# Entity Profiling with Varying Source Reliabilities

Furong Li
National University of Singapore
Computing 1, Computing Drive
furongli@comp.nus.edu.sg

Mong Li Lee
National University of Singapore
Computing 1, Computing Drive
leeml@comp.nus.edu.sg

Wynne Hsu
National University of Singapore
Computing 1, Computing Drive
whsu@comp.nus.edu.sg

## ABSTRACT

The rapid growth of information sources on the Web has intensified the problem of data quality. In particular, the same real world entity may be described by different sources in various ways with overlapping information, and possibly conflicting or even erroneous values. In order to obtain a more complete and accurate picture for a real world entity, we need to *collate* the data records that refer to the entity, as well as *correct* any erroneous values. We observe that these two tasks are often tightly coupled: rectifying erroneous values will facilitate data collation, while linking similar records provides us with a clearer view of the data and additional evidence for error correction.

In this paper, we present a framework called COMET that interleaves record linkage with error correction, taking into consideration the source reliabilities on various attributes. The proposed framework first utilizes confidence based matching to discriminate records in terms of ambiguity and source reliability. Then it performs adaptive matching to reduce the impact of erroneous values. Experiment results demonstrate that COMET outperforms the state-of-the-art techniques and is able to build complete and accurate profiles for real world entities.

## Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Management—*data mining*

## Keywords

Record linkage; source reliability; truth discovery; entity profiling

## 1. INTRODUCTION

In the age of Big Data, a real world entity's information is, more often than not, published by more than one data source. Each of the data source may describe the same entity with name variations and provide incomplete and overlapping information. In order to obtain a complete picture of a real world entity, we need to collate the data records that refer to this entity. To complicate matters, not all the data sources are reliable and may publish erroneous information.

While the problem of record linkage has been well-studied [7, 15], linking and merging information from multiple sources remains a challenging task [20, 18]. The work in [20] observes that sources have varying semantic ambiguity and proposes a framework to apply either a relaxed or a conservative matching criteria depending on how ambiguous is the source. On the other hand, [18] develops a transfer learning approach to learn a unified matching function to link records from multiple sources. These works do not explicitly consider the erroneous information in the data.

The work in [12] is the first attempt to link records in the presence of erroneous values. The authors transform the problem into a k-partite graph clustering problem where each node in the graph represents an attribute value and each edge associates a pair of values from the same record. This approach is computationally expensive and its performance degrades when the percentage of erroneous values increases, as shown in our experiments.

In this paper, we present an effective and efficient framework called COMET to *collate* data records from multiple sources, *correct* any erroneous attribute values, and *construct* profiles for real world entities.

EXAMPLE 1. *Suppose we want to collate information on researchers in Computer Science. We could first obtain a set of reference records from some well-established source such as the acm.org website. Table 1 shows the names and affiliations of selected Computer Science researchers. Since the information in these reference records is limited, we would look at other data sources, such as university home pages, object-level search engines, LinkedIn, to harvest more information. Table 2 shows the information crawled from different sources, after transforming them into structured records. Note that records may contain ambiguous name representations and conflicting attribute values.*

*In order to get a more complete profile of each researcher, a typical solution consists of two main steps. First, compute the similarity between the records in Tables 1 and 2, and form three clusters:*

$c_1 = \{q_1, r_1, r_3, r_5\}$
$c_2 = \{q_2, r_6, r_7, r_9, r_{10}\}$
$c_3 = \{q_3, r_2, r_4, r_8\}$

*Then determine the correct attribute values within each cluster by majority vote, and construct the following profiles:*


*KDD'14,* August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623685.

**Table 1: Reference Records**

| | Name | Affiliation |
|---|---|---|
| $q_1$ | Rakesh Agrawal | MS |
| $q_2$ | Charu Aggarwal | IBM |
| $q_3$ | Alon Y. Halevy | Google |

**Table 2: Input Records from Various Data Sources**

| | Name | Affiliation | Field | Education | Source |
|---|---|---|---|---|---|
| $r_1$ | Rakesh Agrawal | *Bell* | DM | Wisconsin | $src_1$ |
| $r_2$ | Alon Halevy | Google | DB | Stanford | |
| $r_3$ | Rakesh Agrawal | MS | DM | | $src_2$ |
| $r_4$ | A. Halevy | Google | DB | | |
| $r_5$ | Agrawal | MS | | Wisconsin | $src_3$ |
| $r_6$ | Charu Aggarwal | IBM | | MIT | |
| $r_7$ | Agrawal | *IBM* | | Wisconsin | |
| $r_8$ | Halevy | *UW* | DB | Stanford | $src_4$ |
| $r_9$ | Charu Aggarwal | *UIC* | DM | MIT | |
| $r_{10}$ | Agrawal | IBM | DM | *Wisconsin* | $src_5$ |

True matchings:$\{q_1, r_1, r_3, r_5, r_7\}$,$\{q_2, r_6, r_9, r_{10}\}$,$\{q_3, r_2, r_4, r_8\}$

$p_1$ =<*Rakesh Agrawal, MS, DM, Wisconsin*>
$p_2$ =<*Charu Aggarwal, IBM, DM, MIT*>
$p_3$ =<*Alon Y. Halevy, Google, DB, Stanford*>

*However, a closer examination would reveal that records $r_8$ and $r_9$ which are published by source $src_4$ contain affiliations that differ from the corresponding profile records. This leads us to suspect that the affiliation information in record $r_7$ is highly likely to be incorrect since it is provided by the same source $src_4$.*

*On the other hand, the education information published by $src_4$ for $r_8$ and $r_9$ are both correct, giving us the confidence that the value "Wisconsin" in $r_7$ can be trusted. With this key insight on $src_4$ providing unreliable affiliation information, and more reliable education information, we could infer that $r_7$ is more likely to refer to "Rakesh Agrawal" rather than "Charu Aggarwal".*

The above example illustrates that rectifying errors in attribute values and taking into consideration the reliability of data sources can provide additional evidence for linking records, leading to a more complete and accurate profile of an entity. So how can we utilize this observation to improve the accuracy of data collation, and do that efficiently in practice?

First, we introduce the notion of a reliability matrix to capture the reliability of each source for various attributes. Second, we interleave the processes of record linkage and error correction so that they can benefit from each other: rectifying errors in attribute values will facilitate record comparison and linkage, while linking similar records will provide a clearer view of the data and additional evidence of any erroneous values.

The proposed framework COMET consists of two main phases. The first phase is a confidence based matching that links each input record to one or more reference records. This yields a soft clustering of records and reduces the search space for the next phase. Based on the clustering results, we obtain an initial assessment of the reliability of the sources. Then the second phase, adaptive matching, leverages on the source reliability to iteratively determine the correct attribute values within each cluster and eliminate unlikely matches for the input records.

Experiment results on real world multiple source datasets demonstrate that COMET can build more complete and accurate entity profiles efficiently, and outperforms the state-of-the-art techniques.

The rest of the paper is organized as follows. Section 2 summarizes the related work. Section 3 gives the problem definition. The COMET framework is presented in Section 4. Section 5 describes the experimental evaluation and we conclude in Section 6.

## 2. RELATED WORK

Constructing structured profiles of real world entities is a challenging task and has been addressed from various aspects. [17, 10] extract entity-related facts from web pages, which form the input records in our framework. [21] proposes a holistic approach to solve the schema matching problem of web tables. Our work builds upon these information extraction and schema matching techniques, and we focus on resolving the ambiguous references and erroneous values contained in the data records.

Record linkage, also known as entity resolution, aims to identify records that refer to the same real world entity [7, 15]. Techniques to match records can be categorized into learning-based algorithms and non-learning methods. Learning-based algorithms [1] train a classifier to label each record pair as match or unmatch, while non-learning methods [9, 8] utilize a set of rules to link records. Based on the pairwise matching results, clustering algorithms are used to find sets of records where each set refers to a unique entity [14, 3]. The works in [20, 18] examine the problem of linking records from multiple sources. [20] transforms the records into a graph to estimate the ambiguity of each source and finds an optimized matching execution plan. [18] adopts transfer learning to learn a classifier which can capture the common characteristics of all the sources as well as the specific characteristics of individual source pairs. All these methods are not aware of source reliabilities and their performance deteriorates when the data is noisy.

There is a line of research to resolve conflicts and find true values from data provided by different sources [13, 16]. These truth discovery algorithms usually adopt an iterative approach to let the data sources and attribute values vote for each other. The similarity between values [22] and the dependence between sources [6] can be taken into account to better model the complexities in the real world. Apart from these voting-based algorithms, the work in [19] constructs probabilistic models where the true values are regarded as latent variables, while [5] deduces the relative accuracy of attributes based on the available master data and a set of pre-defined rules. These works assume that record linkage has already been done and the input set of records refer to the same real world entity.

The work closest to ours, [12], proposes to link records with uniqueness constraints and erroneous values by modeling it as a k-partite graph clustering problem. This approach assumes the existence of a key attribute that can uniquely identify an entity. It is computationally expensive as it compares all the records in each iteration, and its recall rate suffers when the percentage of erroneous values increases. In contrast, the two-phase approach in our proposed frame-

work is efficient and remains robust when the percentage of errors and the degree of ambiguity vary.

# 3. DEFINITIONS AND NOTATIONS

In this section, we define the notations used as well as a formal definition of the problem addressed.

Let $\mathcal{E}$ be a set of real world *entities*. Each entity $e \in \mathcal{E}$ is described by a set of *attributes* $\mathcal{A}$.

Let $\mathcal{S}$ be a set of *data sources*. Each source $s \in \mathcal{S}$ may publish some information on a subset of the entities in $\mathcal{E}$.

Let $\mathcal{R}$ be a set of *input records* with attributes $\mathcal{A}$. Each input record $r \in \mathcal{R}$ is published by some source $s \in \mathcal{S}$, and refers to some entity $e \in \mathcal{E}$. A published record may have missing or erroneous attribute values. Note that the records from different sources have been mapped to a uniform schema.

Let $\mathcal{Q}$ be a set of *reference records* with attributes $\mathcal{A}_\mathcal{Q} \subset \mathcal{A}$. Each reference record $q \in \mathcal{Q}$ is known to be clean, and refers to some unique entity $e \in \mathcal{E}$.

Let $\mathcal{C}$ be a set of clusters. Each cluster $c \in \mathcal{C}$ comprises of a reference record $q \in \mathcal{Q}$ and a set of input records $\mathcal{R}_c \subset \mathcal{R}$. Each cluster $c$ has a signature $H_c = \{<a, v, pr> \mid \forall a \in \mathcal{A}\}$, where $pr$ is the probability for $v$ being the correct value on attribute $a$.

**Problem Definition.** Given a set of reference records $\mathcal{Q}$ and a set of input records $\mathcal{R}$ published by data sources $\mathcal{S}$, the goal is to augment the records in $\mathcal{Q}$ with the true values (if any) of the attributes $\mathcal{A} - \mathcal{A}_\mathcal{Q}$.

# 4. COMET FRAMEWORK

This section describes the proposed framework COMET that collates data from different sources, corrects any erroneous values, and constructs entity profiles. An overview of COMET is given in Figure 1. The framework is designed to facilitate robust record matching, and leverage on the knowledge from truth discovery to improve its performance. COMET has two main phases: (a) confidence based matching and (b) adaptive matching.

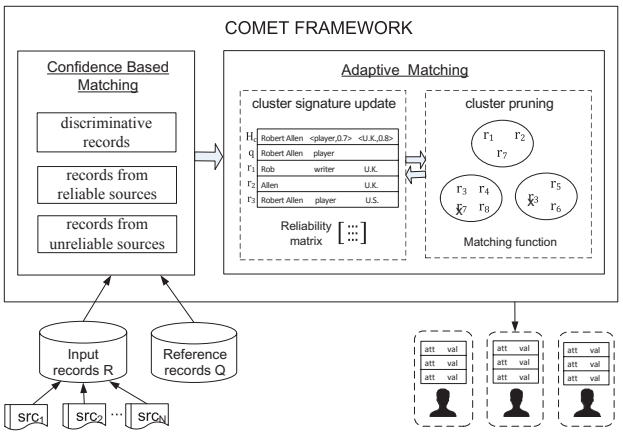



**Figure 1: Overview of COMET**

The first phase finds a set of entities that an input record may potentially describe by linking each input record to one or more reference records. Each reference record and its set of associated records form a cluster. At the same time, it initializes a matrix to capture the reliability of the data sources for various attributes.

The second phase aims to determine the correct attribute values for each entity. It first finds the most accurate attribute values within each cluster to form the cluster signature, and then updates the source reliability. Subsequently, clusters are refined by removing the records whose attribute values deviate significantly from the cluster signatures. This process is repeated until there is no change to the clusters.

Finally, we augment the reference record in each cluster with the values in cluster signature to construct the profile for the corresponding entity.

## 4.1 Confidence Based Matching

In this phase, we generate a set of candidate reference records for each input record. Due to the possible erroneous values contained in the records, we would need a relatively low similarity threshold so that the true matches are not missed. However, the low threshold will affect the quality of the resulting clusters, which will in turn impact the discovery of correct attribute values in the next phase (as shown in Section 5.3).

We overcome this dilemma by adopting a confidence based matching approach such that records which clearly reference some real world entity are linked first, while the decisions on difficult cases are postponed until we have gathered more information about the reliability of the data sources. This leads to a smaller set of candidate matches without sacrificing the recall.

We first create a cluster for each reference record $q$. Then we bootstrap the framework with a small set of *confident matches* between the input records $\mathcal{R}$ and reference records $\mathcal{Q}$. Specifically, we measure the similarity between records using any existing record linkage method, e.g., the Fellegi-Sunter algorithm [9], and link a record $r$ to a reference record $q$ if $r$ is highly similar to $q$ and differs greatly to the records in $\mathcal{Q} \setminus \{q\}$. In other words, we are confident that $r$ refers to $q$. We formalize this into the following definition.

*Definition 1.* Given two thresholds $\delta_H$ and $\delta_L$, where $\delta_H > \delta_L$, and a similarity function sim(), an input record $r$ and a reference record $q$ form a *confident match* if $\exists q \in \mathcal{Q}$ where $\text{sim}(r, q) > \delta_H$, and $\forall q' \in \mathcal{Q} \setminus \{q\}$, $\text{sim}(r, q') < \delta_L$. Then the record $r$ is called a *discriminative record*.

For each confident match, we compare the attribute values of record $r$ with its matching reference record $q$ to obtain an initial assessment of the reliability of the corresponding data source.

We define a *reliability matrix* $M$ where each entry $M[s, a]$ is the reliability of source $s$ on attribute $a$, and initialize $M$ as follows. Let $\mathcal{D}_s$ be the set of discriminative records published by $s$. We first process the sources that have published some discriminative records, that is, $\mathcal{D}_s \neq \emptyset$. Let $\mathcal{D}_s^a \subset \mathcal{D}_s$ be the set of records where both the attribute values $r.a$ and its matching $q.a$ are not null. Then we have

$$M[s, a] = \frac{1}{|\mathcal{D}_s^a|} \sum_{r \in \mathcal{D}_s^a} \text{sim}(r.a, q.a), \quad \text{if} \quad \mathcal{D}_s^a \neq \emptyset \qquad (1)$$

Then for each attribute $a$ where $\mathcal{D}_s^a = \emptyset$, we set its $M[s, a]$ to be the average of the non-null entries in $M[s]$. Finally, for the sources that have not published any discriminative record, we set their entries in $M$ to some small value $\epsilon$.

**Table 3: Clusters Obtained by Confidence Based Matching**

| | | | | | |
|---|---|---|---|---|---|
| $q_1$ | Rakesh Agrawal | MS | | | |
| $r_1$ | Rakesh Agrawal | *Bell* | DM | Wisconsin | $src_1$ |
| $r_3$ | Rakesh Agrawal | MS | DM | | $src_2$ |
| $r_5$ | Agrawal | MS | | Wisconsin | $src_3$ |
| $r_7$ | Agrawal | *IBM* | | Wisconsin | $src_4$ |
| $r_{10}$ | Agrawal | IBM | DM | *Wisconsin* | $src_5$ |

(a) Cluster $c_1$

| | | | | | |
|---|---|---|---|---|---|
| $q_2$ | Charu Aggarwal | IBM | | | |
| $r_6$ | Charu Aggarwal | IBM | | MIT | $src_3$ |
| $r_9$ | Charu Aggarwal | *UIC* | DM | MIT | $src_4$ |
| $r_5$ | Agrawal | MS | | Wisconsin | $src_3$ |
| $r_7$ | Agrawal | *IBM* | | Wisconsin | $src_4$ |
| $r_{10}$ | Agrawal | IBM | DM | *Wisconsin* | $src_5$ |

(b) Cluster $c_2$

| | | | | | |
|---|---|---|---|---|---|
| $q_3$ | Alon Y. Halevy | Google | | | |
| $r_2$ | Alon Halevy | Google | DB | Stanford | $src_1$ |
| $r_4$ | A. Halevy | Google | DB | | $src_2$ |
| $r_8$ | Halevy | *UW* | DB | Stanford | $src_4$ |

(c) Cluster $c_3$

EXAMPLE 2. *Consider the reference records in Table 1 and the input records in Table 2. Suppose the set of confident matches are* $\{(r_1, q_1), (r_3, q_1), (r_6, q_2), (r_9, q_2), (r_2, q_3)\}$. *We compute the reliability of the five sources on attribute Affiliation by comparing these record pairs. Based on the reference records, we observe that records* $r_1$ *and* $r_9$ *provide wrong affiliations, while the others are all correct. Then by Equation 1 we have*

$M[src_1, \text{Affiliation}] = 0.5,$
$M[src_2, \text{Affiliation}] = M[src_3, \text{Affiliation}] = 1.0,$
$M[src_4, \text{Affiliation}] = 0.2,$
$M[src_5, \text{Affiliation}] = \epsilon.$

After initializing the reliability matrix, we want to *distinguish* records that originate from the sources which are significantly more unreliable than the others. If we consider the source reliability[1] as a random variable $X$, then we say a source $s$ is *unreliable* if

$$\frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} M[s, a] < \mu - \sigma \tag{2}$$

where $\mu$ and $\sigma$ is the mean and standard deviation of $X$. Otherwise, we consider the source as *reliable*.

For each record $r$ published by a reliable source, we use all the attribute values to compare $r$ with the reference records $\mathcal{Q}$. We link $r$ with a $q \in \mathcal{Q}$ if the similarity between them exceeds a pre-defined threshold $\delta$.

Finally, we process the records from less reliable sources. For these records, since their attribute values are expected to be more error-prone, we may miss the correct matchings if we simply compare the attribute values. As such, we use the name references, and compare each record $r$ with all the records in each cluster on name. We add $r$ to a cluster if it is similar to some record in that cluster. This approach is similar to the merge closure described in [2].

[1]This is the average reliability of a source on all the attributes.

**Algorithm 1:** Confidence Based Matching

**input** : Input records $\mathcal{R}$ from data sources $\mathcal{S}$ with attributes $\mathcal{A}$, reference records $\mathcal{Q}$, threshold $\delta$
**output**: Set of clusters $\mathcal{C}$, reliability matrix $M$

```
1  foreach q ∈ Q do
2      create cluster c containing q;
3      add c to C;
   /* confident matches */
4  foreach r ∈ R do
5      if r and q form a confident match then
6          add r to the cluster of q;
   /* initialize reliability matrix M */
7  let D_s be the set of discriminative records from s;
8  foreach s ∈ S where D_s ≠ ∅ do
9      let D_s^a ⊂ D_s be the set of records where
       r.a ≠ null ∧ q.a ≠ null;
10     foreach a ∈ A where D_s^a ≠ ∅ do
11         set M[s, a] using Equation 1;
12     foreach a ∈ A where D_s^a = ∅ do
13         let n be the number of non-null entries in M[s];
14         M[s, a] = (1/n) Σ_{a∈A} M[s, a];
15 foreach s ∈ S where D_s = ∅ do
16     foreach a ∈ A do
17         M[s, a] = ε;
   /* records from reliable sources */
18 foreach unprocessed r ∈ R do
19     if Equation 2 does not hold then
20         foreach q ∈ Q do
21             if sim(r, q) > δ then
22                 add r to the cluster of q;
   /* records from unreliable sources */
23 foreach unprocessed r ∈ R do
24     foreach c ∈ C do
25         foreach r' in cluster c do
26             if sim(r.N, r'.N) > δ then
27                 add r to cluster c;
28                 break;
```

EXAMPLE 3. *Table 3 shows the three clusters obtained by the confidence based matching phase. Discriminative records* $r_1$ *and* $r_3$ *are assigned to cluster* $c_1$, $r_6$ *and* $r_9$ *are put in* $c_2$, *while* $r_2$ *is placed in* $c_3$. *We next compare these records with their associated reference records to obtain the source reliabilities . Sources* $src_4$ *and* $src_5$ *are found to be unreliable while the others are reliable. Thus records* $r_4$ *and* $r_5$ *are processed first. Record* $r_4$ *is put in cluster* $c_3$, *while* $r_5$ *is placed in both* $c_1$ *and* $c_2$ *as it is similar to both* $q_1$ *and* $q_2$.

*For the remaining records* $r_7$, $r_8$ *and* $r_{10}$ *published by unreliable sources (*$src_4$ *and* $src_5$*), we compare them with all the records in each cluster on attribute Name. Records* $r_7$ *and* $r_{10}$ *have identical name with* $r_5$ *and are assigned to clusters* $c_1$ *and* $c_2$; $r_8$ *is highly similar to* $r_4$ *and is put in* $c_3$.

Algorithm 1 gives the details of this confidence based matching phase. We first create a cluster for each reference record (lines 1-3). Then we establish the confident matches between reference records and input records (lines 4-6). Lines 7-17 initialize the reliability matrix. Based on the source reliabilities, we process records that originate from the reli-

able sources (lines 19-22) before those from the less reliable sources (lines 23-28). The output is a set of clusters. Note that an input record may not be placed in any cluster if it cannot be linked to any reference record.

## 4.2 Adaptive Matching

After clustering the records, the next phase iteratively refines the clusters by interleaving truth discovery with record matching. There are three main steps in this adaptive matching phase: (a) compute the cluster signatures, (b) update the reliability matrix and (c) refine the clusters.

First, for each cluster $c$ obtained in the previous phase, we build its signature $H_c$ based on the accuracy of the attribute values of the records in $c$. The reliability matrix is updated subsequently. Then each record $r$ is compared with the signatures of the clusters it is associated with, and pruned from the cluster that it is the most dissimilar to. The above steps are repeated until there is no change to the clusters.

In order to form a cluster signature, we need to assign an accuracy score to each attribute value in the cluster. Intuitively, a value published by more reliable sources tends to be more accurate, while a source that provides more accurate values tends to be more reliable. Thus we can let the values and the sources vote for each other. However, since our framework allows records to belong to multiple clusters, these records may skew the value accuracy and source reliability computations with repeated votes. Therefore a likelihood function is introduced to address the issue.

Let $L(r, c)$ be the likelihood of a record $r$ belonging to a cluster $c$, and $q$ be the reference record in $c$. $L(r, c)$ is initialized as follows:

$$L(r, c) = \text{sim}(r, q) \tag{3}$$

We now discuss how we use this likelihood function to compute the cluster signatures and update the reliability matrix.

Let $\mathcal{V}_c^a$ be the set of values on attribute $a$ within cluster $c$. The accuracy of a value $v \in \mathcal{V}_c^a$ is given by the sum of the reliabilities of its sources, weighted by the likelihood:

$$acc(a, v, c) = \sum_{r \in c, r.a = v} L(r, c) \cdot M[s_r, a] \tag{4}$$

where $s_r$ is the source that publishes $r$. The value accuracies are normalized such that they represent the probabilities of the values being true.

The signature $H_c$ for a cluster $c$ contains a triplet $<a, v, pr>$ for each $a \in \mathcal{A}$ such that

$$<a, v, pr> = \begin{cases} <a,\ q.a,\ 1.0> & \text{if } q.a \neq null \\ <a,\ v_m,\ acc(a, v_m, c)> & \text{otherwise} \end{cases} \tag{5}$$

where $v_m = \arg\max_{v \in \mathcal{V}_c^a} acc(a, v, c)$.

Example 4. *Let us calculate the accuracy of the two Education values "MIT" and "Wisconsin" in cluster $c_2$. Suppose all the sources are equally reliable with a score of* 0.8*, and $L(r_5, c_2) = L(r_{10}, c_2) = L(r_7, c_2) = 0.5$. Then the accuracy of "MIT" and "Wisconsion" are* 1.6 *and* 1.2 *respectively. We normalize them into probabilities and insert the triplet* <Education*, "MIT",* 0.6> *into the cluster signature of $c_2$.*

After obtaining the signatures for all the clusters in $\mathcal{C}$, we update the reliability of a source $s$ for attribute $a$ as the average accuracy of the values it publishes:

$$M[s, a] = \frac{1}{|\mathcal{R}_s|} \sum_{r \in \mathcal{R}_s} \sum_{c \in \mathcal{C}_r} L(r, c) \cdot acc(a, r.a, c) \tag{6}$$

where $\mathcal{R}_s$ is the set of records published by $s$, and $\mathcal{C}_r$ is the set of clusters that contain $r$.

Note that a value may have different accuracy scores in different clusters. Hence for a value $r.a$, we calculate its accuracy as the sum of all its accuracy scores weighted by the likelihood.

Example 5. *Consider the entry $M[src_4, \text{Affiliation}]$. Suppose we have determined acc*(Affiliation*,"IBM",$c_1$) = 0.4 and acc*(Affiliation*,"IBM",$c_2$) = 0.6. Then the accuracy of the Affiliation value in $r_7$ is given by* $0.5 \times 0.4 + 0.5 \times 0.6 = 0.5$. *We repeat the computation for the other records published by $src_4$, which are $r_8$ and $r_9$, and obtain* 0.3 *and* 0.2 *respectively. Then $M[src_4, \text{Affiliation}] = \frac{0.5+0.3+0.2}{3} = 0.33$.*

Once we have computed the cluster signatures and updated the reliability matrix, we prune the clusters by removing records that are unlikely to belong to them. While a record is unlikely to belong to a cluster if its similarity with the cluster signature is low, however, a direct comparison of the record with a cluster signature may lead to incorrect pruning due to the possible erroneous values in the record. As such, we incorporate the reliability matrix to adaptively lower the impact of inaccurate attributes on the matching decisions.

Given a record $r$ and a cluster $c$, we define a matching function match$(r, c)$ as follows:

$$\text{match}(r, c) = \frac{\sum_{a \in \mathcal{A}} M[s_r, a] \cdot \text{sim}(r.a, H_c.a)}{\sum_{a \in \mathcal{A}} M[s_r, a]} \tag{7}$$

where $\text{sim}(r.a, H_c.a)$ is the similarity between the values in record $r$ and signature $H_c$ on attribute $a$. Note that this similarity may be weighted.

For each record $r$ that associated with multiple clusters, we compute its match scores with all the clusters in $\mathcal{C}_r$ using Equation 7, and remove $r$ from the cluster with the lowest match score.

Example 6. *Consider record $r_7$ which is associated with clusters $c_1$ and $c_2$. We have determined*

$M[src_4, \text{Affiliation}] = 0.33$, $M[src_4, \text{Education}] = 0.8$.

*Further, we have*

<Affiliation*,"MS"*,1.0>, <Education*,"Wisconsin"*,1.0> *in $H_{c_1}$,*

<Affiliation*,"IBM"*,1.0>, <Education*,"MIT"*,0.6> *in $H_{c_2}$.*

*We compare $r_7$ with both $H_{c_1}$ and $H_{c_2}$ to calculate the match scores by Equation 7. Clearly, the match score with cluster $c_1$ is higher, so we remove $r_7$ from $c_2$.*

Finally, we update the likelihood $L(r, c)$ for the record $r$ w.r.t. each cluster $c \in \mathcal{C}_r$ as follows:

$$L(r, c) = \frac{\text{match}(r, c)}{\sum_{c' \in \mathcal{C}_r} \text{match}(r, c')} \tag{8}$$

Algorithm 2 shows the details of this adaptive matching phase. We first initialize the likelihood for each record in the clusters (lines 1-4). Then we repeat the steps of cluster signature computation (lines 7-11), reliability matrix update (lines 12-13) and cluster pruning (lines 14-21) until there is no more change to the clusters.

**Algorithm 2:** Adaptive Matching

**input** : Reliability matrix $\mathcal{M}$, set of clusters $\mathcal{C}$
**output**: Set of refined clusters $\mathcal{C}$

```
1  foreach r ∈ R do
2      Let C_r be the set of clusters that contain r;
3      foreach c ∈ C_r do
4          initialize L(r, c) using Equation 3;
5  repeat
       /* cluster signature and reliability matrix update */
6      foreach a ∈ A do
7          foreach c ∈ C do
8              foreach v ∈ V_c^a do
9                  calculate acc(a, v, c) using Equation 4;
10             normalization;
11             update cluster signature using Equation 5;
12         foreach s ∈ S do
13             update M[s, a] using Equation 6;
       /* cluster pruning */
14     foreach r ∈ R do
15         if |C_r| > 1 then
16             foreach c ∈ C_r do
17                 calculate match(r, c) using Equation 7;
18             c ← arg min_{c∈C_r} match(r, c);
19             remove r from c;
20         foreach c ∈ C_r do
21             update L(r, c) using Equation 8;
22 until there is no change to the clusters C;
```

## 4.3 Time Complexity

Let $n$ be the number of input records $\mathcal{R}$, $m$ be the number of reference records $\mathcal{Q}$, and $p$ be the number of non-discriminative records from unreliable sources.

In the confidence based matching phase, the time complexity for matching the discriminative records and records from reliable sources are both $O(nm)$, as they compare record pairs from $\mathcal{R}$ and $\mathcal{Q}$. In contrast, matching the records from unreliable sources requires comparing each record to every other record in all the clusters. Let $k$ is the largest number of clusters that a record is associated with. Then this step requires a time complexity of $O(knp)$. Hence the time complexity of the confidence based matching phase is $O(nm + knp)$.

For the adaptive matching phase, each iteration involves cluster signature computation and cluster pruning. Each of these steps has a complexity of $O(kn)$. Since this phase takes $k$ iterations to terminate, the time complexity is $O(k^2 n)$.

In practice, $k$ is usually a small constant. Hence the complexity of COMET is $O(nm + np)$.

# 5. PERFORMANCE STUDY

In this section, we present the results of extensive experiments to evaluate the performance of COMET.

**Datasets**. We use two real world multiple source datasets:

- *Restaurant dataset.*
  We construct a Restaurant dataset by crawling 6 websites for restaurants with zip code 78701. 1082 records on 384 restaurants are obtained. Each record has the attributes Name, Address, Phone, Website, etc. We pre-process this dataset by removing the restaurants that do not have inconsistent values from multiple websites. Finally, we obtain 581 records on 222 restaurants with 380 name values. The ratio of erroneous values to the total number of attribute values is 18.7%. About 9.1% of the records have ambiguous names (a name is ambiguous if there are at least two records with this name but refer to different entities).
  We extract information from *www.yellowpages.com* as true attribute values, and form a reference table consisting of 478 records with attributes Name and Phone. We manually determine the true matchings between records.
- *Football dataset.*
  The Football data in [4] contains 7492 records on the biodata of football players from 20 websites. Each record contains attributes Name, Birthday, Height, Weight, Position and BirthPlace. The ground truth of the attribute values were collected from official websites of the players or the corresponding football clubs. We pre-process the data by removing the records with less than 3 non-null attribute values. Finally, we have 5031 records on 976 players from 12 websites. The ratio of erroneous values to the total number of attribute values is 32.7%.
  We extract information from *Wikipedia* to get biodata on 12419 players and form reference table with attributes Name, Birthday and BirthPlace.

Based on the Football dataset, we generate a series of datasets by varying the accuracy of attribute values and the ambiguity of name references. We vary the following parameters to obtain different datasets.

- $\%err$, the ratio of erroneous values to the total number of attribute values;
- $\%ambi$, the probability to abbreviate the name reference of a record[2].

Table 4 shows the default values and the ranges for these parameters.

**Table 4: Parameter Settings**

| Parameter | Default | Range |
|---|---|---|
| $\%err$ | 0.33 | 0.1-0.5 |
| $\%ambi$ | 0.5 | 0.4-0.9 |

When varying $\%err$, if the desired error rate is higher than that of the original data, we introduce errors to the attribute values; otherwise, we correct some errors contained in the original data. We generate errors as follows: for attribute Birthday, we randomly generate an erroneous value; for the other 3 attributes, we randomly select a value from the domain of that attribute. The erroneous values of the generated data have the same distribution as the original data, that is, the relative accuracy among different sources and attributes remain unchanged.

When varying $\%ambi$, we abbreviate the names of the records in the original data and keep the other attributes unchanged. We use one of the following forms of abbreviation with equal probability: remove first name; remove last

[2]The original dataset provides the full name of each player.

name; keep first name initial and last name; remove last name and the last letter of first name.

**Methods.** We compare the following methods in our performance study:

- MATCH [12]. This is the state-of-the-art method that combines record linkage and truth discovery with uniqueness constraints. For the Football dataset, we concatenate the non-unique attributes, Height and Weight, with Name to form a super-identifier. For the Restaurant dataset, we combine Name and Address to form a multi-attribute identifier.
- PIPELINE. This is the baseline method that performs record linkage and truth discovery sequentially. We use TRANSFER [18], a learning-based method designed to link records from multiple sources, to perform record linkage. Learning-based methods have been shown to provide superior performance [15]. For truth discovery, we employ TRUTHFINDER [22] as the representative method among those surveyed in [16]. We transform the pairwise results of TRANSFER into clusters using the techniques in [11] before calling TRUTHFINDER. To improve performance, records from each pair of data sources are compared, instead of just comparing input records with reference records.
- COMET, the proposed framework. To be fair, we also utilize the TRANSFER method to establish confident matches in the first matching phase.

Note that the reference records are used in both MATCH and PIPELINE where the attribute values in these records are regarded as true values. The same blocking criteria is applied to all the methods to reduce the number of record comparisons. TF-IDF metric is used to measure the similarity of strings, where each token is measured by Jaro-Winkler distance. Levenshtein distance is used for Birthday and Phone. We set $\delta = 0.8$, $\delta_H = 0.95$ and $\delta_L = 0.65$ for COMET.

All the algorithms were implemented in Java, and the experiments were conducted in a Windows 7 machine with 3.40 GHz Intel CPU and 8 GB of RAM. Each experiment was repeated 3 times and the average performance is reported.

## 5.1 Experiments on Record Linkage

We first evaluate whether each input record can be linked to the correct reference record. The evaluation metrics include *Precision* and *Recall*.

Let *Ground* be the set of correct matchings between input records and reference records, and *Result* be the set of matchings returned from the algorithms. Then we have

$$Precision = \frac{|Ground \cap Result|}{|Result|}$$

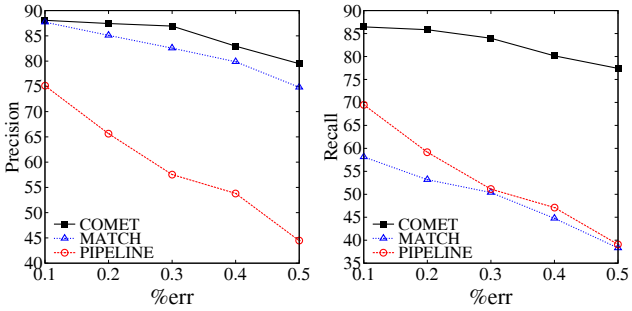
$$Recall = \frac{|Ground \cap Result|}{|Ground|}$$

Table 5 shows the performance of various methods on the Restaurant dataset. We observe that both COMET and MATCH outperform PIPELINE, demonstrating that combining record linkage with truth discovery can lead to more robust record comparisons. COMET gives the best precision and recall since it leverages on the more reliable attributes to reduce the impact of erroneous values.

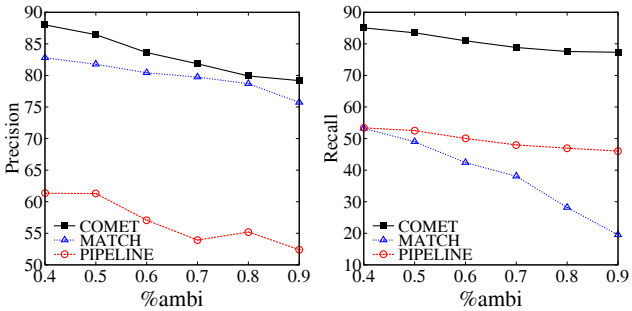
PIPELINE fails to link a considerable amount of records. We found that 57% of these records contain wrong phone numbers, yet the TRANSFER model still assigns a high weight to this attribute. This indicates that the learning-based record linkage methods do not perform well in the presence of erroneous values. In contrast, COMET explicitly considers the source reliabilities and improves the performance significantly.

**Table 5: Record Linkage on Restaurant Dataset**

| | Precision | Recall |
|---|---|---|
| COMET | **96.6** | **96.6** |
| MATCH | 93.0 | 88.1 |
| PIPELINE | 89.1 | 83.5 |




(a) Vary %err




(b) Vary %ambi

**Figure 2: Record linkage on Football dataset**

Figure 2 shows the results on the Football dataset as we vary *%err* and *%ambi*. We observe that the performance of MATCH and PIPELINE drop significantly as the percentage of errors increases. However, COMET remains robust and outperforms the others. Even when $\%err = 0.5$, COMET still achieves a recall of 0.77, which is 97% higher than the second best. This demonstrates its ability to correctly link records in the presence of erroneous values. Compared to MATCH, PIPELINE is less sensitive to *%ambi* but more sensitive to *%err*. That is because the various ambiguities can be learnt more effectively.

## 5.2 Experiments on Truth Discovery

Next, we evaluate whether COMET constructs an accurate and complete profile for each entity. The evaluation metrics are *Accuracy* and *Coverage*.

Let $TrueValue$ be the set of attribute values in the ground truth that contained in the input records, and $ReturnValue$

Table 6: Truth Discovery on Restaurant Dataset

| | Accuracy | Coverage |
|---|---|---|
| COMET | **86.4** | **83.2** |
| MATCH | 75.3 | 76.8 |
| PIPELINE | 82.3 | 71.2 |

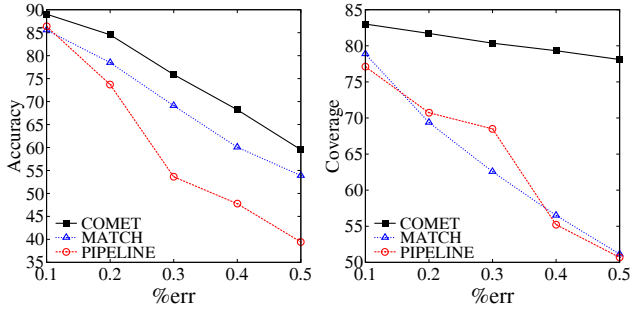
be the set of attribute values returned by the methods and are not present in the reference records. Then we define

$$Accuracy = \frac{|TrueValue \cap ReturnValue|}{|ReturnValue|}$$

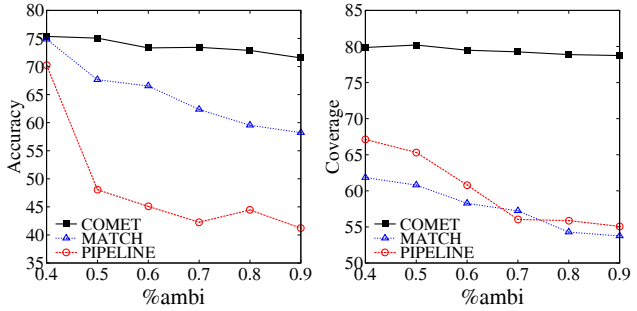
$$Coverage = \frac{|TrueValue \cap ReturnValue|}{|TrueValue|}$$

Table 6 shows the results on Restaurant dataset. We observe that COMET outperforms MATCH by 15% on accuracy and 8% on coverage. One of the reasons for the low accuracy of MATCH is that it is unaware of the source reliabilities. PIPELINE gives the lowest coverage since many input records are not linked to any reference record. As a result, it fails to find all the attribute values of an entity.

Figure 3 shows the results on Football dataset as we vary *%err* and *%ambi*. Again, both COMET and MATCH outperform PIPELINE, indicating the advantage of integrating record linkage with truth discovery. However, MATCH is highly sensitive to the percentage of erroneous values. We observe that COMET improves the coverage significantly when *%err* and *%ambi* become large. Even when *%err* = 0.5, COMET can still recover 78% of the correct attribute values. This contributes to its robust record linkage. Overall, COMET consistently constructs both complete and accurate profiles for the entities when *%err* and *%ambi* vary.




(a) Vary %err




(b) Vary %ambi

**Figure 3: Truth discovery on Football dataset**

## 5.3 Sensitivity Experiments

In this section, we study the impact of the two main components in the COMET framework, namely, confidence based matching and adaptive matching. We use the Football dataset for this set of experiments and implement the following variants of COMET:

- COM-CM. This method does not utilize confidence based matching. In other words, when we form the initial clustering, records are processed in an arbitrary order and linked to a reference record if their similarity exceeds the threshold $\delta$.
- COM-AM. This method does not utilize adaptive matching. In other words, neither the reliability matrix nor the likelihood function is used to refine the clusters.

To evaluate the effect of confidence based matching, we measure the recall of the matchings established in the first phase of the framework, as the goal of this phase is to achieve a high recall.

Figure 4 shows the recall of record linkage after the first phase as we vary *%err* and *%ambi*. We observe that the confidence based matching has more impact when *%err* increases. It discriminates records from unreliable sources and thus significantly improves the recall. When *%ambi* becomes higher, by postponing the decisions to link the ambiguous records, COMET remains robust.

To evaluate the effect of adaptive matching, we measure the error rate of the cluster pruning process. We define error rate as the ratio of the number of true matchings being pruned to the total number of pruned matchings. The main purpose of the adaptive matching is to reduce errors in cluster pruning.

Figure 5 shows the error rate of the cluster pruning when we vary *%err* and *%ambi*. When *%err* increases, adaptive

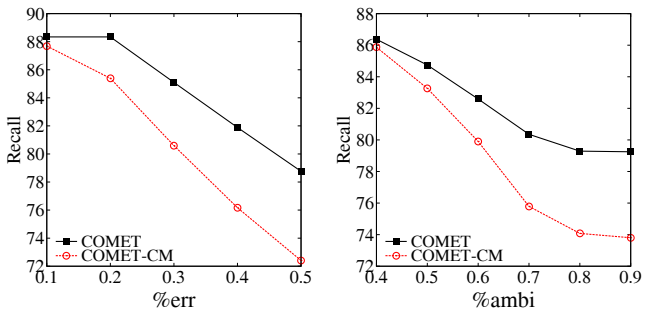



**Figure 4: Effect of confidence based matching**

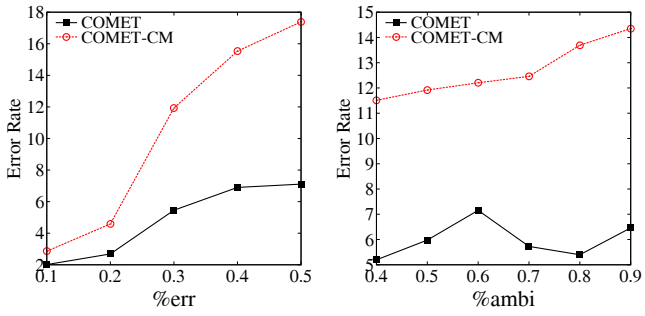



**Figure 5: Effect of adaptive matching**

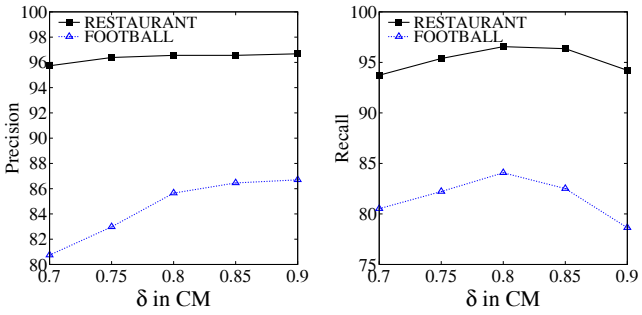
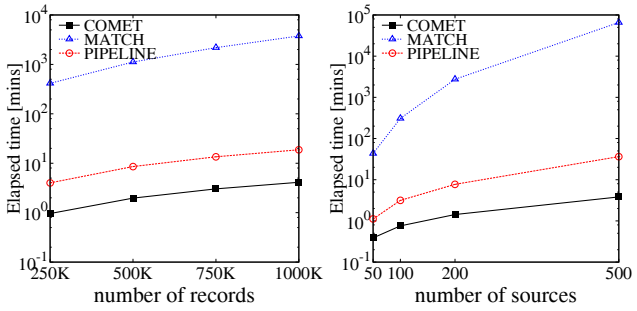



**Figure 6: Effect of similarity threshold $\delta$**




(a) Vary number of entities (b) Vary number of sources

**Figure 7: Scalability results**

matching avoids up to 59% of the errors by incorporating the reliability matrix to reduce the impact of erroneous values on matching decisions. We observe the gap between COMET and COM-AM widens as *%ambi* increases. This is because a record will be associated with more clusters, while the likelihood function in COMET is able to alleviate the bias brought by these records.

We also examine the effect of the similarity threshold $\delta$ in confidence based matching of COMET. We vary $\delta$ from 0.7 to 0.9 and report the performance of record linkage on both the Restaurant and Football datasets in Figure 6. In both datasets, we obtain the best results at 0.8. A lower $\delta$ generates a larger set of candidates for each input record, which hurts the purity of the resultant clusters. Then the cluster signatures computed in the subsequent phase will be less accurate, leading to incorrect pruning. On the other hand, if $\delta$ is too high, correct matchings may be missed in the first phase, resulting in lower recall. We observe similar behaviour when varying $\delta_H$ and $\delta_L$ and we omit the graphs due to lack of space.

## 5.4 Scalability Experiments

We also carried out experiments to compare the scalability of COMET, MATCH and PIPELINE. We generate synthetic records using the Football dataset schema by varying the number of entities and number of data sources. Each source will publish a record on each entity. We partition the records and process them in parallel using a cluster with 10 nodes.

We first fix the number of sources at 100 and vary the number of entities from 2,500 to 10,000, thus increasing the number of records from 250k to 1 million. Figure 7(a) shows that the runtime of all the methods increase almost linearly, since the cluster size for each entity remains the same.

Figure 7(b) shows the runtime as we fix the number of entities at 2,000 and increase the number of sources from 50 to 500. We see that COMET remains scalable, while MATCH becomes computationally expensive when the number of sources is large. This is because there will be more nodes in the encoded graph and its complexity is quadratic to the number of nodes. Further, MATCH will require more iterations to terminate.

## 5.5 Case Studies

Finally, we demonstrate the effectiveness of COMET by case studies. Table 7 shows a sample of the reference records extracted from *www.yellowpages.com*. Each reference record has two attributes: Name and Phone.

**Table 7: Sample of Reference Records**

| Name | Phone |
|---|---|
| Frank Restaurant | 512-494-6894 |
| Frank & Angie's Pizzeria | 512-472-3534 |

Table 8 shows a subset of the input records obtained from 6 websites. We transform the crawled data into a uniform schema, standardize attributes such as Address and Phone, and map range values to categorical values. We observe that the input records have ambiguous names and each provides a subset of attributes. Further, incorrect values are found in the address, phone, opening hours, etc.

Table 9 shows the profile records of two restaurants constructed by COMET after collating data records from the various sources and correcting the erroneous values. We see that each profile record contains more complete information than any of the input record in Table 8 and is more accurate. Further, the prices and ratings in the profiles reduce the bias found in the different websites. Interestingly, we notice that *TripAdvisor* provides more reliable rating information, while *Foursquare* and *Urbanspoon* tend to provide higher ratings.

# 6. CONCLUSION AND FUTURE WORK

In this paper, we have addressed the problem of building entity profiles by collating data records from multiple sources in the presence of erroneous values. We have designed a framework called COMET that interleaves record linkage with truth discovery to facilitate robust record matching and takes into account the reliability of the data sources to improve performance. The proposed framework first generates a soft clustering of records, and then iteratively determines the cluster signatures and refines the clusters. Extensive experimental results show that our approach significantly outperforms the state-of-the-art techniques on record linkage and truth discovery, and is both scalable and effective. Case studies demonstrate that the proposed framework can construct entity profiles that are both complete and accurate.

In the future, we plan to integrate the information extraction and schema matching processes into the framework, to provide a holistic approach. We believe that taking into account the source reliabilities can improve the performance of them, and an integrated approach will lead to more effective knowledge discovery.

**Table 8: Sample Input Records from Various Websites**

| Name | Address | Phone | Cuisine | Recommend | Price | Weekday Hours | Weekend Hours | Rating | Source |
|---|---|---|---|---|---|---|---|---|---|
| Frank | 407 Colorado St | 512-494-6894 | | Hot dog | | Normal[1] | Extend[2] | 8.7 | Urbanspoon[4] |
| Frank | 407 Colorado St | 512-494-6916 | | | $ | Normal | Normal | 6.0 | Find Me GF[5] |
| Frank Restaurant | btw 4th & 5th | 512-494-6894 | | | $ | Normal | Extend | 9.4 | Foursquare[6] |
| Frank | 407 Colorado St | 512-494-6916 | | | $ | | | | LocalEats[7] |
| Frank | 407 Colorado St | 512-494-6894 | | | $$ | Normal | Extend | 8.2 | Yelp[8] |
| Frank | 407 Colorado St | 512-494-6894 | American | Hot dog | $ | | | 8.2 | TripAdvisor[9] |
| Frank&Angie's Pizzeria | 508 West Ave | 512-472-3534 | Italian | Pizza | $ | Normal | Night[3] | 8.8 | Urbanspoon |
| Frank & Angie's | 508 West Ave | 512-472-3534 | | Pizza | | Normal | Night | 8.6 | Foursquare |
| Frank&Angie's Pizzeria | 508 West Ave | 512-472-3524 | | | $$ | Normal | Night | | LocalEats |
| Frank & Angie's | 508 West Ave | 512-472-3534 | | | $$ | Normal | Night | 7.0 | Yelp |
| Frank&Angie's Pizzeria | 508 West Ave | 512-472-3534 | Italian | Pizza | $$ | | | 8.0 | Tripadvisor |

[1]Normal: 10am-10pm. [2]Extend: 10am-2am. [3]Night: 5pm-10pm.
[4]www.urbanspoon.com [5]www.findmeglutenfree.com [6]www.foursquare.com
[7]www.localeats.com [8]www.yelp.com [9]www.tripadvisor.com

**Table 9: Profile Records Constructed for the Two Restaurants in Table 7**

| Name | Address | Phone | Cuisine | Recommend | Price | Weekday Hours | Weekend Hours | Rating |
|---|---|---|---|---|---|---|---|---|
| Frank Restaurant | 407 Colorado St | 512-494-6894 | American | Hot dog | $ | Normal | Extend | 8.2 |
| Frank & Angie's Pizzeria | 508 West Ave | 512-472-3524 | Italian | Pizza | $$ | Normal | Night | 8.0 |

# 7. REFERENCES


[1] K. Bellare, S. Iyengar, A. G. Parameswaran, and V. Rastogi. Active sampling for entity matching. In *KDD*, 2012.

[2] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom. Swoosh: a generic approach to entity resolution. *The VLDB Journal*, pages 255–276, 2009.

[3] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, page 5, 2007.

[4] L. Blanco, V. Crescenzi, P. Merialdo, and P. Papotti. Web data reconciliation: Models and experiences. In *Search Computing*, pages 1–15. 2012.

[5] Y. Cao, W. Fan, and W. Yu. Determining the relative accuracy of attributes. In *SIGMOD*, 2013.

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

[7] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, pages 1–16, 2007.

[8] W. Fan, H. Gao, X. Jia, J. Li, and S. Ma. Dynamic constraints for record matching. *The VLDB Journal*, pages 495–520, 2011.

[9] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, pages 1183–1210, 1969.

[10] N. Garera and D. Yarowsky. Structural, transitive and latent models for biographic fact extraction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 2009.

[11] J. Gemmell, B. I. Rubinstein, and A. K. Chandra. Improving entity resolution with global constraints. *Microsoft Research Technical Report*, 2011.

[12] S. Guo, X. L. Dong, D. Srivastava, and R. Zajac. Record linkage with uniqueness constraints and erroneous values. *VLDB*, 2010.

[13] M. Gupta and J. Han. Heterogeneous network-based trust analysis: a survey. *ACM SIGKDD Explorations Newsletter*, pages 54–71, 2011.

[14] O. Hassanzadeh, F. Chiang, H. C. Lee, and R. J. Miller. Framework for evaluating clustering algorithms in duplicate detection. *VLDB*, 2009.

[15] H. Köpcke, A. Thor, and E. Rahm. Evaluation of entity resolution approaches on real-world match problems. *VLDB*, 2010.

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

[17] X. Liu, Z. Nie, N. Yu, and J.-R. Wen. Biosnowball: automated population of wikis. In *KDD*, 2010.

[18] S. N. Negahban, B. I. Rubinstein, and J. G. Gemmell. Scaling multiple-source entity resolution using statistically efficient transfer learning. In *CIKM*, 2012.

[19] J. Pasternack and D. Roth. Latent credibility analysis. In *WWW*, 2013.

[20] W. Shen, P. DeRose, L. Vu, A. Doan, and R. Ramakrishnan. Source-aware entity matching: A compositional approach. In *ICDE*, 2007.

[21] M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *SIGMOD*, 2012.

[22] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. In *KDD*, 2007.