Subset Discovery for Entity Matching

Zheng Liang, Yafeng Tang, Hongzhi Wang[✉], Haifeng Cheng, and Xiaoou Ding

Harbin Institute of Technology
lz20@hit.edu.cn, 23B903047@stu.hit.edu.cn, wangzh@hit.edu.cn,
2021112251@stu.hit.edu.cn, dingxiaoou@hit.edu.cn

Abstract. Entity Matching (EM) is the task of identifying co-referent manifestations from multiple data sources. It has been a long-standing challenge in data integration, knowledge graph, and recommendation systems. In recent years, Machine Learning models are taking off in commercial EM pipelines. However, we observe that most challenging EM tasks have complex structures of local distributions, making ML model perform poorly when trained on the whole dataset. To mitigate this issue, we propose a rule-based subset discovery approach to identify multiple subsets. Our Subset Discovery Rule (SDR) are similar to conventional entity blocking rules, but aim at partitioning training data instead of filtering unmatched entity pair candidates. By training one EM model for each subset, the combined EM prediction can capture local data distributions, yielding accurate Entity Matching results. Naive SDR searching takes $O(2^n)$ model retraining, while unsupervised clustering methods perform poorly. Therefore, we propose an $O(n^2logn)$ Subset Discovery Rule Searching algorithm. We evaluate SDR on 11 EM benchmarks. The empirical results show that SDR is statistically comparable to SOTA EM methods. Compared with Random Forest (RF)-based EM, the F1-score of SDR-RF is 5.6% higher, and compared with Large Language Model (LLM)-based EM, the F1-score of SDR-LLM is 2.7% higher, showcasing its generalization property. Due to its model-agnostic design, SDR is an extensible plug-in for any machine learning-based EM model.

Keywords: Information Integration · Entity Matching · Subset Discovery.

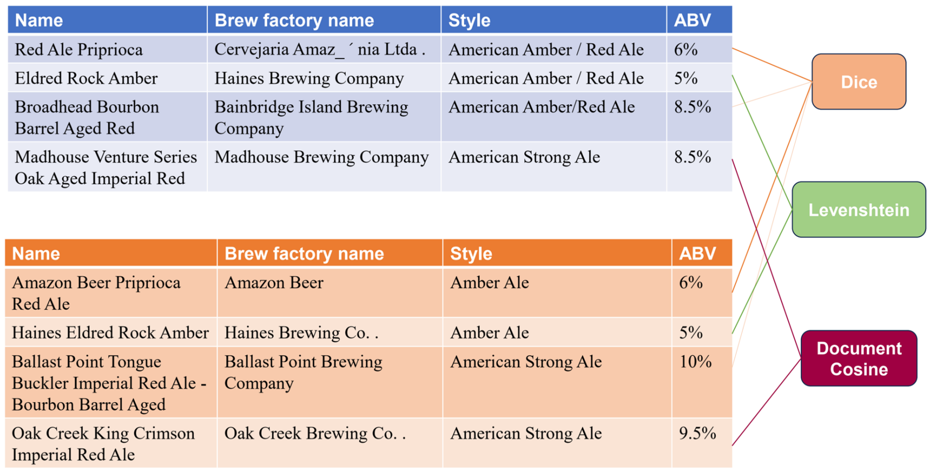
1 Introduction

Data preparation is recognized as the bottleneck in both data management [10] and data-centric AI [6]. To prepare data in terms of volume and quality [19], a crucial step is to collect scattered information for each real-world entity from multiple data sources, known as Entity Matching (EM) [11].

EM solutions originated from expert-designed similarity measures for entity attribute pairs [30]. Modern EM solutions use similarities as features for Machine Learning models like Random Forest. More currently, Pre-trained Language Model(PLM)-based methods [2,13] and Large Language Model(LLM)-based methods has become the State-Of-The-Art.

Despite the dominance of ML models over deep learning models on regular tabular data tasks, ML models still perform far from ideally on EM tasks. As shown in Example 1, one of the vital drawbacks of existing Machine Learning-based EM models is using the same EM model for a complex EM dataset that contains multiple sub-distributions.

Example 1. Suppose a data scientist is asked to build a Random Forest-based EM model for the Beer dataset in Figure 1. The first step is to select similarities as features for Random Forest training. Considering attribute "Brew factory name", although Levenshtein similarity seems suitable for long strings, there are actually some short strings in the attribute for which other similarity functions (e.g., Dice Distance) are more suitable [31]. Moreover, document cosine is more suitable for strings containing rich semantic information. Likewise, for LLM-based EM methods, the practice of using the same in-context examples for the QA call of all examples also fails to obtain good performances [12]. Therefore, the data scientist decides to train an independent EM model for subsets of data. The remaining problem is: **how to find proper subsets?**



Name	Brew factory name	Style	ABV
Red Ale Priprioca	Cervejaria Amaz_´ nia Ltda .	American Amber / Red Ale	6%
Eldred Rock Amber	Haines Brewing Company	American Amber / Red Ale	5%
Broadhead Bourbon Barrel Aged Red	Bainbridge Island Brewing Company	American Amber/Red Ale	8.5%
Madhouse Venture Series Oak Aged Imperial Red	Madhouse Brewing Company	American Strong Ale	8.5%

Name	Brew factory name	Style	ABV
Amazon Beer Priprioca Red Ale	Amazon Beer	Amber Ale	6%
Haines Eldred Rock Amber	Haines Brewing Co. .	Amber Ale	5%
Ballast Point Tongue Buckler Imperial Red Ale - Bourbon Barrel Aged	Ballast Point Brewing Company	American Strong Ale	10%
Oak Creek King Crimson Imperial Red Ale	Oak Creek Brewing Co. .	American Strong Ale	9.5%

Fig. 1. An example from the Beer dataset.

Previous study [31] mentioned this problem, but falls short on the solution, using AutoML to generate features for random forest training on *the whole* dataset. Our work targets this problem of subset discovery, divides the EM dataset into multiple subsets, and trains an independent EM model for each aforementioned subset. The idea is shown in Figure 1.

Unfortunately, dividing EM datasets into subsets is non-trivial, due to the following challenges:

(1) Redundancy of Subset Space: The subset candidates are exponential to the dataset size. The exhaustive search of such subset space is irrational and computationally intractable, while the design choice of entity subset is neither theoretically nor empirically explored.

(2) Extensive model retraining: The efficiency problem is outstanding. EM models require extremely high computation resources to deploy, and online

LLM prompts are charged per token, which cannot be reduced using parallel methods.

To our knowledge, few existing methods can handle the above two challenges simultaneously. Traditional rule-based techniques, exemplified by Autofuzzyjoin [17], may falter when dealing with less structured datasets and extensive textual information, due to their reliance on narrow word-level similarity metrics and TF-IDF scores [25]. On the other hand, advanced deep learning approaches, such as Sudowoodo [32], present challenges in interpretability, making it hard for users to comprehend the rationale behind entity matches and mismatches. These sophisticated models often entail significant expenses for debugging, necessitating numerous iterative configurations and expert insight to make informed decisions. Although dataset preparation for EM has been extensively studied in unsupervised EM and entity blocking, none of such method aims at dividing dataset into subsets for independent model training.

Inspired by recent success on regression [15], we propose to use entity attribute similarity predicates as conditional rules to divide an EM dataset into subsets, and the validation/testing datasets are also divided according to such conditional rules. Training an individual model on training data in each subset, such model predicts EM results for the corresponding validation/testing datasets. As shown in Figure 1, the bottleneck of such a proposal is searching rules to obtain the best EM performance.

Our contributions are summarized as follows.

- **Formalizing Subset Discovery.** We formally define the problem of Subset Discovery for Entity Matching (SDEM), discuss a straightforward clustering-based approximate solution, and propose a Subset Discovery Rule (SDR) to trim the size of solution space with attribute similarities which have been well-celebrated in EM literature. (Section 2)
- **Subset Discovery Rule Searching Algorithm.** To solve the SDEM problem with SDR, we propose an exact algorithm called Subset Discovery Rule Searching (SDRS), reducing the time cost from $O(2^n)$ to $O(n^2 logn)$ via EM model sharing (Section 3.2) and a Bayesian parameter recommendation method for SDRS algorithm. (Section 3.3)
- **Extending Entity Matching Similarity Library.** Additionally, we extend the conventional similarity library for Random Forest-based EM, introducing 7 simple but effective similarities that have not been explored in EM literature. (Section 3.1)
- **Evaluation.** Empirical results on 11 benchmarks show that SDR enhances the F1-score of Random Forest methods by 5.6% and Large Language Model methods by 2.7%, yielding performance comparable to SOTA EM models. Also, our similarity Library can enhance the F1-score of SOTA Random Forest methods by 2.8%. (Section 4)

2 Overview

2.1 Problem Statement

EM solutions for structured data consist of at least two phases: blocking and matching [1], and often assume that upstream tasks have prepared datasets with aligned attributes [28]. Therefore, we focus on entity matching on relational data as follows, and treat entity blocking as an orthogonal problem.

Definition 1. *Entity Matching (EM): Given two relational tables T_1 and T_2 with same schema $A = (A_1, A_2, ..., A_m)$, an Entity Matching task $EM(T_1, T_2)$ builds a model $\mathcal{M}$ to determine whether each pair of records $t_1, t_2 \in T_1 \times T_2$ refers to the same entity in the real world.*

General EM methods [22,14] ignore latent sub-distribution during EM model training, thus cannot obtain ideal performance on EM task. Therefore, we focus on the problem of Subset Discovery for Entity Matching defined as follows.

Problem 1. Subset Discovery for Entity Matching(SDEM): Given $EM(T_1, T_2)$ on attribute set $A = (A_1, A_2, ..., A_m)$, SDEM task finds data partition DP on a fixed EM model M as follows.

$$\begin{aligned}
\max \quad & P\left(\bigcup_{dp \in DP} M(dp(T_1 \times T_2))\right) \\
\text{s.t.} \quad & \bigcup_{dp \in DP} dp(T_1 \times T_2) = T_1 \times T_2, \\
& \forall dp_1, dp_2 \in DP, \quad dp_1(T_1 \times T_2) \cap dp_2(T_1 \times T_2) = \emptyset
\end{aligned} \tag{1}$$

where $dp \in DP$ is a projection from $T_1 \times T_2$ to a subset of $T_1 \times T_2$, and $P()$ is the evaluation metric.

Traversing all solutions of the above problem is redundant in terms of computation, so we use attribute similarity to trim the size of solution space. The attribute similarity is a crucial concept in rule-based EM, machine learning-based EM, and even LLM-based EM. We give its definition as follows.

Definition 2. *Similarity: Given $EM(T1, T2)$, a similarity s is a mapping from attribute values A_p of tuple pair $t^1 \in T_1, t^2 \in T_2$ to a domain $\mathcal{S}$, so the candidate similarity set is:*

$$S = \{s | s : f(t_i^1.A_p, t_j^2.A_p) \to \mathcal{S}\}$$

Based on Similarity, we can define our Subset Discovery Rule as follows:

Definition 3. *Subset Discovery Rule: Given $EM(T_1, T_2)$ on attribute set $A = (A_1, A_2, ..., A_m)$ and Similarity Function set $S = \{s_1, s_2, \ldots\}$, A Subset Discovery Rule (SDR) r is:*

$$r : \bigwedge_{A_p \in R, s_q \in S} s_q(A_p) \ \phi \ \delta_{pq} \tag{2}$$

,where $\delta_{pq} \in R$, $\Phi = \{>, <\}$, the subset of $T_1 \times T_2$ satisfying r is $r(T_1 \times T_2)$.

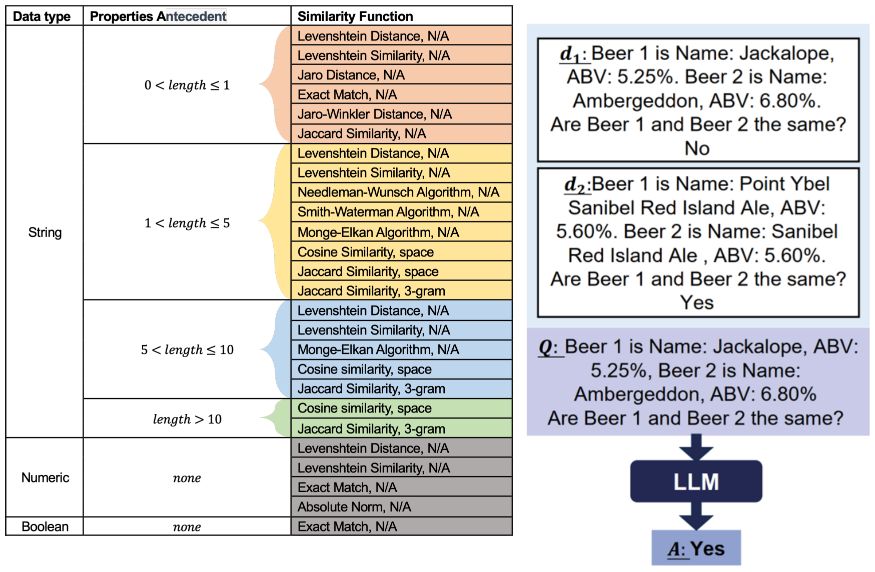
It should be noted that although Subset Discovery Rules follow the same form of intuitive rule-based entity blocking rule, our target is completely different from entity blocking, which can be defined as follows.

Problem 2. Subset Discovery Rule Searching(SDRS): Given $EM(T_1, T_2)$ on attribute set $A = (A_1, A_2, ..., A_m)$, the attribute description set $D = \{D_i\}$ and the similarity function set $S = \{S_j\}$, SDRS task finds rule set R as follows.

$$\begin{aligned} \max \quad & P(\cup_{r \in R} M(r(T_1 \times T_2))) \\ \text{s.t.} \cup_{r \in R} \, & r(T_1 \times T_2) = T_1 \times T_2 \end{aligned}$$

In Problem 2, subsets within the same SDR will be used as a training dataset for the same model, which aligns with the similarity usage in Rule-based EM [6], LLM-based EM [8], and entity blocking [26].

2.2 Entity Matching Models



Data type	Properties Antecedent	Similarity Function
String	$0 < length \leq 1$	Levenshtein Distance, N/A
		Levenshtein Similarity, N/A
		Jaro Distance, N/A
		Exact Match, N/A
		Jaro-Winkler Distance, N/A
		Jaccard Similarity, N/A
	$1 < length \leq 5$	Levenshtein Distance, N/A
		Levenshtein Similarity, N/A
		Needleman-Wunsch Algorithm, N/A
		Smith-Waterman Algorithm, N/A
		Monge-Elkan Algorithm, N/A
		Cosine Similarity, space
		Jaccard Similarity, space
		Jaccard Similarity, 3-gram
	$5 < length \leq 10$	Levenshtein Distance, N/A
		Levenshtein Similarity, N/A
		Monge-Elkan Algorithm, N/A
		Cosine similarity, space
		Jaccard Similarity, 3-gram
	$length > 10$	Cosine similarity, space
		Jaccard Similarity, 3-gram
Numeric	*none*	Levenshtein Distance, N/A
		Levenshtein Similarity, N/A
		Exact Match, N/A
		Absolute Norm, N/A
Boolean	*none*	Exact Match, N/A

d_1: Beer 1 is Name: Jackalope, ABV: 5.25%. Beer 2 is Name: Ambergeddon, ABV: 6.80%. Are Beer 1 and Beer 2 the same? No

d_2: Beer 1 is Name: Point Ybel Sanibel Red Island Ale, ABV: 5.60%. Beer 2 is Name: Sanibel Red Island Ale , ABV: 5.60%. Are Beer 1 and Beer 2 the same? Yes

Q: Beer 1 is Name: Jackalope, ABV: 5.25%, Beer 2 is Name: Ambergeddon, ABV: 6.80% Are Beer 1 and Beer 2 the same?

(a) Similarities of RF-based EM (b) LLM-based EM

Fig. 2. EM models this work focus on.

We concentrate on subset partition for entity matching and, therefore, opt for typical entity matching models in Problem 1. To provide clarified context for subsequent discussions, we introduce the details of the selected entity-matching models we focus on and how they can benefit from SDR in this section.

Random Forest-based EM. Random Forest-based EM [24] uses similarities for features of the well-celebrated Random Forest model. Such methods are widely applied in commercial EM pipelines, due to the following advantages: (1) Computational Efficiency. The RF-based EM does not rely on deep neural networks or LLM API on a GPU server. (2) Explainability. Tree models are explainable, and the feature set is easy to understand. Several RF-based EM explainers have been proposed. (3) Flexibility. It is easy for a data scientist to debug the data, model, and feature set of RF-based EM.

The mainstream feature space for RF-based EM model is shown in Figure 2. Given a SDR set, we apply each SDR to an EM benchmark to get the corresponding training/test data and train an EM model on each subset to predict the labels for the test data. Finally, the predicted labels of each subset are gathered as the final EM result.

Large Language Model-based EM. LLM-based EM [8] is a specialized algorithm using in-context learning, prompts contain demonstration examples, task instructions, etc. Its distinct advantages over other algorithms include: (1) Effectiveness. Such methods achieve SOTA performance, and their performance are further alleviated with new LLM models(2) Flexibility. Such methods only need proper questions, so they are flexible for debugging as well. (3) Efficiency. LLM can be accelerated via batch prompting and parallel questioning.

Originally, LLM-based EM is designed with the same prompt for all examples. Given a SDR set, LLM-based EM can divide the dataset into subsets, and each subset can have different in-context examples.

2.3 Overview of our Approach

The essence of subset discovery hinges on harnessing insights from distribution and data labels within EM datasets to guide the EM model training, and, when combined, achieving feasible performance. To mitigate this issue, our approach emphasizes three primary facets, as illustrated in Figure 3. Firstly, to enrich the limited feature space of RF-based EM, we implement a similarity library to amplify its size. Secondly, we introduce two algorithms for subset discovery, one for the unsupervised scenario and the other one for the supervised scenario. Subsequently, we use Bayesian optimization to recommend parameters for SDR Searching algorithm on the new dataset.

3 Methodology

3.1 Extended Similarity Library

In this section, we briefly introduce several similarity functions for RF-based EM, which have rarely been explored in EM literature. Supposing the input attribute pairs are presented as s_1 and s_2, our extended similarities are as follows.

Affine Similarity. Affine Similarity [8] assigns scores to the four operations between characters in Levenshtein similarity: match/mismatch, insert, and delete.

$$affine(s_1, s_2) = \frac{Levenshtein(match, mismatch, insert, delete)}{\max(len(s_1), len(s_2))}$$

BulkDelete Similarity. BulkDelete similarity [29] is based on LCS distance [11] and the length of attribute pairs, which is defined as follows.

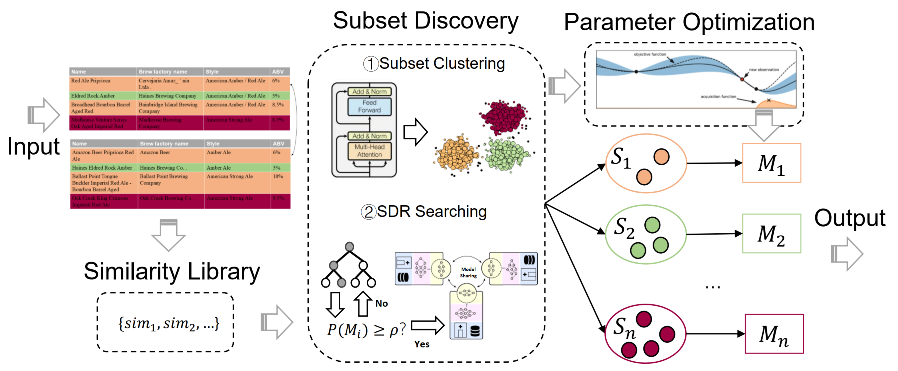


Fig. 3. Process of our approach. Note that 'Similarity Library' is only used for the feature set of Random Forest-based EM Models.

$$BulkDelete(s_1, s_2) = \frac{LCS(s_1, s_2)}{min(len(s_1), len(s_2))}$$

Simpson. As an extension of Jaccard, Simpson [33] is calculated as the ratio of the intersection size to the size of the smaller set.

$$SimpsonSimilarity(s_1, s_2) = \frac{|s_1 \cap s_2|}{min(len(s_1), len(s_2))}$$

Document-cosine. Document-cosine [18] is related to vector-based similarities for calculating numerical attributes [4].

$$Doc - Cosine = \frac{entropy(str_1) \cdot entropy(str_2)}{|entropy(str_1)||entropy(str_2)|}$$

Birnbaum Similarity. Given a co-occurrence matrix M, where M_{ij} represents the frequency of character j co-occurring with attribute value i. R is the global schema. Birnbaum similarity [5] can be defined as follows.

$$Birnbaum(s_1, s_2) = \frac{\sum_{i \in R, j \in s_1 \cup s_2} \min(M_{ij}, M'_{ij})}{\sum_{i \in R, j \in s_1 \cup s_2} \max(M_{ij}, M'_{ij})}$$

where M_i and M'_i present the co-occurrence matrix of s_1 and s_2.

3.2 Subset Discovery

To solve Problem 1, the brute force solution is to traverse all possible subset partition candidates and find the solution with the best overall EM performance, taking $O(2^n)$ model retraining for n entity pairs. In this section, we introduce two computational-friendly methods for Problem 1: Subset Clustering and SDR Searching.

The Subset Clustering Solution. A straightforward solution for Problem 1 is to divide an EM dataset into subsets with clustering algorithms. This approximate solution has three steps: (i) all entity pairs are projected into a representation space *rep*(e.g. a Pretrained Language Model encoder). (ii) a clustering algorithm *clst* forms subsets. (iii) Training data in each subset will be used to train an EM model to match entity pairs in the test data in the corresponding subset.

Compared to brute force search, the clustering solution takes only the cost of the clustering algorithm(e.g., $O(nlogn)$ for DBSCAN), but has no performance guarantee from the feedback of training data and validation data labels, resulting in poor performance as shown in Table 3. As a trade-off between EM performance and subset discovery time cost, we propose the SDR searching algorithm.

Algorithm 1: SDR Searching

```
Require: An Entity Pair Set $T_1 \times T_2$, label set $Y$, performance threshold $\rho_0$,
    Entity Pair similarity $X$, predicates space $P$
Ensure: The entity matching results for $T_1 \times T_2$
 1: Rule set $\Sigma = \emptyset$, $totalError = 0$, $blockNum = 0$
 2: Duplicate Record Detection Model set $F = \emptyset$
 3: Priority Queue $Q = \{(C = \emptyset, i(C) = 0\}$
 4: while queue $Q$ is not empty do
 5:   Conjunction $C = Top(Q)$
 6:   if $\exists f \in F, s.t. 1 - P(f(D_C)) \leq \rho_M$ then
 7:     Update $\rho = 1 - P(f(D_C)), blockNum = blockNum + 1, totalError =$
        $totalError + \rho, \Sigma = \Sigma \cup \{(f, \rho, C)\}$
 8:   else
 9:     $ind(C) = max_{f \in F} \frac{|\{t|t \in D_C \wedge |t.Y - f(t.X)| \leq \rho_M\}|}{|D_C|}$
10:     train $f$ under data $D_C$
11:     if $totalError + 1 - P(f(D_C)) \leq \rho_M \cdot (blockNum + 1)$ then
12:       Update
          $\rho = 1 - P(f(D_C)), blockNum = blockNum + 1, totalError =$
          $totalError + \rho, \Sigma = \Sigma \cup \{(f, \rho, C)\}, F = F \cup \{f\}$
13:     else
14:       Decide split predicate $p_1, ..., p_n \in P' \subset P$
15:       for $p_i \in \{p_1, ..., p_n\}$ do
16:         Update $C_i = C \wedge p_i$
17:         Add to queue $Q = Q \cup \{(C_i, ind(C))\}$
18:       end for
19:     end if
20:   end if
21: end while
22: return $F(\Sigma)$
```

The SDR Searching Algorithm. Inspired by recent success of conditional regression [15], we explore a fascinating yet unexplored idea: *Can we develop an independent EM model trained and tested on "conditional entity pairs"?*

To solve Problem 2, we propose a top-down SDR Searching algorithm, accelerating SDR-based EM subset discovery by model sharing. The pseudo code is given in Algorithm 1. In each iteration, we take the most "promising" EM data subset(line 9), try to find the possible model sharing(line 6 to line 7) for it, and train a new model(line 10 to line 12) or split the dataset via top-down tree-based search [9]. Algorithm 1 ensures each subset shared in line 6 to line 7 with F1 score higher than ρ_M.

Line 14's predicates aid binary searches for more data splits. Line 10 adds an EM model with each new SDR, limiting Line 6's model tests to $|\Sigma| \leq n$. SDR discovery halts per tuple's regression, ensuring $O(logn)$ partitions in Line 19's search. The worst-case is $O(n)$ models tested O(logn) times per tuple, yielding $O(nlogn)$ complexity, so the time cost for Algorithm 1 is $O(n^2 logn)$.

3.3 Parameter Recommendation

A bottleneck of Algorithm 1 is that the performance is sensitive to the error parameter ρ. If it is set too high, the algorithm ends up with poor performance or even a single subset. If too low, the algorithm takes tens of hours, making it unrealistic to use. We use Bayesian optimization [7] to enhance the ultimate performance. For the EM performance function $P : \rho \to R, \rho \in \varrho$, the purpose of is to find the value on ϱ that miximizes the function, which is $\rho^* = \arg\max_{\rho \in \varrho} P(\varrho)$. The idea of Bayesian optimization is selecting the search direction at locations with high uncertainty in the predicted values to identify the optimal solution for the performance function of EM models.

Algorithm 2: Bayesian Parameter Recommendation

```
Input  : Surrogate function $G$, evaluation function $SDR$, acquisition
         function $\alpha$.
Output: Recommended parameter $\rho \in \varrho$
1 $\hat{\rho} \leftarrow \rho_0$;
2 for $j = 1, 2, ...$ do
3     $\rho^{j+1} \leftarrow \arg\max_{\rho^{j+1}} \quad \alpha(\rho^j, (D_\rho)^j)$;
4     Asking the evaluation function $SDR$ for $y^{j+1} = SDR(\rho^{j+1})$;
5     Extend the data $(D_\rho)^{j+1} \leftarrow \{(D_\rho)^j, (\rho_i^{j+1}, y^{j+1})\}$;
6     Update surrogate model $G$;
7 return $\rho$
```

Algorithm 2 describes the Bayesian Optimization approach. We first manually set variable for initialization (lines 1). In each iteration, the optimal parameter is determined with acquisition function α (line 3). Specifically, α seeks its maximum value, and computes performance on SDR $SDR(\rho)$ for the parameter (line 4). This newly obtained parameter is appended to the dataset points D_ρ (line 5) for subsequent iterations after surrogate model updates(line 6).

In summary, we harness the inherent relationships among the curated datasets to efficiently and precisely advocate hyper-parameters for optimizing the performance of the entity-matching algorithm.

4 Evaluation

We test the effectiveness of our algorithm using random forest and gpt-4o-mini as the LLM to handle entity matching tasks.

Table 1. EM Datasets

Type	Dataset	Size	|Attr|
Small&Easy	Fodors-Zagats(FZ)	946	6
	iTunes-Amazon(IA)	540	8
	s-iTunes-Amazon(s-IA)	540	8
	Beer	450	4
Large&Easy	s-DBLP-ACM(s-DA)	12363	4
	s-DBLP-GoogleScholar(s-DG)	28708	4
	DBLP-ACM(DA)	12363	4
	DBLP-GoogleScholar(DG)	28707	4
Large&Hard	Amazon-Google(AG)	11460	3
	Walmart-Amazon(WA)	10242	5
	s-Walmart-Amazon(s-WA)	10242	5

4.1 Experimental Setting

Datasets. All of the benchmarks have been evaluated in previous work [18,31]. We split the dataset and use 80% for training and 20% for testing.

Baseline. To validate the effectiveness of SDR, we selected six mainstream EM solutions for comparison.

• AutoML-EM [31] is a machine-learning-based approach that proceeded with active learning and trained features based on AutoML.

• Magellan [16] is the first approach to establish the connection between attribute statistics and summary statistics, which kept the developer in the loop in every step.

• Ditto [18] is a deep-learning based subsequent state-of-the-art EM solution which injects Domain-Knowledge with NLP techniques.

• Rotom [21] is a data augmentation framework for EM, which outperforms DeepMatcher [23], RoBERTa-based models [3] with limited training samples.

• Sudowoodo [32] is the state-of-the-art data preparation framework based on contrastive self-supervised learning.

Evaluation Metric and Environment. We use F1 score to evaluate the performance, and time (in seconds) to evaluate efficiency. We also report the number of subsets in Figure 5. All experiments were run on an Ubuntu 20.04 server with an Intel Xeon E5-2678 v3 CPU and 32GB memory, Tesla M40 GPU.

4.2 End-to-End Performance

For the 11 EM benchmarks, we conduct a comparison experiment of results across different entity matching methods.

Table 2. End-to-End EM results on machine learning and deep learning EM models

Method	AutoML-EM		Magellan		Ditto		Sudowoodo		Rotom		SDR	
Dataset	F1	time	F1	time	F1	time	F1	time	F1	time	F1	time
FZ	**100.0**	3683	**100.0**	32	95.2	53	**100.0**	538	97.7	57	**100.0**	2
IA	71.7	3686	52.1	33	62.3	135	87.3	659	**92.9**	95	87.5	6
s-IA	96.4	3681	91.2	39	87.2	26	94.3	802	94.5	98	**98.4**	2
BR	93.6	3686	78.8	34	45.2	53	63.6	373	52.2	44	**94.5**	2
DS	88.3	3708	81.2	98	89.9	2451	-	-	88.3	341	**90.4**	183
DA	94.7	3698	88.3	70	**96.2**	2704	-	-	90.5	177	91.6	61
s-DG	94.5	3732	92.3	111	90.2	3052	89.5	2351	84.8	299	**97.2**	931
s-DA	**98.5**	3691	98.4	67	96.9	1187	94.1	2403	90.4	165	98.4	18
AG	56.0	3695	49.6	45	58.7	251	-	-	57.8	103	**73.2**	53
WA	45.6	3668	36.7	53	49.7	311	33.0	1440	**71.4**	154	64.6	230
s-WA	75.7	3667	71.9	54	65.0	577	30.0	1510	68.5	125	**80.5**	443
average	83.2	3690	76.4	58	76.0	982	74.0	1260	81.5	151	**88.8**	175

Enhancing Random Forest-based EM with SDR. We apply SDR on Random Forest-based EM with our similarity library. On average, SDR-enhanced RF achieves over 5.5% improvement while saving 3515 seconds compared to AutoML-EM. We also evaluate SDR against state-of-the-art deep learning solutions, using the same experimental setup as the state-of-the-art method, Sudowoodo. The results of deep learning-based EM baselines are consistent with [27]. In Table 2, SDR surpasses most existing deep learning solutions on several datasets, particularly where only limited training data is available. Nevertheless, for larger datasets such as Walmart-Amazon, with up to 10k entity pairs, Rotom can capture complex features and perform better.

Enhancing LLM-based EM with SDR. We further apply SDR on LLM-based EM. We test GPT-4o and Llama3 on a LLM-EM example selection methods called validation cluster. The validation cluster will apply DBSCAN on zero-shot wrongly predicted labels in training data, and select examples from the clusters. In Table 3, SDR achieves 2.7% improvement compared with baseline at the cost of 4× more input-token on GPT-4o-mini. Although GPT-4o-mini is a representative LLM model, it is notable that its financial cost is still a concern. As a financially friendly substitute, we also test SDR-enhanced LLM-EM on llama3-70b, and SDR demonstrates consistent impact on this LLM model, with over 2.96% improvement on F1 score observed.

Critical Difference Diagram. Additionally, Figure 4 reports the *mean ranking of F1 score* among all EM datasets. Competitors falling in one clique (the bold horizontal line) have no statistical difference, while the opposite for

Table 3. End-to-End EM Results on Large Language Models

	GPT-4o		GPT-4o-SDR		llama3-70B		llama3-70B-SDR	
Dataset	F1	token	F1	token	F1	token	F1	token
FZ	**100.0**	244756	**100.0**	244756	95.2	747472	**100.0**	747472
IA	96.3	170193	**97.8**	5928008	94.7	627613	96.2	4456484
s-IA	96.2	170193	**96.6**	2352284	85.2	518477	96.2	645585
BR	92.9	58720	92.9	58720	93.3	189775	**96.0**	918177
s-DG	88.9	3991412	88.9	3991412	**90.8**	8235507	**90.8**	8359379
DG	**90.6**	4143875	**90.6**	4143875	89.7	8520594	90.0	12111326
DA	94.2	2873827	97.3	5139594	93.3	6135433	**97.5**	11063183
s-DA	91.6	2140264	97.2	2659084	97.3	5984289	**97.6**	14635185
s-AG	51.8	1312908	63.5	2459459	68.3	3879676	**70.3**	11684373
WA	83.2	1768550	**88.9**	27734163	76.8	4233366	83.5	10383822
s-WA	82.1	1167243	**84.5**	4794199	82.4	3471388	82.4	3471388
average	88	1640176	**90.7**	5551414	89.0	3867599	**91.00**	7134215

the methods from different cliques. SDR consistently outperforms all, offering a significant enhancement over AutoML-EM.

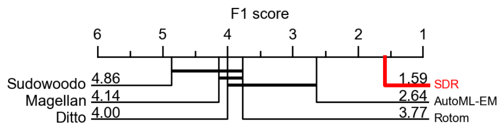


Fig. 4. Critical difference diagram of EM approaches.

4.3 Ablation Study

In this section, we decompose the modules in the SDR framework and analyze their effectiveness.

Similarity Library: Extended library(ours) V.S. AutoML-EM. We first compare the effectiveness and efficiency of our similarity library against the State-Of-The-Art machine learning approach, AutoML-EM [20]. With SDR Search and Bayesian Optimization, Table 4 shows that the F1 score of Extended library is averagely 2.9% higher than AutoML-EM library with slightly higher runtime for all datasets. Therefore, it is evident that subset partition can amplify the effectiveness of Extended library. This aligns with our design of using similarities for the antecedent of SDR: more similarities expand the solution space of subset discovery, leading to better performance.

Subset Discovery: SDR V.S. Clustering. We compare our subset discovery algorithms: DBSCAN-based Clustering and SDR, against the ablation RF-based EM model trained on the whole dataset without any subset discovery. As displayed in Table 4, SDR outperforms DBSCAN by 5.6% in terms of F1 score using Extended library. With the AutoML-EM library in Figure 2, the

Table 4. Ablation Study of Feature Library and Subset Discovery Method on Random Forest-based EM. For simplicity, we use "AE" for the Similarity library of AutoML-EM, while "ET" is used for Extended Similarity Library.

	AE		SDR-AE		Dbscan-ET		ET		SDR-ET		Dbscan-AE	
Dataset	F1	time	F1	time	F1	time	F1	time	F1	time	F1	time
FZ	**100.0**	2	**100.0**	2	97.6	2	**100.0**	2	**100.0**	2	**100.0**	1
IA	80.0	2	83.6	2	68.0	2	66.7	2	**87.5**	5	76.4	1
s-IA	93.5	1	93.5	1	98.4	2	98.4	1	98.4	1	**100.0**	1
BEER	88.2	1	91.3	1	88.2	2	82.4	1	**94.5**	1	78.6	1
DG	83.8	17	89.9	4042	89.3	27	83.9	18	**90.4**	183	88.6	53
DA	91.0	8	92.0	13	**94.0**	6	90.0	40	91.0	60	94.0	23
s-DG	92.5	24	93.2	185	95.0	36	92.7	26	**97.2**	930	95.5	54
s-DA	97.8	9	98.1	14	98.7	13	97.6	10	98.4	17	**99.0**	16
AG	12.2	4	13.1	6	52.1	8	49.9	32	**73.2**	52	32.1	16
WA	39.8	7	58.4	1010	51.0	15	44.5	9	**64.6**	230	42.9	11
s-WA	71.6	8	70.2	323	73.4	9	**80.5**	442	79.9	10	72.9	14
average	77.1	7	80.3	509	82.9	11	80.0	14	**88.6**	175	77.0	18

performance gap is still 3.3%. In conclusion, SDR can find more effective subsets with minor computation overhead, especially when equipped with extended library to enlarge the rule space.

4.4 Parameter Analysis

Finally, we test the sensitivity of parameter ρ, demonstrating the effectiveness of Bayesian optimization design module in SDR. We fix the similarity library and subset discovery as default and analyze the variations of the end-to-end performance and subset numbers in 11 EM benchmarks as ρ rises. Figure 5 shows our results. The selected ρ is the best of 50 rerun with different ρ in SDR, which can be seen as an oracle of the suboptimal solution for ρ.

Considering EM performance, ρ exhibits poor robustness. The variance in performance variation due to ρ goes up to 30%(Amazon-Google), while the variance in subset number variation is at most 14×. Also, the amount of subsets fluctuates severely as parameter increase. Therefore, the red line demonstrates SDR with the ρ of Bayesian Optimization, which dominates all five within only 10 iterations. In summary, the variant of SDR using ρ bayesian optimization can serve an automatic SDR module in application.

5 Conclusions

In conclusion, our proposed Subset Discovery Rule (SDR) approach effectively partitions training data into multiple subsets, allowing EM models to capture local data distributions more accurately. The SDR searching algorithm significantly reduces computational cost, making it feasible for practical applications. Evaluation results on 11 EM benchmarks demonstrate SDR's superiority over

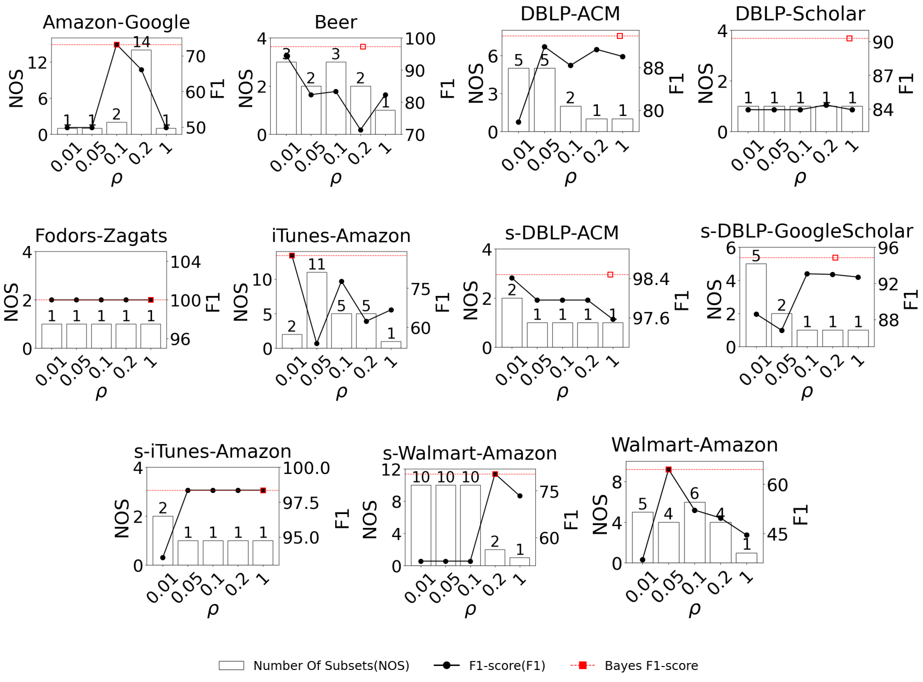


Fig. 5. Impact of ρ on F1 score and number of subsets divided by SDR

state-of-the-art methods, with notable improvements in F1 scores. SDR serves as a versatile plug-in for any machine learning-based EM model, promising future work in enhancing EM performance across diverse domains.

Acknowledgements

This work is supported by National Natural Science Foundation of China (NSFC) (62202126, 62232005), Natural Science Foundation Project of Heilongjiang Province (No YQ2024F005).

References

1. Bogatu, A., Paton, N.W., Douthwaite, M., Davie, S., Freitas, A.: Cost–effective variational active entity resolution. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE). pp. 1272–1283 (2021)
2. Brunner, U., Stockinger, K.: Entity matching with transformer architectures - a step forward in data integration. In: International Conference on Extending Database Technology (2020)
3. Brunner, U., Stockinger, K.: Entity matching with transformer architectures - a step forward in data integration. OpenProceedings (2020)

4. Chai, C., Li, G., Li, J., Deng, D., Feng, J.: A partial-order-based framework for cost-effective crowdsourced entity resolution. The VLDB Journal **27** (12 2018)
5. Chaudhuri, S., Chen, B.C., Ganti, V., Kaushik, R.: Example-driven design of efficient record matching queries. In: Proceedings of the 33rd International Conference on Very Large Data Bases. p. 327–338. VLDB '07, VLDB Endowment (2007)
6. Chen, S., Tang, N., Fan, J., Yan, X., Chai, C., Li, G., Du, X.: Haipipe: Combining human-generated and machine-generated pipelines for data preparation. Proc. ACM Manag. Data **1**(1) (May 2023)
7. Chen, S., Ding, X., Liang, Z., Tang, Y., Wang, H.: Hyper-parameter recommendation for truth discovery. In: Onizuka, M., Lee, J., Tong, Y., Xiao, C., Ishikawa, Y., Amer-Yahia, S., Jagadish, H.V., Lu, K. (eds.) Database Systems for Advanced Applications - 29th International Conference, DASFAA 2024, Gifu, Japan, July 2-5, 2024, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14852, pp. 277–292. Springer (2024). `https://doi.org/10.1007/978-981-97-5555-4_18`, `https://doi.org/10.1007/978-981-97-5555-4_18`
8. Das, S., G.C., P.S., Doan, A., Naughton, J.F., Krishnan, G., Deep, R., Arcaute, E., Raghavendra, V., Park, Y.: Falcon: Scaling up hands-off crowdsourced entity matching to build cloud services. In: Proceedings of the 2017 ACM International Conference on Management of Data. p. 1431–1446. SIGMOD '17, Association for Computing Machinery, New York, NY, USA (2017)
9. Ding, X., Lu, Y., Wang, H., Wang, C., Liu, Y., Wang, J.: Dafdiscover: Robust mining algorithm for dynamic approximate functional dependencies on dirty data. Proc. VLDB Endow. **17**(11), 3484–3496 (Aug 2024)
10. Ding, X., Wang, H., Su, J., Wang, M., Li, J., Gao, H.: Leveraging currency for repairing inconsistent and incomplete data. IEEE Trans. Knowl. Data Eng. **34**(3), 1288–1302 (2022). `https://doi.org/10.1109/TKDE.2020.2992456`, `https://doi.org/10.1109/TKDE.2020.2992456`
11. Doan, A., Halevy, A., Ives, Z.: Principles of Data Integration. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edn. (2012)
12. Ebaid, A., Thirumuruganathan, S., Aref, W.G., Elmagarmid, A., Ouzzani, M.: Explainer: Entity resolution explanations. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). pp. 2000–2003 (2019)
13. Ebraheem, M., Thirumuruganathan, S., Joty, S., Ouzzani, M., Tang, N.: Distributed representations of tuples for entity resolution. Proc. VLDB Endow. **11**(11), 1454–1467 (Jul 2018)
14. Jaro, M.A.: Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. Journal of the American Statistical Association **84**(406), 414–420 (1989)
15. Kang, R., Song, S., Wang, C.: Conditional regression rules. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE). pp. 2481–2493 (2022)
16. Konda, P., Das, S., Suganthan G. C., P., Doan, A., Ardalan, A., Ballard, J.R., Li, H., Panahi, F., Zhang, H., Naughton, J., Prasad, S., Krishnan, G., Deep, R., Raghavendra, V.: Magellan: toward building entity matching management systems. Proc. VLDB Endow. **9**(12), 1197–1208 (Aug 2016)
17. Li, P., Cheng, X., Chu, X., He, Y., Chaudhuri, S.: Auto-fuzzyjoin: Auto-program fuzzy similarity joins without labeled examples. In: Proceedings of the 2021 International Conference on Management of Data. p. 1064–1076. SIGMOD '21, Association for Computing Machinery, New York, NY, USA (2021)
18. Li, Y., Li, J., Suhara, Y., Doan, A., Tan, W.C.: Deep entity matching with pre-trained language models. Proc. VLDB Endow. **14**(1), 50–60 (Sep 2020)

19. Li, Z., Ding, X., Wang, H.: An effective constraint-based anomaly detection approach on multivariate time series. In: Wang, X., Zhang, R., Lee, Y., Sun, L., Moon, Y. (eds.) Web and Big Data - 4th International Joint Conference, APWeb-WAIM 2020, Tianjin, China, September 18-20, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12318, pp. 61–69. Springer (2020). https://doi.org/10.1007/978-3-030-60290-1_5, https://doi.org/10.1007/978-3-030-60290-1_5
20. Liang, Z., Wang, H., Ding, X., Mu, T.: Industrial time series determinative anomaly detection based on constraint hypergraph. Know.-Based Syst. **233**(C) (Dec 2021). https://doi.org/10.1016/j.knosys.2021.107548, https://doi.org/10.1016/j.knosys.2021.107548
21. Miao, Z., Li, Y., Wang, X.: Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond. In: Proceedings of the 2021 International Conference on Management of Data. p. 1303–1316. SIGMOD '21, Association for Computing Machinery, New York, NY, USA (2021)
22. M.K, V., K., K.: A survey on similarity measures in text mining (2016)
23. Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., Raghavendra, V.: Deep learning for entity matching: A design space exploration. In: Proceedings of the 2018 International Conference on Management of Data. p. 19–34. SIGMOD '18, Association for Computing Machinery, New York, NY, USA (2018)
24. Peeters, R., Bizer, C.: Dual-objective fine-tuning of bert for entity matching. Proc. VLDB Endow. **14**(10), 1913–1921 (Jun 2021)
25. Shahbazi, N., Danevski, N., Nargesian, F., Asudeh, A., Srivastava, D.: Through the fairness lens: Experimental analysis and evaluation of entity matching. Proc. VLDB Endow. **16**(11), 3279–3292 (Jul 2023)
26. Shahbazi, N., Wang, J., Miao, Z., Bhutani, N.: Fairness-aware data preparation for entity matching. In: 2024 IEEE 40th International Conference on Data Engineering (ICDE). pp. 3476–3489 (2024)
27. Steorts, R.C., Ventura, S.L., Sadinle, M., Fienberg, S.E.: A comparison of blocking methods for record linkage. In: Privacy in Statistical Databases (2014)
28. Tu, J., Fan, J., Tang, N., Wang, P., Li, G., Du, X., Jia, X., Gao, S.: Unicorn: A unified multi-tasking model for supporting matching tasks in data integration. Proc. ACM Manag. Data **1**(1) (May 2023)
29. Tu, J., Han, X., Fan, J., Tang, N., Chai, C., Li, G., Du, X.: Dader: hands-off entity resolution with domain adaptation. Proc. VLDB Endow. **15**(12), 3666–3669 (Aug 2022)
30. Wang, H., Ding, X., Li, J., Gao, H.: Rule-based entity resolution on database with hidden temporal information. IEEE Trans. Knowl. Data Eng. **30**(11), 2199–2212 (2018). https://doi.org/10.1109/TKDE.2018.2816018, https://doi.org/10.1109/TKDE.2018.2816018
31. Wang, P., Zheng, W., Wang, J., Pei, J.: Automating entity matching model development. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE). pp. 1296–1307 (2021)
32. Wang, R., Li, Y., Wang, J.: Sudowoodo: Contrastive self-supervised learning for multi-purpose data integration and preparation. In: 2023 IEEE 39th International Conference on Data Engineering (ICDE). pp. 1502–1515 (2023)
33. Wu, R., Chaba, S., Sawlani, S., Chu, X., Thirumuruganathan, S.: Zeroer: Entity resolution using zero labeled examples. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. p. 1149–1164. SIGMOD '20, Association for Computing Machinery, New York, NY, USA (2020)