in Bayesian theory and finally reduced to the ratio of matching probability across clusters (e.g., venues) and the maximal counterpart within clusters.

Additionally, we also list extensively some other closely related research work.

Duplicate Record Detection Duplicate record detection have been extensively studied in the literature [3,27]. Most effective and widely used approaches [28–30] were built on value matching on individual attribute fields. They solved the additional problem of how to merge the matching results on attribute fields. Obviously, our work on attribute value matching is complementary to them in that determining attribute value equivalence can effectively improve the accuracy of record matching.

For efficient duplicate detection, there also exist some reasonably recent work on blocking techniques [31–33], which separate records into blocks likely to contain matching pairs such that unnecessary pair-wise comparisons can be avoided. It can be observed that our work on attribute value equivalence reasoning can be used to improve the blocking accuracy.

Schema Matching Schema matching [34,35] studies the problem of identifying columns that represent the same concept in two relational tables. Effective techniques usually take a hybrid approach by leveraging different criteria (e.g. linguistic matching, instance-based matching, structured-based matching, constraint-based matching and rule-based matching). The task of instance-based matching studied in schema matching treats all the value instances belonging to an attribute as a set and measures the set similarity by token-based metrics. Its matching techniques are similar to the techniques of string similarity measurement used for value equivalence reasoning in that both of them reason based on contents. In contrast, our work in this paper proposes to reason about two attribute values' equivalence by analyzing their correlation with other attribute values. On the other hand, given a schema matching result, our work can effectively reduce equivalence ambiguity between attribute values and thus improve the accuracy of record linkage.

Entity Name Disambiguation There also exist work on entity name disambiguation in relational data [19] or on Web [36,37]. They assumed that identical names may refer to different real-world entities and focused on distinguishing these names.

In the circumstance that two identical values may refer to two different values, equivalence reasoning is very challenging because every appearance of value should be assumed to correspond to a distinct entity at the outset. An alternative solution is to first cluster distinct values into equivalent groups, which is the topic of this paper, and then distinguish the values in the same group. It is worthy to point out that our proposed approach can be similarly used to distinguish two identical values. In the case that two identical values refer to two distinct entities, it can be supposed that they are correlated with largely different other values. The application of the proposed approach to value disambiguation is however beyond the scope of this paper. *Using Information from External Knowledge Base* The Web presents a huge pool of useful knowledge, and we observe that various systems [38,39] have been designed for information extraction on Web. Those extracted information can be used either by adding more columns or rows, to enrich the relational data, on which *value correlation analysis* is run.

6. Conclusion

This paper first presents a novel probabilistic approach to reason about attribute value equivalence in relational data by value correlation analysis. It then proposes a unified framework that can leverage both string similarity measurement and value correlation analysis. Finally, our extensive experiments on real datasets have validated the efficacy of the unified framework.

In this paper, we have focused on equivalence reasoning on string values. Theoretically, the effectiveness of value correlation analysis does not depend on the data type of attribute values. Equipped with appropriate similarity metrics, the unified framework can also be used for equivalence reasoning on numerical values. However, the existing methods for capturing similarity between numerical values are rather primitive. Therefore, the efficacy of our proposed approach on numerical values needs to be further investigated in future work.

In our default setting, only one attribute is taken as target attribute, on which value pairs will be matched. In the case that value equivalence should be considered on multiple attributes, they can be processed one by one. However, determining value equivalence on multiple attributes simultaneously is an interesting problem, which may deserve much more complex model or algorithm, thus can be taken as an interesting research point in future.

Acknowledgements

The work was supported by the Ministry of Science and Technology of China, National Key Research and Development Program (Project Number 2016YFB1000703), the National Natural Science Foundation of China under No.61332006, No.61672432 No.61472321 and No.61502390.

References

[1] C. Batini, M. Scannapieco, Data Quality: Concepts, Methodologies and Techniques, Springer Publishing Company, Incorporated, 2010.

[2] F. Naumann, M. Herschel, An introduction to duplicate detection, Synth. Lect. Data Manage. 2 (1) (2010) 1–87.
[3] A.K. Elmagarmid, P.G. Ipeirotis, V.S. Verykios, Duplicate record detection: a survey, IEEE Trans. Knowl. Data Eng. 19 (1) (2007) 1–16.
[4] M. Weis, F. Naumann, U. Jehle, J. Lufter, H. Schuster, Industry-scale duplicate detection, Proc. VLDB Endowment 1 (2) (2008) 1253–1264.
[5] C. Fei, R.J. Miller, Discovering data quality rules, Proc. VLDB Endowment 1 (1) (2008) 1166–1177.
[6] W. Fan, F. Geerts, J. Li, M. Xiong, Discovering conditional functional dependencies, IEEE Trans. Knowl. Data Eng.. 23 (5) (2011) 683–698.
[7] M. Cheatham, P. Hitzler, String similarity metrics for ontology alignment, in: International Semantic Web Conference, Springer, 2013, pp. 294–309.
[8] Y.U. Minghe, L.I. Guoliang, D. Deng, J. Feng, String similarity search and join: a survey, Front. Comput. Sci. 10 (3) (2016) 399–417.
[9] G. Navarro, A guided tour to approximate string matching, ACM Comput. Surv. 33 (1) (2000) 31–88.
[10] M.A. Jaro, Unimatch: a Record Linkage System: Users Manual, Bureau of the Census, 1978.
[11] L. Gravano, P.G. Ipeirotis, N. Koudas, D. Srivastava, Text joins in an rdbms for web data integration, in: World Wide Web Conference Series, 2003, pp. 90–101.
[12] J. Gong, L. Wang, D.W. Oard, Matching person names through name transformation, in: ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November, 2009, pp. 1875–1878.
[13] M. Bilenko, R.J. Mooney, Adaptive duplicate detection using learnable string similarity measures, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 39–48.
[14] S. Tejada, C.A. Knoblock, S. Minton, Learning domain-independent string transformation weights for high accuracy object identification, in: SIGKDD, 2002, pp. 350–359.
[15] L. Rokach, Ensemble-based classifiers, Artif. Intell. Rev. 33 (1) (2010) 1–39.
[16] G. Shafer, et al., A mathematical theory of evidence, 1, Princeton university press Princeton, 1976.
[17] J.B. Yang, D.L. Xu, Evidential reasoning rule for evidence combination, Artif. Intell. 205 (205) (2013) 1–29.
[18] G. Navarro, A guided tour to approximate string matching, ACM Comput. Surv. 33 (1) (2001) 31–88.
[19] I. Bhattacharya, L. Getoor, Collective entity resolution in relational data, ACM Trans. Knowl. Disc. Data (TKDD) 1 (1) (2007) 5.
[20] M.S. Waterman, T.F. Smith, W.A. Beyer, Some biological sequence metrics, Adv. Math. 20 (3) (1976) 367–387.
[21] J. Wang, G. Li, J. Fe, Fast-join: an efficient method for fuzzy token matching based string similarity join, in: IEEE International Conference on Data Engineering, 2011, pp. 458–469.
[22] J. Wang, G. Li, J. Feng, Extending string similarity join to tolerant fuzzy token matching, ACM Trans. Database Syst. (TODS) 39 (1) (2014) 7.
[23] P. Bohannon, W. Fan, F. Geerts, X. Jia, A. Kementsietsidis, Conditional functional dependencies for data cleaning, in: IEEE International Conference on Data Engineering, 2007, pp. 746–755.
[24] G. Cong, W. Fan, F. Geerts, X. Jia, S. Ma, Improving data quality: consistency and accuracy, in: International Conference on Very Large Data Bases, 2007, pp. 315–326.
[25] N. Koudas, A. Saha, D. Srivastava, S. Venkatasubramanian, Metric functional dependencies, in: IEEE International Conference on Data Engineering, 2009, pp. 1275–1278.
[26] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., 2006.
[27] I.M. Nguena, Fast semantic duplicate detection techniques in databases, J. Softw. Eng. Appl. 10 (6) (2017) 529–545.
[28] I.P. Fellegi, A.B. Sunter, A theory for record linkage, J. Am. Stat. Assoc. 64 (328) (1969) 1183–1210.
[29] Y.R. Wang, S.E. Madnick, The inter-database instance identification problem in integrating autonomous systems, in: International Conference on Data Engineering, 1989. Proceedings, 1989, pp. 46–55.
[30] M. Cochinwala, V. Kurien, G. Lalk, D. Shasha, Efficient data reconciliation, Inf. Sci. 137 (1–4) (2001) 1–15.
[31] B. Kenig, A. Gal, Mfiblocks: an effective blocking algorithm for entity resolution, Inf. Syst. 38 (6) (2013) 908–926.
[32] G. Papadakis, E. Ioannou, T. Palpanas, C. Niederee, W. Nejdl, A blocking framework for entity resolution in highly heterogeneous information spaces, IEEE Trans. Knowl. Data Eng. 25 (12) (2013) 2665–2682.
[33] G. Simonini, S. Bergamaschi, H.V. Jagadish, Blast: a loosely schema-aware meta-blocking approach for entity resolution, VLDB Endow., 2016, pp. 1173–1184.
[34] E. Sutanta, R. Wardoyo, K. Mustofa, E. Winarko, Survey: models and prototypes of schema matching, Int. J. Electr. Comput. Eng. 6 (3) (2016) 1011.
[35] Q.V.H. Nguyen, T.T. Nguyen, V.T. Chau, T.K. Wijaya, Z. Miklos, K. Aberer, A. Gal, M. Weidlich, Smart: A tool for analyzing and reconciling schema matching networks, in: Data Engineering (ICDE), 2015 IEEE 31st International Conference on, IEEE, 2015, pp. 1488–1491.
[36] Z.C. Zheng, X.Y. Zhu, Entity disambiguation with type taxonomy, J. Comput. Inf. Syst. 9 (3) (2007) 1199–1207.
[37] J. Fu, J. Qiu, Y. Guo, L. Li, Entity linking and name disambiguation using svm in chinese micro-blogs, in: International Conference on Natural Computation, 2016, pp. 468–472.
[38] R. Krishnamurthy, Y. Li, S. Raghavan, F. Reiss, S. Vaithyanathan, H. Zhu, Systemt: a system for declarative information extraction, ACM Sigmod Rec. 37 (4) (2008) 7–13.
[39] P. Jiménez, R. Corchuelo, Roller: a novel approach to web information extraction, Knowl. Inf. Syst. 49 (1) (2016) 197–241.