Effective and Efficient Data Cleaning for Entity Matching

Jing Ao
North Carolina State University
jao@ncsu.edu

Rada Chirkova
North Carolina State University
rychirko@ncsu.edu

ABSTRACT

As a key data-integration step, entity matching (EM) identifies tuples referring to the same real-world entities in disparate data sources. In many cases, the EM quality can be improved by repairing incorrect values in the data; at the same time, it is well known that the time costs of data cleaning by human experts could be prohibitive. In this paper, we focus on the time-consuming human-in-the-loop data-cleaning problem for relational EM, by recommending to human experts a time-efficient order in which values of attributes could be cleaned in the given data. Our proposed domain-independent cleaning framework aims to save human users' time, by guiding them in cleaning the EM inputs in an attribute order that is as conducive to maximizing EM accuracy as possible within a given constraint on the time they spend on cleaning. In guiding the cleaning process, our attribute-recommendation methods discover and take advantage of information provided by the data, and also use feedback from the EM engine. Our preliminary experimental results suggest that the proposed approach leads to measurable speedup, for a variety of time constraints, in the improvement of EM accuracy over the baseline approach, in which domain experts choose the sequence in which to clean the attributes of the inputs.

CCS CONCEPTS

• Information systems → Data cleaning; **Entity resolution**; **• Human-centered computing → Interactive systems and tools**.

KEYWORDS

Entity matching, data cleaning, feature.

ACM Reference Format:
Jing Ao and Rada Chirkova. 2019. Effective and Efficient Data Cleaning for Entity Matching. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'19), July 5, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328519.3329127

1 INTRODUCTION

Entity Matching *(EM)* identifies tuples that refer to the same real-world entities in disparate data sources. It is a key data-integration step that can contribute significantly to improving data quality, and thus to facilitating downstream data analysis and decision making [6, 11, 13–15]. Incorrect values in the input data can severely harm the quality of EM outputs, thus potentially leading to unreliable analysis results. Due to its importance, the problem of data cleaning has received significant attention in the research community. Many of the proposed approaches involve humans in the loop, see, e.g., [3, 7, 17, 18, 27, 29], and are based on iterative repairs of data values until some data-quality standards are satisfied. In the EM pipeline, existing data-cleaning approaches are typically used as a data-preprocessing step intended to improve downstream EM quality [6, 12], with all of the time-consuming data-cleaning work being completed prior to the EM step. While promising, this approach suffers from some limitations, including the following:

(1) As we have discovered in our preliminary experiments, incorrect data values do not necessarily hinder downstream EM quality. As such, cleaning data with the objective of meeting outside data-quality standards may not have the intended effects on the quality of EM outputs.
(2) It has been observed [9] that EM practitioners tend to minimize their data-cleaning efforts in the EM pipeline. To the best of our knowledge, existing data-cleaning approaches do not address explicitly the challenge of saving human efforts in data cleaning in the EM pipeline.

Our Approach. In this paper, we address the human-in-the-loop data-cleaning problem for EM. We introduce the problem of improving EM quality by data cleaning, in presence of a constraint on human experts' cleaning efforts and of feedback from the EM engine. To address the problem for the case of relational data, we propose a data-cleaning approach that, intuitively, strives to achieve "maximal speedup" in the improvement of EM quality with respect to the amount of cleaning time expended by human experts. We model the problem as a search problem in a state space, and propose a cleaning framework and two heuristics for directing the search. Given ground-truth tuple-match/nonmatch information and in presence of an EM engine viewed as a black box (that is, no assumptions are made about how the matching is done by the engine), our approach iteratively recommends to users to focus on individual data attributes whose cleaning is likely to be the most conducive to improving the accuracy of the output of the EM engine. The users accept or reject each recommendation based on whether the cleaning time for the attribute would be feasible within their remaining time budget. Our approach is domain independent, in that, instead of relying on users' domain expertise, it automatically discovers and takes advantage of information provided by the data, as well as using feedback from the EM black box.

Clearly, if ground-truth tuple-match/nonmatch information were available on much of the given data, then further EM might not be needed, and there might be no room for *EM-oriented* data cleaning. We make instead the realistic assumption that the ground-truth EM information is available (e.g., developed manually by domain experts) for relatively small samples of the data. Our approach can be deployed on the samples, leading to discovery of sequences of

HILDA'19, July 5, 2019, Amsterdam, Netherlands
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6791-2/19/07…$15.00
https://doi.org/10.1145/3328519.3329127

attributes such that cleaning the samples following the sequences is likely to maximize the "speedup" in improving EM accuracy. We then expect the resulting attribute sequences to be as effective in cleaning the original, typically much larger-scale, data, as long as the samples are representative and the same EM engine is used. As such, our approach fits well into, e.g., the paradigm of [10, 12, 20] that separates the "EM development stage," involving small data samples, from the "EM production stage," for the entire data.

2 MOTIVATING EXAMPLE

Fig. 1 showcases a toy example involving restaurant data. The EM pipeline takes as input the two tables shown in Fig. 1a, with incorrect data values circled. We use the term *attribute pair* to refer to the attributes from the tables that describe the same aspect of restaurants, such as `name-rname` and `city-rcity`.

As shown in Fig. 1b, before the incorrect values are repaired, the downstream EM engine can identify correctly the tuple-pairs $t_1 - t_5$ and $t_2 - t_6$ as a match and a non-match, respectively, see Fig. 1a for the EM ground truth. The procedure cannot resolve $t_3 - t_7$ or $t_4 - t_8$, because each pair has different names but the same city, and $t_3 - t_7$ is a match while $t_4 - t_8$ is a non-match.

To correctly identify more tuple-pairs, a user aims to repair incorrect values. Her time budget permits her to clean one attribute-pair, either for `city` or for `name`. Without assistance, she might choose `city`, as removing *", NY"* seems easier than fixing t_7. After the cleaning, however, the EM engine identifies no more correct tuple-pairs, as the user has made no change to values in $t_3 - t_7$ or $t_4 - t_8$. Thus, her cleaning efforts are wasted.

Suppose the user is directed instead to repair the value for `name`, rather than for `city`. Then the EM engine would correctly identify all the matches and non-matches shown in Fig. 1a.

3 PROBLEM FORMULATION

Efficient EM-oriented data cleaning is inherently a multi-objective problem, which can be refined into two legitimate concerns among real-life users: (I) How to maximize EM accuracy by data cleaning given a constraint of users' cleaning efforts, and (II) How to minimize user efforts in achieving desired EM accuracy. In this paper, we focus on the first problem. We understand EM quality in terms of *matching accuracy*: It indicates how many tuple-pairs across the tables have been correctly classified as matches or non-matches, and is commonly evaluated by F1-score as a balance of precision and recall [6, 20]. We evaluate users' cleaning efforts in terms of the amount of wall-clock time that they contribute to data cleaning.

In this paper we solve the following problem:

Efficient EM-Oriented Data-Cleaning Problem. Given two tables to be matched, achieve the maximum matching accuracy by asking users to repair incorrect values into their presumed ground-truth values in the tables until they run out of time. To solve this problem, we require the following information as input:

- a set of similarity metrics associated with each attribute-pair;
- a black-box EM engine, e.g., rule based or learning based;
- ground-truth matching status (labels) of tuple-pairs for evaluating matching accuracy;
- the current matching accuracy;
- a wall-clock time constraint on users' cleaning efforts.

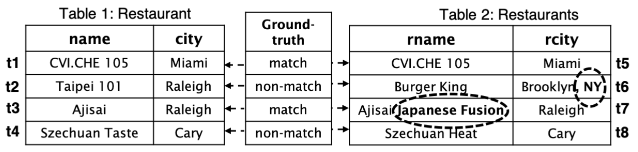


Table 1: Restaurant

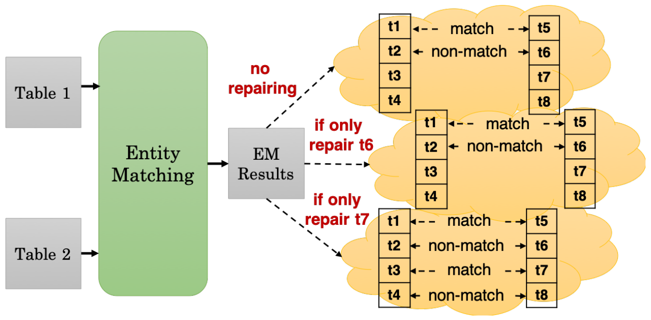
	name	city
t1	CVI.CHE 105	Miami
t2	Taipei 101	Raleigh
t3	Ajisai	Raleigh
t4	Szechuan Taste	Cary

Ground-truth
match
non-match
match
non-match

Table 2: Restaurants

rname	rcity	
CVI.CHE 105	Miami	t5
Burger King	Brooklyn, NY	t6
Ajisai Japanese Fusion	Raleigh	t7
Szechuan Heat	Cary	t8

(a) The input tables for the motivating example. The incorrect values are circled. For example, *"Brooklyn, NY"* is an incorrect value, with the true value being *"Brooklyn"* (i.e., with *", NY"* removed).



(b) The EM results on the input tables before data cleaning, after repairing only t_6, and after repairing only t_7. For example, after the value in t_7 is repaired, the EM engine can label correctly all the four tuple-pairs, see the ground-truth values in Fig. 1a.

Figure 1: The motivating example.

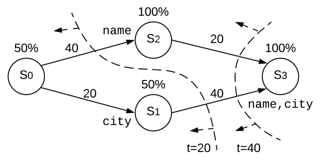


Figure 2: A state space for cleaning the data of Fig. 1a. S_1 and S_2 represent the data versions achieved by repairing errors in `city` and `name`, respectively, with associated time costs. The space to the left of each dotted line denotes the subspace determined by each time constraint.

The output includes the sequence of attributes to clean and the EM accuracy reached via the cleaning within the time budget.

Solving this problem means maximizing the number of correctly classified tuple-pairs for both matches and non-matches, with respect to the given ground-truth matching labels.

4 SOLUTION OVERVIEW

We model the problem as a search problem, and introduce search heuristics. In the cleaning process, our heuristics iteratively recommend to users to clean the attribute-pairs that can achieve the maximum speedup of matching accuracy with respect to the remaining time budget, by leveraging the underlying knowledge in the data and the feedback provided by the EM process.

4.1 Modeling the Problem

In our model, users identify data errors (incorrect values) and provide repair actions to correct them into user-presumed ground-truth

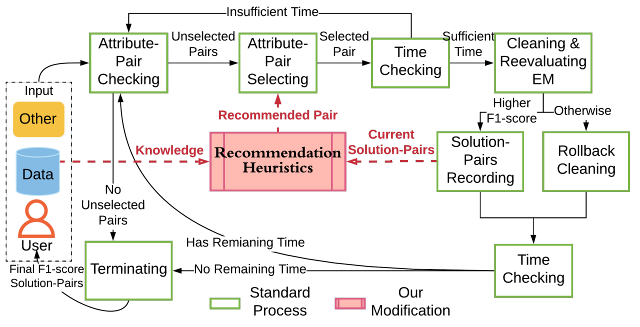
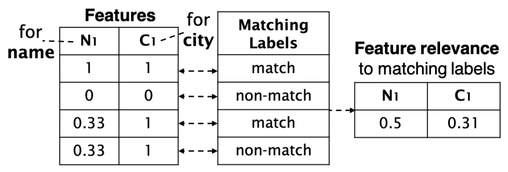


Figure 3: Our proposed cleaning framework for the efficient EM-oriented data-cleaning problem, where the cleaning process is guided by our attribute-pair recommendations. We interact with users to learn unknown cleaning time costs.



N_1	C_1	Matching Labels
1	1	match
0	0	non-match
0.33	1	match
0.33	1	non-match

N_1	C_1
0.5	0.31

Figure 4: Features (left) and feature relevance (right) for the attribute-pairs of Fig. 1a. E.g., N_1 is a feature for attribute-pair `name`, and the second 0.33 value is the string similarity between *Szechuan Taste* and *Szechuan Heat*. The feature relevance for N_1 and C_1 is computed using Equation 1.

values, as in, e.g., [17, 22, 28]. We assume that users presume only one ground-truth value for each data error, and that users repair errors by column at a time. These assumptions are natural in real-life cleaning processes, especially when cleaning is done using automated functions, and are supported by recent data-cleaning endeavors adopting the same setting [1, 2]. Our model is independent of how users repair data errors, such as by cleaning manually, coding automated functions, or interacting with third-party tools.

We model the cleaning process as a state-transition process in a state space, where each state (node) represents a distinct version of the data with all data errors in some (zero or more) attribute-pairs repaired, and each transition (edge) denotes the set of user-defined actions for repairing all data errors in an attribute-pair. Specific matching accuracy and cleaning-time costs are associated with states and transitions, respectively. E.g., in Fig. 2, state S_1 associated with matching accuracy 50% is reached by repairing the error "$, NY$" in Fig. 1, costing 20 time units. (It is easy to expand this model into one where each transition denotes a single-value repair.)

For a fixed cleaning-time budget, all the reachable states and their transitions make up a subspace, that is, the fraction of the full space determined by the time constraint. The state(s) associated with the maximum matching accuracy in the subspace are the *goal states*. For example, in Fig. 2, given the time constraints $t = 20$ and $t = 40$, states S_1 and S_2 are the respective goal states.

Search Problem. Given the state space, we *hypothesize* that the efficient EM-oriented data-cleaning problem can be solved by finding and following a path (*solution path*) in the subspace determined by the given time constraint; the path starts from the input-data state and terminates at a goal state. Each solution path is a sequence of actions that clean individual attribute-pairs (*solution-pairs*). Our hypothesis calls for searching and cleaning the solution pairs.

4.2 Our Approach

A major challenge to our hypothesis of Section 4.1 arises from the incompleteness of the information available for the subspace of interest, including unknown costs of transitions and unknown matching accuracy of states. Even if we had all the information, the full space would have 2^n states for n attribute-pairs, and the number of candidate paths would be exponential in the number of attribute-pairs, which makes brute-force search impractical. Thus, we consider greedy search for sequences of solution pairs.

Our overall strategy is to iteratively identify as solution-pairs the attribute-pairs that are the most effective in improving matching accuracy, among those that can be completely cleaned within the remaining time. Intuitively, these are the pairs that can achieve the highest F1-score increases after the users repair their data errors.

Fig. 3 describes our proposed efficient EM-oriented data-cleaning framework and details our strategy. Our framework modifies the standard cleaning process, in which users iteratively select and clean an attribute-pair until running out of time or of attribute-pairs, by introducing an attribute-pair-recommendation step. This step iteratively estimates the attribute-pair that is the most effective in increasing the F1-score based on our proposed search heuristics (Section 5), and recommends to the users to clean it. We then ask the users whether our suggested attribute-pair can be cleaned within the remaining time. If the users decline it due to insufficient time, we recommend to them the next most effective pair for increasing the F1-score. If cleaning our suggested attribute-pair introduces higher matching accuracy, we identify it as a solution-pair, otherwise we roll back the cleaning work to avoid risking more cleaning time for recovering the loss of matching accuracy. The eventually cleaned attribute-pairs are our best estimate of true solution-pairs, and the final state of the data is our best estimate of the goal state.

We next introduce two recommendation (search) heuristics for directing the cleaning process. Our heuristics are domain independent; they discover and take advantage of the underlying information in the data, and also use feedback provided by the downstream EM process. Our framework is parameterized, in that any other attribute-pair-recommendation method can be plugged in instead.

5 SEARCH VIA ATTRIBUTE-PAIR RANKING

In this section, we provide details of our heuristics for recommending attribute-pairs that are effective in improving entity-matching accuracy. Our *basic heuristic* suggests attribute-pairs based on their relationships with the ground-truth matching labels, while our *enhanced heuristic* also considers their inter-relationships and the feedback provided by the EM process.

5.1 Power to Discriminate among Entities

In Fig. 1b, we observe that cleaning the attribute-pair for restaurant `name` introduces higher matching accuracy than cleaning for `city`.

Based on this observation, we *hypothesize* that attribute-pairs' potential for improving matching accuracy can be represented by their power to discriminate between real-world entities. Thus, we can achieve the maximum improvement in matching accuracy by cleaning the data in the most discriminative attribute-pairs. More discriminative attribute-pairs can provide EM engines with more identifying information about real-world entities to help distinguish them, and thus to reach higher matching accuracy.

In this paper, we propose ways to measure the level of discriminative power for attribute-pairs. Based on this measure, we introduce our *basic* and *enhanced* heuristics. Our *basic heuristic* recommends attribute-pairs following the ranking of their discriminative power. In each iteration of the cleaning framework, the highest-ranked (most discriminative) attribute-pair not yet recommended will be returned as the most effective one in increasing the F1-score.

5.2 Detecting Discriminative Power and Generating Ranking of Attribute-Pairs

5.2.1 Detecting Discriminative Power of Attribute-Pairs. We define the discriminative power of attribute-pairs as follows.

Definition 1 (*Notion of level of discriminative power, LoD).* The *level of discriminative power (LoD)* of an attribute-pair is the degree to which we are confident that in the completely clean input data, if we have the same value in the pair, we must have the same entity, otherwise we must have different entities. The LoD range is [0, 1], from completely unsure to completely confident.

As an example, in Fig. 1a, name has more LoD than city, because it can provide more identifying information for restaurants.

Challenges in detecting LoD for attribute-pairs arise from the lack of integrity constraints, e.g., functional dependencies, in the problem input and from the presence of data errors. Therefore, we have to estimate LoDs for attribute-pairs; we propose doing it using the notion of *feature relevance* [5, 19, 25, 26, 31].

5.2.2 Feature and Feature Relevance for Entity Matching. A *feature* is a numeric representation of an attribute-pair in terms of its string similarities between values [6, 11, 14, 20]. We introduce *feature relevance, rel(F; L),* as the *Information Gain (I)* of a feature (*F*) to the ground-truth matching labels (*L*):

$$rel(F;L) = I(F;L) = H(F) - H(F|L), \quad (1)$$

where $H(F)$ and $H(F|L)$ denote the *entropy* of F and *conditional entropy* of F given L, respectively [5, 25, 26]. For the computation, we treat each similarity inside a feature as a discrete value, and employ its frequency as its probability. Intuitively, feature relevance suggests the amount of identifying information that values in the attribute-pair can provide to distinguish real-world entities. We estimate LoDs of attribute-pairs as their feature relevance $rel(F;L)$. If a pair can be converted into multiple features using different similarity metrics, we use its highest feature relevance.

For example, in Fig. 4, each attribute-pair of Fig. 1a is converted into a feature using the *Jaccard* similarity metric [6, 11]. N_1 is more relevant to the ground-truth matching labels than C_1, because its information gain 0.5 is higher than that (0.31) of C_1. Therefore, attribute-pair name should have more LoD, as its feature N_1 can help better distinguish the truly matched and non-matched tuple-pairs.

5.2.3 Ranking Attribute-Pairs. Recall that our heuristic recommends attribute-pairs following their ranking of discriminative power (LoD). Their ranking score is defined as follows.

Definition 2 (*The basic ranking score).* Let i be a candidate attribute-pair to be ranked; its *ranking score,* denoted $r(i)$, is defined as

$$r(i) = max_{F \in \mathcal{F}}(rel(F;L)),$$

where $\mathcal{F}$ denotes the set of all features for the attribute-pair i generated by its associated similarity metrics.

Our **basic approach** generates, in its first iteration, the overall ranking of the attribute-pairs, by sorting them in the decreasing order of their ranking scores; attribute-pairs i with larger feature relevance would have higher ranking scores $r(i)$, and would therefore be ranked higher. In the later iterations, the ranking is used to recommend to the users the highest ranked attribute-pair that has not been recommended yet. Intuitively, the users are always asked to clean those remaining attribute-pairs that are the most powerful in discriminating real-world entities. Thus, the approach can lead users into the most effective search directions toward improving the matching accuracy and reaching the goal state whenever possible.

5.3 Enhancing the Ranking of Attribute-Pairs

The basic approach of Section 5.2 considers just the LoD of attribute-pairs, regardless of the inter-relationships between attribute-pairs or any feedback from the matching process. At the same time, the iterative nature of the framework offers an opportunity to leverage the feedback to guide the recommendations, by weeding out non-promising attribute-pairs. We now describe how to enhance the basic approach by taking into account the complementarity between attribute-pairs and the solution-pairs identified so far.

5.3.1 Complementarity. The complementarity, $LoC_{A;\mathcal{S}}$, of attribute-pair A to set of attribute-pairs $\mathcal{S}$ signifies the amount of new identifying information that A can provide to help $\mathcal{S}$ in distinguishing real-world entities. We estimate $LoC_{A;\mathcal{S}}$ using the highest *feature complementarity,* $cmp(F,\mathcal{H};L)$, of a feature F of A to all features $\mathcal{H}$ of $\mathcal{S}$, with respect to the ground-truth matching labels L, computed as the *Joint Mutual Information* of F and $\mathcal{H}$ to L [5, 25, 26, 30]:

$$cmp(F,\mathcal{H};L) = I(F;L) - \frac{1}{|\mathcal{H}|}\sum_{H \in \mathcal{H}} I(H;F) + \frac{1}{|\mathcal{H}|}\sum_{H \in \mathcal{H}} I(H;F|L).$$

For example, in Fig. 4, the complementarity of city to name, $LoC_{\texttt{city;name}}$, is 0, as city does not help name distinguish any more restaurants. On the other hand, an attribute for a restaurant phone number could add positive LoC to name, as, intuitively, it could offer more identifying information in addition to restaurant name.

5.3.2 Ranking Attribute-Pairs. We treat the already identified solution-pairs as feedback from the EM process. When users repair attribute-pairs that are largely complementary to these solution-pairs, they collectively can potentially provide much more identifying information to distinguish entities. We enhance our ranking in such a way that the next ranked attribute-pairs should be as complementary to the already identified solution-pairs as possible.

Cleaning Time Costs for Music Data

Attribute-pairs	Real cleaning time costs (mins)
Song	18
Album	12
Artist	11
Genre	11

Collected Rankings Using Different Approaches

Approaches	Collected Rankings
Our basic approach	Song->Album->Genre->Artist
Our enhanced approach	Song->Genre->Album->Artist
Baseline: User 1	Artist->Song->Genre->Album
Baseline: User 2	Song->Album->Artist->Genre
Baseline: User 3	Genre->Album->Song->Artist
Baseline: User 4	Artist->Song->Album->Genre
Baseline: User 5	Genre->Artist->Song->Album

Figure 5: Cleaning-time costs and rankings for the music data. Our approaches ranked the more time-consuming pairs higher. This could be counterintuitive to users, as they generally prefer cleaning lower-cost attribute-pairs first.

Definition 3 (*The enhanced ranking score).* Let $\mathcal{S}$ be the set of identified solution-pairs, i be a candidate attribute-pair to be ranked, and L be the vector of ground-truth matching labels. We denote by $r'(i)$ the *enhanced ranking score* of i, defined as

$$r'(i) = max_{F \in \mathcal{F}}(cmp(F, \mathcal{H}; L)).$$

Here, $\mathcal{F}$ and $\mathcal{H}$ denote the set of all features for the attribute-pair i and the solution-pairs $\mathcal{S}$, respectively.

Note that the attribute-pairs having larger feature complementarity to the features of the identified solution-pairs would have higher ranking scores, and would therefore be ranked higher.

Our **enhanced approach** takes into account the already identified solution-pairs, and dynamically generates the enhanced ranking by iteratively adding attribute-pairs one at a time. Algorithm 1 provides the pseudocode for the approach. Here, attribute-pairs are selected based on their basic ranking score $r(i)$, until we identify a solution-pair. The rankings are then iteratively appended with the attribute-pairs that have the largest enhanced ranking score $r'(i)$. For example, when selecting the second pair to be ranked, if the first pair is identified as a solution-pair, $r'(i)$ is computed for the other non-ranked pairs, otherwise $r(i)$ is computed.

We use the enhanced ranking to recommend attribute-pairs to users. Intuitively, we ask users to clean the combinations of attribute-pairs that are the most powerful in discriminating real-world entities, and thus lead users to the most promising search directions toward reaching the goal state whenever possible.

```
Algorithm 1: Append an attribute-pair to enhanced ranking
Data: Two tables with ground-truth matching labels $L$ for
      tuple-pairs, the set of $m$ not-yet-recommended
      attribute-pairs ($A_i$) each associated with a set of
      similarity metrics ($\mathcal{M}_i$), $\mathcal{A} = \{(A_i, \mathcal{M}_i) | 1 \le i \le m\}$, and
      a sequence of $n$ identified solution-pairs ($S_j$) with their
      similarity metrics ($\mathcal{N}_j$), $\mathcal{S} = \{(S_j, \mathcal{N}_j) | 1 \le j \le n\}$.
Result: A recommended attribute-pair for the users to clean.
begin
    $\mathcal{F} = \emptyset$; $\mathcal{H} = \emptyset$;
    for $(A_i, \mathcal{M}_i) \in \mathcal{A}$ do
        $\mathcal{F}_i \leftarrow$ apply $\mathcal{M}_i$ to $A_i$ and get a set of features for $A_i$;
        $\mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}_i$;
    if $\mathcal{S}$ is empty then
        Find the $F \in \mathcal{F}$ with the highest $rel(F; L)$;
    else
        for $(S_j, \mathcal{N}_j) \in \mathcal{S}$ do
            $\mathcal{H}_j \leftarrow$ apply $\mathcal{N}_j$ to $S_j$ to get set of features for $S_j$;
            $\mathcal{H} \leftarrow \mathcal{H} \cup \mathcal{H}_j$;
        Find the $F \in \mathcal{F}$ with the highest $cmp(F, \mathcal{H}; L)$;
    $r \leftarrow$ the attribute-pair of $F$;
    Recommend $r$ to users.
```

6 PRELIMINARY EXPERIMENTS

In our preliminary experiments, we have validated our approach of finding solution-pairs by algorithmic recommendations, against the baseline approach of cleaning based on users' domain knowledge. Our enhanced approach consistently outperformed the baseline, by achieving higher speedup of improvement of EM accuracy with respect to the time constraints, and in some cases even delivering EM accuracy not achieved by the baseline, see Section 6.3.

6.1 Experimental Settings

Experimental Setup. Our experiments were conducted over the *music* and *baby-product* data domains, using samples of the data sets published by the *Magellan* group [8]. The samples have 80 and 79 tuples in the two music tables, and 67 and 76 tuples in the two baby-product tables. In line with existing data-cleaning endeavors, see, e.g., [16], we assume that the input tables are above 50% clean, as is typically assumed in real-life data-cleaning projects. Each domain provides ground-truth matching labels for 40 matched and 40 non-matched tuple-pairs. We employed `Magellan` as the black-box EM engine [10, 20], and implemented our approach on top of `Magellan` and the `scikit-feature` repositories [24].

The Baseline Approach. We compared our basic and enhanced approaches (Sections 5.2 and 5.3) with the baseline approach, which requires domain experts to direct the cleaning process, cf. e.g., [21]. In the baseline approach, users iteratively select attribute-pairs to clean based on their domain knowledge, with the objective of introducing higher EM accuracy. The sequence of user-selected attribute-pairs can be viewed as a ranking. We compared the EM accuracy achieved by using our rankings with that for the baseline rankings, with respect to different cleaning-time budgets.

Experimental Design. We worked with ten users with various backgrounds, dividing them equally into two groups. Each group was provided necessary information, e.g., the purposes of EM and data cleaning, and how to retrieve attribute-pair recommendations. Then each group performed EM-oriented data cleaning over the two data domains, first using their own rankings for one domain, and then using our recommended rankings (either basic or enhanced) for the other domain; the two groups explored the two domains in different order. We controlled the cleaning work by having the users repair values to the same ground-truth values.

The users came in as non-experts. We provided to them sufficient domain knowledge by giving them the ground-truth values for the samples, and by having them examine and compare the raw-data version with the ground-truth version before doing the baseline experiments. The outcomes served as the users' domain knowledge in directing the search and developing their attribute-pair rankings.

6.2 Experimental Results

Fig. 5 lists the rankings collected from the users and from our approaches. (Due to the space limit, we only list the rankings for the music domain.) Fig. 6 illustrates the improvements in EM accuracy achieved via cleaning with these rankings; the x-axis denotes the different time constraints, and the y-axis represents the final F1-scores after running out of the time budgets.

Our basic approach generally performed better than the baseline for both domains, occasionally achieving lower EM accuracy. On the other hand, our enhanced approach outperformed the baseline in such a way that, in almost all cases, we saw the more assured advantages as the more attribute-pairs were repaired, until the point at which the users had cleaned all the attribute-pairs. These advantages were presented even for short time budgets.

6.3 Summary

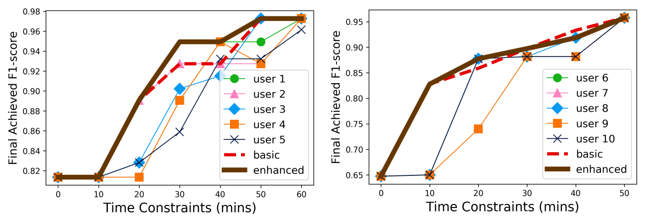
Our experimental results suggest two potential advantages of our proposed approaches over the baseline. First, our enhanced approach appears to introduce higher speedup in the improvement of EM accuracy with respect to time constraints. Given a cleaning-time budget, it can help users to take full advantage of their efforts and reach the highest cleaning efficiency as early as possible. It also accommodates well different ways of repairing incorrect values. Second, when users are given sufficient time, our approaches can even result in the EM accuracy that may not be achievable by the baseline approach. E.g., in Fig. 6a, starting from $t = 60$, the baseline approach by user 5 achieved around 96% EM accuracy, while our approach achieved around 97%. This is because our approach returned a sequence of solution-pairs consisting of *Song*, *Genre*, and *Album*, while the baseline sequence used all the pairs.

In the preliminary experimental results, our enhanced approach consistently outperformed the baseline approach, in which the users' domain knowledge was leveraged as heuristics to direct the cleaning. We posit that our enhanced approach looks promising, as it can help users reach the maximum EM accuracy with respect to the given time constraints, thus addressing our stated problem.

7 RELATED WORK

Entity Matching. Over the past few decades, a significant number of EM techniques have been developed, see, e.g., [6, 11, 14, 20, 23]. Due to the potentially severe impact of dirty data on EM quality, real-world EM practitioners look for effective and efficient data-cleaning solutions for improving EM accuracy [9, 10, 12]. For example, Johnson Control Inc. calls for the EM system Magellan to be extended with data-cleaning capabilities [20]. At this time, we are not aware of EM-focused research that would address this need.

Data Cleaning. In this paper, we introduced the problem of improving user efficiency in EM-oriented data cleaning. To the best of our knowledge, this research thrust has not been pursued in the literature. A direction of related work has focused on interactive data cleaning that would minimize user efforts [3, 4, 17, 22, 27–29]. These and related solutions typically interact with users by asking for, or confirming, valid data updates or repair rules. In the process,



(a) Music (b) Baby Product

Figure 6: F1-score improvements against time constraints t. To avoid bias, we used averaged F1-score over six EM matchers provided by Magellan, with the score at $t = 0$ achieved before any cleaning. Starting from $t = 60$ in (a) or $t = 50$ in (b), the users had cleaned all the attribute-pairs. Generally, our rankings improved the F1-scores more than the user-proposed rankings. This suggests that doing the more time-consuming cleaning steps first (see Fig. 5) could pay off.

the user effort is minimized in two ways: Directing users to the most beneficial data items and soliciting user-defined/preferred updates on them [3, 27, 29], and generalizing the cleaning behaviors of users to relevant data items [17, 27, 29]. Our work differs from these approaches in that our focus is on improving the quality of a downstream application (EM engine), rather than on enforcing certain data-quality standards. Our motivation stems from the fact that, as discussed earlier, higher data quality may not necessarily introduce higher downstream-application quality. QOCO, SampleClean, and ActiveClean [4, 22, 28] address problems that are similar to ours, by leveraging application-specific knowledge to guide the cleaning process. Unlike these projects, in our work we do EM-oriented data cleaning, and we do not limit the scope of possible types of data errors or restrict the choices for the downstream applications.

8 CONCLUSION

In this paper we studied the problem of maximizing EM quality by data cleaning under a time constraint on the cleaning efforts by users. We modeled the problem as state-space search, and introduced a framework and two recommendation heuristics to address it. The domain-independent framework and heuristics iteratively identify and direct users to clean the attribute-pairs that are the most effective in improving the EM accuracy within the time constraint. This is accomplished by discovering and taking advantage of the underlying knowledge in the data and of the feedback provided by the downstream EM process, rather than relying on the users' domain expertise. The preliminary experimental results suggest that our extended approach achieves measurable speedup in improving EM accuracy with respect to different time constraints. The approach can even deliver EM accuracy that is not achievable by directing the cleaning process by domain experts.

ACKNOWLEDGMENTS

We acknowledge the invaluable suggestions on this project from AnHai Doan, as well as helpful discussions with Ran Tan.

REFERENCES

[1] 2019. OpenRefine(GoogleRefine). http://openrefine.org

[2] 2019. Trifacta. https://www.trifacta.com/

[3] Ahmad Assadi, Tova Milo, and Slava Novgorodov. 2018. Cleaning Data with Constraints and Experts. In *Proceedings of the 21st International Workshop on the Web and Databases (WebDB'18)*. ACM, New York, NY, USA, 1:1–1:6. https://doi.org/10.1145/3201463.3201464

[4] Moria Bergman, Tova Milo, Slava Novgorodov, and Wang-Chiew Tan. 2015. Query-Oriented Data Cleaning with Oracles. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD '15)*. ACM, New York, NY, USA, 1199–1214. https://doi.org/10.1145/2723372.2737786

[5] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. 2012. Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection. *J. Mach. Learn. Res.* 13 (Jan. 2012), 27–66. http://dl.acm.org/citation.cfm?id=2188387

[6] Peter Christen. 2012. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection.* Springer, New York, NY, USA.

[7] Xu Chu, Ihab F. Ilyas, Sanjay Krishnan, and Jiannan Wang. 2016. Data Cleaning: Overview and Emerging Challenges. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. ACM, New York, NY, USA, 2201–2206. https://doi.org/10.1145/2882903.2912574

[8] Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, and Pradap Konda. 2019. The Magellan Data Repository. https://sites.google.com/site/anhaidgroup/projects/data.

[9] AnHai Doan. 2018. Human-in-the-Loop Data Analysis: A Personal Perspective. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics (HILDA'18)*. ACM, New York, NY, USA, 1:1–1:6. https://doi.org/10.1145/3209900.3209913

[10] AnHai Doan, Adel Ardalan, Jeffrey Ballard, Sanjib Das, Yash Govind, Pradap Konda, Han Li, Sidharth Mudgal, Erik Paulson, G. C. Paul Suganthan, and Haojun Zhang. 2017. Human-in-the-Loop Challenges for Entity Matching: A Midterm Report. In *Proceedings of the 2Nd Workshop on Human-In-the-Loop Data Analytics (HILDA'17)*. ACM, New York, NY, USA, 12:1–12:6. https://doi.org/10.1145/3077257.3077268

[11] AnHai Doan, Alon Halevy, and Zachary Ives. 2012. *Principles of Data Integration* (1st ed.). Morgan Kaufmann Publishers, San Francisco, CA, USA.

[12] AnHai Doan, Pradap Konda, Paul Suganthan G C, Adel Ardalan, Jeffrey R Ballard, Sanjib Das, Yash Govind, Han Li, Philip Martinkus, Sidharth Mudgal, Erik Paulson, and Haojun Zhang. 2018. Toward a System Building Agenda for Data Integration (and Data Science). *IEEE Data Eng. Bull.* 41, 2 (2018), 35–46. http://sites.computer.org/debull/A18june/p35.pdf

[13] Xin Luna Dong and Divesh Srivastava. 2013. Big Data Integration. In *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE, Piscataway, NJ, USA, 1245–1248. https://doi.org/10.1109/ICDE.2013.6544914

[14] Xin Luna Dong and Divesh Srivastava. 2015. *Big Data Integration.* Morgan & Claypool Publishers, San Rafael, CA, USA.

[15] Ahmed K Elmagarmid, Panagiotis G Ipeirotis, and Vassilios S Verykios. 2007. Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering* 19, 1 (Jan 2007), 1–16. https://doi.org/10.1109/TKDE.2007.250581

[16] Wenfei Fan and Floris Geerts. 2012. *Foundations of Data Quality Management.* Morgan & Claypool Publishers, San Rafael, CA, USA.

[17] Jian He, Enzo Veltri, Donatello Santoro, Guoliang Li, Giansalvatore Mecca, Paolo Papotti, and Nan Tang. 2016. Interactive and Deterministic Data Cleaning. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. ACM, New York, NY, USA, 893–907. https://doi.org/10.1145/2882903.2915242

[18] Ihab F. Ilyas and Xu Chu. 2015. *Trends in Cleaning Relational Data: Consistency and Deduplication.* Now Publishers, Hanover, MA, USA.

[19] George H. John, Ron Kohavi, and Karl Pfleger. 1994. Irrelevant Features and the Subset Selection Problem. In *Machine Learning, Proceedings of the Eleventh International Conference.* Morgan Kaufmann Publishers, San Francisco, CA, USA, 121–129. https://doi.org/10.1016/B978-1-55860-335-6.50023-4

[20] Pradap Konda, Sanjib Das, Paul Suganthan G. C., AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra. 2016. Magellan: Toward Building Entity Matching Management Systems. *Proc. VLDB Endow.* 9, 12 (Aug. 2016), 1197–1208. https://doi.org/10.14778/2994509.2994535

[21] Sanjay Krishnan, Daniel Haas, Michael J. Franklin, and Eugene Wu. 2016. Towards Reliable Interactive Data Cleaning: A User Survey and Recommendations. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics (HILDA '16)*. ACM, New York, NY, USA, 9:1–9:5. https://doi.org/10.1145/2939502.2939511

[22] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J. Franklin, and Ken Goldberg. 2016. ActiveClean: Interactive Data Cleaning for Statistical Modeling. *Proc. VLDB Endow.* 9, 12 (Aug. 2016), 948–959. https://doi.org/10.14778/2994509.2994514

[23] Guoliang Li. 2017. Human-in-the-loop Data Integration. *Proc. VLDB Endow.* 10, 12 (Aug. 2017), 2006–2017. https://doi.org/10.14778/3137765.3137833

[24] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. 2016. Feature Selection: A Data Perspective. (2016). arXiv:1601.07996 http://arxiv.org/abs/1601.07996

[25] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. 2017. Feature Selection: A Data Perspective. *ACM Comput. Surv.* 50, 6 (Dec. 2017), 94:1–94:45. https://doi.org/10.1145/3136625

[26] Patrick Emmanuel Meyer, Colas Schretter, and Gianluca Bontempi. 2008. Information-theoretic Feature Selection in Microarray Data Using Variable Complementarity. *IEEE Journal of Selected Topics in Signal Processing* 2, 3 (June 2008), 261–274. https://doi.org/10.1109/JSTSP.2008.923858

[27] Protiva Rahman, Courtney Hebert, and Arnab Nandi. 2018. ICARUS: Minimizing Human Effort in Iterative Data Completion. *Proc. VLDB Endow.* 11, 13 (Sept. 2018), 2263–2276. https://doi.org/10.14778/3275366.3284970

[28] Jiannan Wang, Sanjay Krishnan, Michael J. Franklin, Ken Goldberg, Tim Kraska, and Tova Milo. 2014. A Sample-and-clean Framework for Fast and Accurate Query Processing on Dirty Data. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD '14)*. ACM, New York, NY, USA, 469–480. https://doi.org/10.1145/2588555.2610505

[29] Mohamed Yakout, Ahmed K. Elmagarmid, Jennifer Neville, Mourad Ouzzani, and Ihab F. Ilyas. 2011. Guided Data Repair. *Proc. VLDB Endow.* 4, 5 (Feb. 2011), 279–289. https://doi.org/10.14778/1952376.1952378

[30] Howard Hua Yang and John Moody. 1999. Data Visualization and Feature Selection: New Algorithms for Nongaussian Data. In *Proceedings of the 12th International Conference on Neural Information Processing Systems (NIPS'99)*. MIT Press, Cambridge, MA, USA, 687–693. http://dl.acm.org/citation.cfm?id=3009657.3009755

[31] Lei Yu and Huan Liu. 2004. Efficient Feature Selection via Analysis of Relevance and Redundancy. *J. Mach. Learn. Res.* 5 (Dec. 2004), 1205–1224. http://dl.acm.org/citation.cfm?id=1005332.1044700