Entity Matching with Quality and Error Guarantees

Yufei Tao[*]
Chinese University of Hong Kong
Hong Kong
taoyf@cse.cuhk.edu.hk

ABSTRACT

Given two sets of entities X and Y, *entity matching* aims to decide whether x and y represent the same entity for each pair $(x, y) \in X \times Y$. In many scenarios, the only way to ensure perfect accuracy is to launch a costly inspection procedure on every (x, y), whereas performing the procedure $|X| \cdot |Y|$ times is prohibitively expensive. It is, therefore, important to design an algorithm that carries out the procedure on only some pairs, and renders the verdicts on the other pairs automatically with as few mistakes as possible. This article describes an algorithm that achieves the purpose using the methodology of *active monotone classification*. The algorithm ensures an asymptotically optimal tradeoff between the number of pairs inspected and the number of mistakes made.

1. INTRODUCTION

Given two sets of entities X and Y, *entity matching* aims to decide whether x and y form a *match*, i.e., whether they represent the same entity, for each pair $(x, y) \in X \times Y$. For example, X (or Y) can be a set of advertisements placed at *Amazon* (or *eBay*, resp.), where each advertisement has attributes like `prod-name`, `prod-description`, `year`, `price`, and so on. The goal is to decide whether advertisements x and y are about the same product, for all $(x, y) \in X \times Y$.

What makes the problem challenging is that the aforementioned decision cannot be made through a simple comparison on the attributes of x and y, because even a pair of matching x and y may still disagree on the attribute values. This is obvious for `prod-description` and `price` since x and y can introduce the same product in different ways, and price it differently. In fact, x and y may not agree even on a "supposedly standardized" attribute like `prod-name` (e.g., x.`prod-name` = "MS Word" vs. y.`prod-name` = "Microsoft Word Processor"), although it would be reasonable to expect x.`year` = y.`year` because advertisements are required to be correct.

In applications like the above one, the only way to ensure perfect accuracy is to call upon human experts to inspect each pair $(x, y) \in X \times Y$. This is, however, expensive because the amount of manual efforts, e.g., reading advertisements x and y in detail, is huge. It is, therefore, important to design an algorithm that asks humans to look at only some pairs, and renders the verdicts on the other pairs automatically, perhaps at the expense of a small number of errors.

Towards the above purpose, a dominant methodology behind existing approaches (e.g., [1, 3, 6, 7, 12, 14, 21, 22, 24, 25]) is to transform the task into a multidimensional classification problem with the following preprocessing:

1. First, shrink the set of all possible pairs to a subset $T \subseteq X \times Y$, by eliminating the pairs that obviously cannot be matches. This is known as *blocking*, which is carried out based on application-dependent heuristics. This step is optional; if skipped, then $T = X \times Y$. In the Amazon-eBay example, T may involve only those advertisement pairs (x, y) with x.`year` = y.`year`.

2. For each remaining entity pair $(x, y) \in T$, create a multidimensional point $p_{(x,y)}$ using a number d of similarity functions

   $$sim_1, sim_2, \ldots, sim_d,$$

   each evaluated on a certain feature. The i-th coordinate of $p_{(x,y)}$ equals $sim_i(x, y)$: a higher value means that x and y are more similar on the i-th feature. This creates a d-dimensional point set $P = \{p_{(x,y)} \mid (x, y) \in T\}$.

   In our example, from a numerical attribute such as `price`, one may extract a feature that equals $-|x.\texttt{price} - y.\texttt{price}|$ (the purpose of the negation is to be consistent with "larger means more similar"). From a text attribute (such as `prod-name` and `prod-description`) one may extract a feature by evaluating the similarity between the corresponding texts of x and y using an appropriate metric, e.g., edit distance for short texts, and cosine similarity for long texts. Multiple feature dimensions may be derived even on the same attribute. For instance, one can extract two features by computing the edit-distance and Jaccard-distance of x.`prod-name` and y.`prod-name` separately.

Every point $p_{(x,y)} \in P$ carries a *label*, which is 1 if (x, y) is

The original version of this article was published in PODS 2018, titled "Entity Matching with Active Monotone Classification".

[*]This work was partially supported by a direct grant (Project Number: 4055079) from CUHK and by a Faculty Research Award from Google.

a match, or 0 otherwise. The original entity matching task on X and Y is thus converted to inferring the labels of the points in P. Still, human inspection is the last resort for revealing the label of each $p_{(x,y)}$ with no errors.

1.1 Problem Formalization

Let P be a set of points in $\mathbb{R}^d$, where $\mathbb{R}$ is the real domain, and d is a positive integer. Each point $p \in P$ carries a label, denoted as *label*(p), which equals either 0 or 1. The point labels need not follow any geometric patterns, namely, *label*(p) can be 0 or 1 regardless of the labels of the other points.

All the labels are hidden at the beginning. There is an *oracle* which an algorithm can call to disclose the label of a point $p \in P$ selected by the algorithm. When this happens, we say that p is *probed*. The algorithm's *cost* is defined as the number of points probed.

A *classifier* is a function $\mathcal{F} : \mathbb{R}^d \to \{0, 1\}$. Its *error* on P is the number of points mis-labeled, namely:

$$\mathit{error}(\mathcal{F}, P) = |\{p \in P \mid \mathcal{F}(p) \neq \mathit{label}(p)\}|.$$

$\mathcal{F}$ is *monotone* if $\mathcal{F}(p) \geq \mathcal{F}(q)$ for any points $p, q \in P$ satisfying $p[i] \geq q[i]$ on all $i \in [1, d]$, where $p[i]$ is the i-th coordinate of p. Denote by $\mathbb{C}_{mono}$ the set of all possible monotone classifiers; note that $|\mathbb{C}_{mono}|$ is infinite.

We consider two problems closely related to *active learning*:

Problem 1 (Active Monotone Classification): Find a monotone classifier $\mathcal{F}$ with small $\mathit{error}(\mathcal{F}, P)$ by paying a low probing cost.

Problem 2 (Active Monotone Classification with Exemptions): Probe a small set Z of points in P to find a monotone classifier $\mathcal{F}$ with small $\mathit{error}(\mathcal{F}, P \setminus Z)$.

Note that the two problems differ in whether the set Z of points probed is exempted from calculating the error of the returned classifier $\mathcal{F}$. The challenges behind these problems can be seen from the following extreme solutions:

- For Problem 1, one can simply probe all the points of P, paying the worst possible cost $|P|$, and then take all the time needed to find the best classifier $\mathcal{F}^* \in \mathbb{C}_{mono}$ with the smallest error on P (we do not explicitly constrain CPU computation). Note that the error of $\mathcal{F}^*$ need *not* be zero because P may not—actually most likely will not—fully obey monotonicity. In the other extreme, we can choose to probe nothing and return some classifier $\mathcal{F}$ by "wild guessing", which has the smallest cost 0, but risks suffering from the worst error $|P|$ for $\mathcal{F}$. The main challenge is to achieve the lowest error with as few probes as possible.

- For Problem 2, we can trivially achieve the minimum value 0 for $\mathit{error}(\mathcal{F}, P \setminus Z)$ by probing all the points of P, noticing that in this case $P \setminus Z = \emptyset$. In the other extreme, one can return a wild guess $\mathcal{F}$ with no probes at all, but again may suffer from the largest error $|P|$ for $\mathit{error}(\mathcal{F}, P \setminus Z)$. The main challenge is to strike an attractive tradeoff between $|Z|$ and $\mathit{error}(\mathcal{F}, P \setminus Z)$.

The above definitions extend in a natural way to a randomized algorithm A_{ran}. Both the classifier $\mathcal{F}$ returned by A_{ran} and the set Z of points probed by A_{ran} are random variables. For Problem 1, the *expected error* of A_{ran} is defined as $\mathbf{E}[\mathit{error}(\mathcal{F}, P)]$, and its *expected cost* as $\mathbf{E}[|Z|]$. Likewise, for Problem 2, the *expected error* of A_{ran} is defined as $\mathbf{E}[\mathit{error}(\mathcal{F}, P \setminus Z)]$, and its *expected cost* still as $\mathbf{E}[|Z|]$. In all cases, expectation is over the random choices made by the algorithm.

Remarks. The input P to both problems corresponds to the set of points obtained from the set T in the entity matching framework explained earlier. Problems 1 and 2 are designed for two different scenarios that arise frequently in practice:

- Scenario 1: the entity sets X and Y are "training sets" that represent the distributions D_X and D_Y of entities to be encountered from two sources, respectively. The purpose of finding a classifier $\mathcal{F}$ is to apply it on new $(\tilde{x}, \tilde{y}) \notin X \times Y$ to be received online where $\tilde{x}$ (or $\tilde{y}$) follows D_X (or D_Y, resp.). That $\mathcal{F}$ is accurate on P (a.k.a. T) implies that $\mathcal{F}$ should also work statistically well on $(\tilde{x}, \tilde{y})$. This is the application backdrop of Problem 1.

- Scenario 2: unlike the previous scenario, here X and Y are already the "ground sets". In other words, there are no future pairs, such as $(\tilde{x}, \tilde{y})$ in Scenario 1, to be cared for; and it suffices to match only the elements of X and Y. Thus, the "overall accuracy" of $\mathcal{F}$ on all the points in P is unimportant because if a point already has its label revealed, it does not need to be guessed by $\mathcal{F}$, and thus should be excluded from error calculation. This is the application backdrop behind Problem 2.

The rationale behind monotonicity is that, if x and y form a match according to the features picked, then any pair (x', y') more similar than (x, y) on *every* feature should also be regarded as a match. Indeed, any classifier that defies monotonicity is *awkward* because it indicates that at least some of the features have been selected inappropriately.

1.2 Relevant Research

This subsection will first give an introduction to several key findings in active learning that are relevant to Problems 1 and 2, and then review the existing entity matching solutions we are aware of.

Active Learning. Classification is a fundamental topic in machine learning. Let U be a possibly infinite set of points in $\mathbb{R}^d$. The input is an infinite stream of pairs $(p, \mathit{label}(p))$ where p is a point from U, and *label*(p) is its binary label, i.e., either 0 or 1. Each pair is sampled independently from an unknown distribution D on $U \times \{0, 1\}$. A *classifier* is a function $\mathcal{F} : U \to \{0, 1\}$, whose *error probability* equals

$$\mathbf{Pr}\{p \sim D \mid \mathcal{F}(p) \neq \mathit{label}(p)\}$$

namely, the probability of wrongly predicting the label of a point p drawn from D. Let $\mathbb{C}$ be a *candidate class* of classifiers, and ν be the smallest error probability of all the classifiers in $\mathbb{C}$. The learning objective is the achieve the following *probabilistically approximately correct* (PAC) guarantee:

> With probability at least $1 - \delta$, find a classifier $\mathcal{F}$ from $\mathbb{C}$ with error probability at most $\nu + \epsilon$, where $\delta > 0$ and $\epsilon > 0$ are input parameters.

In the more traditional *passive* setup, $label(p)$ is directly disclosed in every pair $(p, label(p))$, where the efficiency goal is to minimize the *sample cost*, which equals the number of stream pairs that an algorithm needs to see before ensuring the PAC guarantee. In practice, deciding the label of a point can be so expensive that its cost far dominates the cost of learning. This motivated *active* learning, where point labels are all hidden originally. Given an incoming point p, an algorithm can choose whether to *probe* p, i.e., paying a unit cost for the revelation of $label(p)$. The primary efficiency goal is to perform the least number of probes; efforts should still be made to avoid a high sample cost, although this now becomes a secondary goal.

Active learning has been extensively studied; see excellent surveys [16, 23]. A main challenge is to identify the *intrinsic parameters* that determine the *label complexity*, i.e., the number of probes mandatory to ensure the PAC guarantee. Considerable progress has been made in various scenarios [4, 8, 15]. Our subsequent discussion will concentrate on *agnostic* active learning, where (i) $\nu > 0$, meaning that even the best classifier in $\mathbb{C}$ cannot perfectly separate points of the two labels because, intuitively, D has "noise", and (ii) no assumptions are made on that noise. This is the branch of active learning most relevant to our work.

The state-of-the-art understanding on agnostic learning is based on two intrinsic parameters:

- the *VC dimension* λ of $\mathbb{C}$ on U;
- the *disagreement coefficient* θ of $\mathbb{C}$ under D.

We will not delve into the precise definitions of the above parameters (the interested reader may see [24] for details). For this article, it suffices to understand that a higher λ or θ indicates the necessity of probing more labels.

The dominant solution to agnostic active learning is an algorithm named A^2. Its initial ideas were developed by Balcan et al. [2], and have been substantially improved/extended subsequently [4, 9, 16]. As shown in [16], the algorithm achieves the PAC guarantee with a probing cost of

$$\tilde{O}\left(\theta \cdot \lambda \cdot \frac{\nu^2}{\epsilon^2}\right) \tag{1}$$

where $\tilde{O}(.)$ hides factors polylogarithmic to $\theta, 1/\epsilon$, and $1/\delta$. On the lower bound side, extending an earlier result of Kaariainen [17], Beygelzimer et al. [4] proved that the probing cost needs to be

$$\Omega\left(\frac{\nu^2}{\epsilon^2} \cdot \left(\lambda + \log\frac{1}{\delta}\right)\right). \tag{2}$$

There is a gap of θ between the upper and lower bounds. When this parameter is $O(1)$, the two bounds match up to polylog factors. Indeed, most success stories in the literature are based on candidate classes $\mathbb{C}$ and distributions D that give rise to a small θ (e.g., see [8, 10, 13, 26]).

Unfortunately, as will be explained later in Section 1.3, the class $\mathbb{C}_{mono}$ of monotone classifiers can have a very high VC dimension λ, and simultaneously, a very large disagreement coefficient θ. The consequences are two-fold:

- The θ gap between (1) and (2) becomes significant, suggesting that agnostic active learning has not been well understood on monotone classifiers.
- With both θ and λ being large, the A^2 algorithm incurs expensive probing costs on $\mathbb{C}_{mono}$, and may no longer be attractive.

The reader may have noticed that Problem 1 can be cast as agnostic active learning by setting U (in active learning) to P (in Problem 1), and generating an input stream (for active learning) by repeatedly sampling P. This makes A^2 a viable solution to entity matching. We will discuss its performance guarantees in relation to our results in the next subsection.

Entity Matching. There is a rich literature on entity matching; see [1, 3, 5–7, 12, 14, 18–22, 25] and the references therein. Most of these works focused on designing heuristics that perform well in practice, instead of establishing theoretical bounds. The papers [1, 3] are exceptions. In [1], Arasu et al. observed that entity matching can be approached using active learning. They presented algorithms to solve some subproblems that arose in their framework. Unfortunately, their overall solutions do not have attractive bounds for Problem 1 or 2. In [3], Bellare et al. showed that if one can solve Problem 1, the algorithm can be utilized to attain small errors of other types, e.g., those based on recalls and precisions, under certain assumptions.

1.3 Our Contributions

An Intrinsic Parameter. Recall that Problems 1 and 2 are defined on a set P of points in $\mathbb{R}^d$. Given two points $p, q \in P$, we say that p *dominates* q if $p[i] \geq q[i]$ holds on all $i \in [1, d]$. Notice that a point dominates itself by this definition. The dominance relation

$$R = \{(p, q) \in P \times P \mid p \text{ dominates } q\}$$

is a poset (partially ordered set).

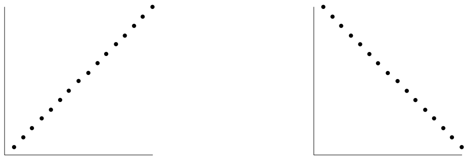
It turns out that an intrinsic parameter characterizing the hardness of both problems is the *width* w of R. Formally, w is the size of the largest $S \subseteq P$ such that no two different points in S dominate each other; we will sometimes refer to it as the *dominance width* of P. Any one-dimensional P has $w = 1$. When $d \geq 2$, w can be anywhere between 1 and n; see Figure 1 for two extreme examples.

Problem 1. Denote by $\mathcal{F}^*$ an optimal monotone classifier on P, namely, this is a classifier in $\mathbb{C}_{mono}$ with the smallest error on P. Set $k = error(\mathcal{F}^*, P)$ and $n = |P|$ throughout the article. Our first main result is:

Theorem 1. *For Problem 1:*

- *there is a randomized algorithm that has expected error at most $2k$, and probes $O(w(1 + \log \frac{n}{w}))$ points in expectation;*
- *there exists a constant c such that, when $w \geq 2$ and $k \leq cn/w$, any algorithm with expected error $O(k)$ must have an expected probing cost of $\Omega(w \log \frac{n}{(k+1)w})$.*

Several observations can be made. First, when $k = 0$—the *noise-free* scenario where the label-1 points of P can be perfectly separated from the label-0 ones by a monotone classifier—our algorithm in the first bullet always returns such a classifier. Second, when $w = \Omega(n)$, our lower bound in the second bullet evaluates to $\Omega(n)$, meaning that the



(a) A point set of width 1 (b) A point set of width n

Figure 1: Illustration of dominance width

naive solution of probing all points in P can no longer be improved by more than a constant factor in cost, for the purpose of ensuring the smallest error asymptotically. Third, we improve the aforementioned naive solution as long as $w = o(n)$, because for such w the upper bound in the first bullet is $o(n)$. Fourth, our upper and lower bounds nearly match each other for $k = O(n/w)$. Remember that no algorithms can have an expected error less than k. Therefore, under $k = O(n/w)$, our solution is asymptotically optimal in both expected error and expected probing cost.

As explained in Section 1.2, one can apply the A^2 algorithm of agnostic active learning to solve Problem 1 by repeatedly sampling from P uniformly. To see its performance, let us fit in the appropriate values for ν, ϵ, λ, and θ, as are defined in Section 1.2. First, $\nu = k/n$, according to the definition of $error(\mathcal{F}^*, P)$. Second, to match our expected error $2k$, ϵ should be no more than $\nu = k/n$; setting ϵ to this value makes $\nu^2/\epsilon^2 = 1$. As we proved in [24], when $k = O(n/w)$, both θ and λ can reach w even in 2D space. By (1), A^2 has expected probing cost $\tilde{O}(w^2)$, which Theorem 1 improves by a factor of $\tilde{O}(w)$.

Note that (2) does *not* give a lower bound on Problem 1. Recall that (2) applies to agnostic active learning which is just one possible way to approach Problem 1.

Problem 2. If an algorithm returns a monotone classifier $\mathcal{F}$ by probing a set Z of points, it always holds that $error(\mathcal{F}, P \setminus Z) \leq error(\mathcal{F}, P)$. Hence, Theorem 1 implies:

Corollary 1. *For Problem 2, there is a randomized algorithm that has expected error at most $2k$, and expected probing cost $O(w(1 + \log \frac{n}{w}))$.*

What is intriguing is the opposite: can we substantially reduce the error of Problem 2 without significantly increasing the probing cost? This, subtly, is a question on the usefulness of *exempting* Z from the error calculation. After all, the intended purpose of Z is to push $error(\mathcal{F}, P \setminus Z)$ *below* k (recall that, in Problem 1, the best achievable error is k). Unfortunately, we are able to show:

Theorem 2. *Fix any integers k and n such that $k \geq 1$, and n/k is an integer at least 2. There is a set S of one-dimensional (i.e., $d = 1$) inputs to Problem 1 with the same n and k such that any algorithm guaranteeing an expected error at most $k/2$ on every input in S must entail an expected probing cost of $\Omega(n/k)$ on at least one input in S.*

Corollary 1 and Theorem 2 together point out a surprising fact. If we are satisfied with an expected error of $2k$, the expected probing cost is no more than $O(w(1 + \log \frac{n}{w}))$ universally for *all* values of k. Even better, in the context of Theorem 2 where $d = 1$, w equals 1, making $O(w(1 + \log \frac{n}{w})) = O(\log n)$. However, if we demand an expected error of $k/2$, the expected probing cost surges to $\Omega(n/k)$, which is worse than $O(\log n)$ for any $k = o(n/\log n)$.

Theorem 2 also implies that, for $k \leq n/(w \log_2 \frac{n}{w})$, the expected error must be at least $\Omega(k)$ if the expected probing cost has to be $O(w \log(n/w))$. Thus, on those values of k, our algorithm in Corollary 1 is already asymptotically optimal. Phrased differently, subject to $O(w \log \frac{n}{w})$ expected probing cost, the hardness of the problem comes from guessing the labels of points that have not been probed, such that excluding Z from error calculation makes little difference.

Content of This Article. We will focus on establishing the upper bound for Problem 1 in Theorem 1 by describing our algorithm and its analysis in full. The proofs for the lower bounds in Theorems 1 and 2, which can be found in [24], are omitted from the article.

2. THE PROPOSED ALGORITHM

Our algorithm for Problem 1—named *random probe with elimination* (RPE)—can be described in 6 lines as shown in Figure 2. If Z is the set of points probed, the algorithm produces a classifier $\mathcal{F}$ defined as:

$$\mathcal{F}(p) = \begin{cases} 1 & \text{if } p \text{ dominates a label-1 point in } Z \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

To illustrate, consider that P consists of the 16 points in Figure 3, where the black (or white) points carry label 1 (or 0, resp.). Here, k equals 3, noticing that no monotone classifiers can have an error 2 or less, while it is easy to design a monotone classifier with error 3, e.g., such a classifier would correctly capture the labels of all points except p_1, p_{11}, and p_{15}. Assume that, at Step 2, RPE happens to probe p_1 first, after which it eliminates the entire P except p_6, p_7, and p_8. Suppose also that, when Step 2 is executed again, the algorithm chooses to probe p_8, which removes all the remaining points in P. With $Z = \{p_1, p_8\}$, the classifier of (3) has an error 5 because it incorrectly maps p_2, p_3, p_5, p_{11}, and p_{15} to 1.

Lemma 1. *The classifier $\mathcal{F}$ in (3) is monotone.*

Proof. Suppose that there exist points p, q such that p dominates q, but $\mathcal{F}(p) = 0$ and $\mathcal{F}(q) = 1$. By (3), $\mathcal{F}(q) = 1$ means that Z has a label-1 point that is dominated by q, and hence, also dominated by p. This contradicts $\mathcal{F}(p) = 0$. □

The next lemma, together with (3), indicates a sense of symmetry between labels 0 and 1 with respect to the classifier $\mathcal{F}$ returned by RPE.

Lemma 2. *For any $p \in P$, $\mathcal{F}(p) = 0$ if and only if p is dominated by a label-0 point in Z.*

```
Algorithm RPE($P$)
/* $P$ is the input set of Problem 1 */
1. while $P \neq \emptyset$
2.     pick a point $p$ from $P$ uniformly at random
3.     probe $p$
4.     if $label(p) = 1$ then
5.         discard from $P$ the points dominating $p$
       else
6.         discard from $P$ the points that $p$ dominates
```

Figure 2: The RPE algorithm

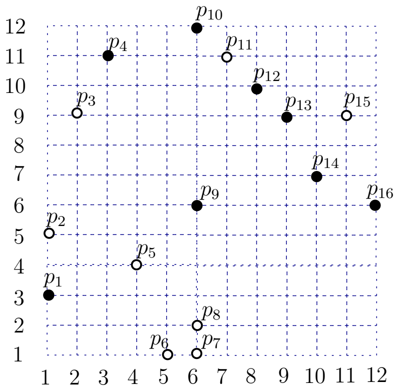


Figure 3: An input to Problem 1 where $k = 3$; black and white points have label 1 and 0, respectively.

Proof. By (3), $\mathcal{F}(p) = 0$ means that p does not dominate any label-1 point in Z. Hence, the deletion of p from P must have been triggered by RPE probing a label-0 point p' dominating p (note that p' could be p and that a point is considered to dominate itself). This proves the "only-if" direction.

To establish the "if" direction, we first need to prove that Z obeys monotonicity, namely, for any $p, q \in Z$, if p dominates q, then $label(p) \geq label(q)$. Assume, on the contrary, that this is not true, meaning that $label(p) = 0$ and $label(q) = 1$. If p was probed before q, then q must have been discarded after discovering that p has label 0. Likewise, if q was probed before p, then p must have been discarded after discovering that q has label 1. This contradicts the fact that both p and q were probed.

The monotonicity of Z suggests that, if p is dominated by a label-0 point in Z, p cannot dominate any label-1 point in Z. Hence, $\mathcal{F}(p) = 0$, thus establishing the "if" direction. □

RPE can be implemented in $O(n \operatorname{polylog} n)$ time for fixed dimensionality d. By maintaining P in a binary search tree, we can draw a random point $p \in P$ at Step 2 in $O(\log n)$ time, such that in total we spend $O(n \log n)$ time on this step. By maintaining a range tree [11] on P, Steps 5 and 6 can be implemented in $O(x \log^d n)$ time, if x points are eliminated from P. Each point contributes to the x-term exactly once (it can be deleted only once). Hence, the total CPU time spent on these two steps is bounded by $O(n \log^d n)$.

The above is everything that a practitioner needs to know in order to apply the RPE algorithm in practice. We will delve into the theory behind RPE in the next two sections.

3. ANALYZING THE COST OF RPE

An Attrition-and-Elimination Game. We will now take a detour to discuss a relevant problem. Consider the following game between two players Alice and Bob. At the beginning, Alice is given the set $S = \{1, 2, \dots, s\}$ for some integer $s \geq 1$. The game goes in *rounds*. In each round:

- Bob performs "attrition" first, by either doing nothing or arbitrarily deleting some elements from S.
- Alice then performs "elimination" by picking a number $p \in S$ uniformly at random, and deleting from S all the numbers larger than or equal to p.

The game ends when S becomes empty. The number of rounds is a random variable depending on Bob's strategy. How should Bob play in order to maximize the expectation of that variable?

It is fairly intuitive that Bob should do nothing at all in every round, in which case the expected number of rounds is $\Theta(1 + \log s)$. To prove this, denote by function $f(s)$ the *largest* real number such that Bob has a strategy to make the expected number of rounds equal $f(s)$. Consider the first attrition of Bob. Clearly, what matters is the number x of elements that Bob decides to remove (what those elements actually are is not relevant due to symmetry). If $x < s$, the game continues for Alice to work on a set S of size $s - x$. After her elimination, S has i elements—for each $i \in [0, s - x - 1]$—left with probability $1/(s - x)$. Therefore:

$$f(s) \quad = \quad 1 + \max_{x=0}^{s-1} \left\{ \frac{1}{s-x} \sum_{i=0}^{s-x-1} f(i) \right\}.$$

As the base case, $f(0) = 0$. Solving the recurrence gives $f(s) = O(1 + \log s)$ for $s \geq 1$, regardless of the value of x.

Dominance Width and Chain Decomposition. Let us return to Problem 1. Given a non-empty subset C of the input P, we say that C is:

- A *chain*, if it is possible to linearize the points of C into a sequence $p_1, p_2, \dots, p_{|C|}$ such that p_{i+1} dominates p_i for every $i \in [1, |C| - 1]$. We will refer to the sequence as the *ascending order* of C.
- An *anti-chain*, if none of the points in C dominate each other.

In Figure 3, $\{p_6, p_8, p_{10}\}$ is a chain, whereas $\{p_4,\ p_{12},\ p_{13},\ p_{14}\}$ is an anti-chain.

A *chain decomposition* is a collection of disjoint chains C_1, C_2, ..., C_t (for some $t \geq 1$) whose union equals P. How to determine the smallest number t is a fundamental result in order theory:

> **Dilworth's Theorem:** Consider any poset; let w be the largest size of all anti-chains. Then, (i) there is a chain decomposition that contains w chains, and (ii) no chain decompositions can have less than w chains.

For instance, the input set P of Figure 3 can be divided into 6 chains: $C_1 = \{p_1, p_2, p_3, p_4, p_{10}\}$, $C_2 = \{p_{11}\}$, $C_3 = \{p_5, p_9, p_{12}\}$, $C_4 = \{p_{16}\}$, $C_5 = \{p_{13}\}$, and $C_6 = \{p_6, p_7, p_8, p_{14}, p_{15}\}$. This is a smallest chain decomposition due to the anti-chain $\{p_{10}, p_{11}, p_{12}, p_{16}, p_{13}, p_{14}\}$. Hence, the dominance width w of P is 6.

Probing Cost of RPE. Let $\{C_1, C_2, \dots, C_w\}$ be an arbitrary smallest chain decomposition (which is *not* known to RPE). We will prove that in expectation RPE probes $O(1 + \log |C_i|)$ points in C_i for all $i \in [1, w]$. It will then follow that the total expected number of probes is $O(\sum_{i=1}^{w} (1 + \log |C_i|))$, which peaks at $O(w(1 + \log \frac{n}{w}))$ when all the chains have the same size n/w.

Without loss of generality, let us focus on C_1. Break C_1 into (i) the set C_1^{true} of points with label 1, and (ii) the set C_1^{false} of points with label 0. Due to symmetry, it suffices to prove that RPE probes $O(1 + \log |C_1^{true}|)$ points from C_1^{true} in expectation.

Set $s = |C_1^{true}|$. List the points of C_1^{true} in ascending order $p_1, p_2, \dots, p_s$. The operations that RPE performs on this chain can be captured as an attrition-and-elimination game on an initial input $S = \{p_1, p_2, \ldots, p_s\}$:

- Bob formulates his strategy by observing the execution of RPE. Suppose that the algorithm probes a point $p \notin C_1^{true}$, and shrinks P at Step 5 or 6. Bob deletes from S all those points of C_1^{true} that are discarded in the shrinking.

- When RPE probes a point $p \in C_1^{true}$, Bob finishes his attrition in this round, and passes S to Alice. *Conditioned* on $p \in C_1^{true}$, p was chosen uniformly at random from the *current* S, i.e., the set of points from C_1^{true} that are still in P. Hence, p can be regarded as the choice of Alice. When RPE shrinks P at Step 5, Alice discards p, as well as all the points behind p, from S. This finishes a round of the game. The control is passed back to Bob to start the next round.

By our earlier analysis on the attrition-and-elimination game, RPE probes $O(1 + \log s)$ points from C_1^{true} in expectation. This establishes the upper bound in Theorem 1 on the cost of RPE.

4. ANALYZING THE ERROR OF RPE

We now proceed to analyze the number of points mislabeled by the classifier $\mathcal{F}$ returned by RPE.

Fix an arbitrary optimal monotone classifier $\mathcal{F}^*$, i.e., $k = error(F^*, P)$. Henceforth, a point $p \in P$ is said to be an *ordinary* point if $\mathcal{F}^*(p) = label(p)$, or a *noise* point, otherwise. Define:

$$\begin{aligned} G_1^* &= \{p \in P \mid label(p) = 1 \text{ and } p \text{ is ordinary}\} \\ G_0^* &= \{p \in P \mid label(p) = 0 \text{ and } p \text{ is ordinary}\}. \end{aligned}$$

Since P unions G_1^*, G_0^*, and the k noise points, we know:

$$error(\mathcal{F}, P) \le error(\mathcal{F}, G_1^*) + error(\mathcal{F}, G_0^*) + k. \qquad (4)$$

Let k_0 be the number of label-0 noise points, that is, points p satisfying $label(p) = 0$ but $\mathcal{F}^*(p) = 1$. The rest of the section serves as a proof of:

LEMMA 3. $\mathbf{E}[error(\mathcal{F}, G_1^*)]$ *is at most* k_0.

By the symmetry shown in Lemma 2, the above lemma implies that $\mathbf{E}[error(\mathcal{F}, G_0^*)]$ is at most the number k_1 of label-1 noise points. Equation (4) then gives $\mathbf{E}[error(\mathcal{F}, P)] \le k_0 + k_1 + k = 2k$, thus establishing the upper bound in Theorem 1 on the error of RPE.

4.1 RPE by Permutation

To analyze $error(\mathcal{F}, G_1^*)$, it will be convenient to consider an alternative implementation of RPE named *RPE-perm*, which is described in Figure 4. Compared to RPE, RPE-perm differs only in how randomization is injected: this is now done by randomly permuting P. We defer the proof of the lemma below to Section 4.4.

LEMMA 4. *RPE and RPE-perm have the same expected error and expected probing cost on every input P.*

```
Algorithm RPE-perm($P$)
1. randomly permute the points of $P$
   /* if a point $p \in P$ is the $i$-th ($i \in [1, n]$) in the
   permutation, define its rank $r(p)$ to be $i$ */
2. while $P \neq \emptyset$
3.     pick the point $p \in P$ with the smallest rank
4.     probe $p$
5.     if $label(p) = 1$ then
6.         discard from $P$ the points dominating $p$
       else
7.         discard from $P$ the points that $p$ dominates
```

Figure 4: The permutation-version of RPE

4.2 Influence of Noise Points

Let us first gain some intuition on why $error(\mathcal{F}, G_1^*)$ is small in expectation. Consider the example in Figure 3, and the optimal $\mathcal{F}^*$ that mis-labels only p_1, p_{11}, and p_{15}. Here, G_1^* consists of all the black points except p_1. The bad news is that, if noise point p_{15} is probed first, the classifier $\mathcal{F}$ output by RPE will map the ordinary points p_9, p_{13}, p_{14} to 0 incorrectly (see Lemma 2), causing an increase of 3 to $error(\mathcal{F}, G_1^*)$. The good news is that, if p_{15} is probed after *any* of the ordinary points p_9, p_{13}, p_{14}, then p_{15} will be discarded and can do no harm. Under a random permutation, p_{15} has only 1/4 probability to rank before all of p_9, p_{13}, p_{14}, which seems to suggest that p_{15} could trigger an increase of only 3/4 to $error(\mathcal{F}, G_1^*)$ in expectation.

Unfortunately, the analysis is not as simple as this, due to the presence of noise point p_{11}, which complicates the conditions for p_{15} to be probed. For example, observe that, if p_{11} did not exist, p_{15} can never be probed when p_9 ranks before p_{15}. This is no longer true with the presence of p_{11}. To see this, imagine that p_{11} ranks before p_9, which in turn ranks before p_{15}. The probing of p_{11} evicts p_9 from P. On the other hand, p_{15} remains in P, and hence, gets a chance to be probed later.

The above issue arises because p_9 is dominated—and thereby is "influenced"—by both noise points p_{11} and p_{15}. *Separating* and *quantifying* the influence of each noise point turns out to be the most crucial idea behind our analysis. Let N_0 be the set of label-0 noise points, i.e., $k_0 = |N_0|$. Next, we will describe a way to calculate the "exclusive influence" $I(q)$ of each point $q \in N_0$. In particular, we will do so incrementally by observing how RPE-perm executes.

At the beginning, initialize $I(q) = 0$ for every $q \in N_0$. Whenever RPE-perm is *about* to probe a point $q \in N_0$, capture the set—denoted as $P(q)$—of points that are still in P at this moment. Then, finalize $I(q)$ as:

$$I(q) \quad = \quad \begin{array}{l} \text{the number of points in } P(q) \cap G_1^* \text{ that are} \\ \text{dominated by } q. \end{array}$$

At the end of RPE-perm, if a point $q \in N_0$ is never probed, define $P(q) = \emptyset$ and finalize its $I(q)$ to be 0.

The next lemma explains why the set $\{I(q) \mid q \in N_0\}$ separates and quantifies the influence of the noise points in N_0.

LEMMA 5. $\sum_{q \in N_0} I(q) = error(\mathcal{F}, G_1^*)$.

PROOF. Consider an arbitrary $q \in N_0$ that was probed by RPE-perm. Let p be any point in $P(q) \cap G_1^*$ that is dominated by q. By Lemma 2, $\mathcal{F}(p) = 0$ because of q.

Thus, p contributes 1 to $error(\mathcal{F}, G_1^*)$. Hence, $\sum_{q \in N_0} I(q) \le error(\mathcal{F}, G_1^*)$.

Conversely, let p be a point contributing 1 to $error(\mathcal{F}, G_1^*)$, that is, $label(p) = 1$ but $\mathcal{F}(p) = 0$. Let S be the set of label-0 points in Z that dominate p. By Lemma 2, $|S| \ge 1$. Define q to be the point in S that was probed the earliest. Because p is an ordinary point with label 1 dominated by q, q must be a noise point, i.e., $q \in N_0$. Next, we argue that p must be in $P(q)$, meaning that p contributes 1 to $I(q)$, which in turn indicates $error(\mathcal{F}, G_1^*) \le \sum_{q \in N_0} I(q)$.

On the contrary, suppose that $p \notin P(q)$. Thus, p had already disappeared when RPE-perm was about to probe q. By definition of q, this implies that RPE-perm had probed a label-1 point dominated by p; but doing so should have got q discarded, giving a contradiction. □

4.3 Proof of Lemma 3

Let us denote the points of N_0 as $q_1, q_2, \ldots, q_{k_0}$ in an arbitrary order, and introduce a random variable

$$X = \sum_{i=1}^{k_0} I(q_i).$$

We will show $\mathbf{E}[X] \le |N_0| = k_0$, which will prove Lemma 3 by way of Lemma 5. Given a subset S of P and any point $q \in P$, define:

$$D_S(q) = \{p \in S \mid \ q \text{ dominates } p\}.$$

Our proof of $\mathbf{E}[X] \le k_0$ is inductive on k_0.

4.3.1 The Base Case

Let us start with the case $k_0 = 1$, namely, $N_0 = \{q_1\}$.

LEMMA 6. *$I(q_1) > 0$ only if q_1 has a smaller rank than all the points in $D_{G_1^*}(q_1)$.*

PROOF. Suppose that $D_{G_1^*}(q_1)$ has a point p that ranks before q_1 in the permutation. We argue that RPE-perm will not probe q_1.

Suppose that RPE-perm probes q_1. Consider the moment right before the probing happens. Point p must have disappeared from P (otherwise, RPE-perm cannot have chosen to probe q since the rank of p is smaller). Could it have been discarded due to the probing of a label-0 point $p' \ne q_1$? No, because otherwise, $p \in G_1^*$ asserts that p' must also be a label-0 noise point, contradicting $k_0 = 1$. Thus, p must have been discarded due to the probing of a label-1 point that p dominates. But this should have evicted q_1 as well, also giving a contradiction. □

Hence, $I(q_1) > 0$ with a probability at most $1/(1+|D_{G_1^*}(q_1)|)$. Since $I(q_1)$ obviously cannot exceed $|D_{G_1^*}(q_1)|$, we have:

$$\mathbf{E}[I(q_1)] \le \frac{|D_{G_1^*}(q_1)|}{1 + |D_{G_1^*}(q_1)|} < 1.$$

4.3.2 The Inductive Case

Assuming $\mathbf{E}[X] \le k_0$ when $k_0 = t - 1$ for some integer $t \ge 2$, we will prove that the same holds also for $k_0 = t$.

Define $J(i)$ ($i \in [1, t]$) as the event that q_i has the largest permutation rank among $q_1, q_2, \ldots, q_t$. We will show

$$\mathbf{E}[X \mid J(i)] \quad \le \quad t \qquad (5)$$

for all i, which will give

$$\mathbf{E}[X] \quad = \quad \sum_{i=1}^{t} \mathbf{E}[X \mid J(i)] \cdot \mathbf{Pr}[J(i)] \le \sum_{i=1}^{t} t \cdot \frac{1}{t} = t$$

as is needed to complete the inductive argument.

Due to symmetry, the subsequent discussion will prove (5) only for $i = t$, and hence, will be conditioned on the event $J(t)$. Recall that RPE-perm probes points in ascending order of rank. Let us define the *watershed moment* as:

- The moment right before RPE-perm probes the *first* point with a *larger* rank than all of $q_1, q_2, \ldots, q_{t-1}$;
- End of RPE-perm, if it does not probe any point that ranks after $q_1, q_2, \ldots, q_{t-1}$.

At the watershed moment, $I(q_1), \ldots, I(q_{t-1})$ have been finalized. Set $Y = \sum_{i=1}^{t-1} I(q_i)$. Denote by P_{water} the content of P at this instant.

The inductive assumption implies that $\mathbf{E}[Y] \le t - 1$. To understand why, imagine deleting q_t from P, after which the input set P' has $t - 1$ label-0 noise points, but the same G_1^*. The permutation after removing q_t is a random permutation of P'. Thus, Y is exactly the value of $error(\mathcal{F}, G_1^*)$ on P'.

The remainder of the proof shows $\mathbf{E}[I(q_t) \mid J(t)] \le 1$. This will establish (5) because

$$\mathbf{E}[X \mid J(t)] = \mathbf{E}[Y] + \mathbf{E}[I(q_t) \mid J(t)].$$

$I(q_t) = 0$ when q_t is not in P_{water} (and hence, will not be probed). Hence, it suffices to prove

$$\mathbf{E}[I(q_t) \mid J(t), q_t \in P_{water}] \le 1.$$

Towards the purpose, we expand the left hand side over all possible sets W that P_{water} may be equal to:

$$\begin{aligned} & \mathbf{E}[I(q_t) \mid J(t), q_t \in P_{water}] \\ = \quad & \sum_{W} \mathbf{E}[I(q_t) \mid J(t), q_t \in P_{water} = W] \cdot \mathbf{Pr}[W]. \qquad (6) \end{aligned}$$

We will concentrate on proving that

$$\mathbf{E}[I(q_t) \mid J(t), q_t \in P_{water} = W] \le 1$$

regardless of W, with which (6) can be bounded from above by $\sum_W \mathbf{Pr}[W] = 1$.

Subject to the joint event "$J(t)$ and $q_t \in P_{water} = W$", the elements of W are symmetric with respect to their *relative* ordering in the permutation: any of the $|W|!$ orderings can take place with an equal probability. The analysis of $\mathbf{E}[I(q_t)]$ under that joint event is essentially the same as the base case. By the same argument as in the proof of Lemma 6, we assert that $I(q_t) > 0$ only if q_t ranks before all the points in $D_{W \cap G_1^*}(q_t)$, which happens with a probability of $1/(1 + |D_{W \cap G_1^*}(q_t)|)$. As $I(q_t)$ cannot exceed $|D_{W \cap G_1^*}(q_t)|$ under the joint event, we conclude that $\mathbf{E}[I(q_t) \mid J(t), q_t \in P_{water} = W]$ is no more than

$$\frac{|D_{W \cap G_1^*}(q_t)|}{1 + |D_{W \cap G_1^*}(q_t)|} \le 1.$$

4.4 Proof of Lemma 4

Both RPE and RPE-perm can be described as a randomized decision tree T defined as follows. Each node u of T is associated with a subset of P, denoted as $u(P)$. If u is the root, $u(P) = P$, whereas if u is a leaf, $u(P) = \emptyset$. An internal node u has $|u(P)|$ child nodes. Each directed edge (u, v) from u to a child v stores a point—denoted as $point(u, v)$—of $u(P)$. Every point of $u(P)$ is stored on one and exactly one outgoing edge of u. For each child v, the set $v(P)$ is determined as:

- If $label(p) = 0$, $v(P)$ is the set of points in $u(P)$ that are not dominated by $point(u, v)$ (recall that a point dominates itself).
- If $label(p) = 1$, $v(P)$ is the set of points in $u(P)$ that do not dominate $point(u, v)$.

Each root-to-leaf path π represents a possible probing sequence of RPE or RPE-perm. Specifically, for each node u on π, $u(P)$ represents the content of P after the algorithm probes the points stored on (the edges of) the root-to-u path. We will prove that, for every leaf z of T, RPE and RPE-perm reach z with exactly the same probability. This establishes Lemma 4 because both error and probing cost are determined by the sequence of points probed.

Let $u_1, u_2, \ldots, u_\ell$ be the nodes on the root-to-z path (u_1 is the root and $z = u_\ell$). Obviously, RPE reaches z with probability $\Pi_{i=1}^{\ell-1} \frac{1}{|u_i(P)|}$. It remains to show that this is also true for RPE-perm.

The execution of RPE-perm is a function of the permutation of P—denoted as P_{perm}—obtained at Step 1. For each node u of T, denote by $S(u)$ the set of all possible P_{perm} that will bring the execution to u. When u is the root, $S(u)$ is the set of all $n!$ permutations.

LEMMA 7. *For $i \in [2, \ell]$, $S(u_i)$ is the set of permutations $\pi \in S(u_{i-1})$ such that $point(u_{i-1}, u_i)$ has the smallest rank in π among all the points in $u_{i-1}(P)$.*

PROOF. We prove the claim by induction. It holds for $i = 2$ because RPE-perm descends from u_1 (the root) to u_2 only when $point(u_1, u_2)$ is the first point of P_{perm}. Inductively, assume that the claim is true for $i = j - 1$. As mentioned before, $u_{j-1}(P)$ is the content of P after RPE-perm probes the points stored on the root-to-u_{j-1} path. Hence, the algorithm branches to u_j only if $point(u_{j-1}, u_j)$ is the next to pick in P_{perm} among the points in $u_{j-1}(P)$. So the claim holds also for $i = j$. □

The lemma indicates that $|S(u_i)| = |S(u_{i-1})|/|u_{i-1}(P)|$. Hence, $|S(u_\ell)| = |S(u_1)| \cdot \Pi_{i=1}^{\ell-1} \frac{1}{|u_i(P)|}$. The probability that RPE-perm reaches u_ℓ equals $|S(u_\ell)|/n!$ which is simply $\Pi_{i=1}^{\ell-1} \frac{1}{|u_i(P)|}$. This concludes the proof of Lemma 4.

5. REFERENCES

[1] A. Arasu, M. Götz, and R. Kaushik. On active learning of record matching packages. In *SIGMOD*, pages 783–794, 2010.

[2] M. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *JCSS*, 75(1):78–89, 2009.

[3] K. Bellare, S. Iyengar, A. G. Parameswaran, and V. Rastogi. Active sampling for entity matching with guarantees. *TKDD*, 7(3):12:1–12:24, 2013.

[4] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *ICML*, pages 49–56, 2009.

[5] G. D. Bianco, R. Galante, M. A. Goncalves, S. D. Canuto, and C. A. Heuser. A practical and effective sampling selection strategy for large scale deduplication. *TKDE*, 27(9):2305–2319, 2015.

[6] P. Christen, D. Vatsalan, and Q. Wang. Efficient entity resolution with adaptive and interactive training data selection. In *ICDM*, pages 727–732, 2015.

[7] X. Chu, I. F. Ilyas, and P. Koutris. Distributed data deduplication. *PVLDB*, 9(11):864–875, 2016.

[8] S. Dasgupta. Coarse sample complexity bounds for active learning. In *NIPS*, pages 235–242, 2005.

[9] S. Dasgupta, D. J. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *NIPS*, pages 353–360, 2007.

[10] S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. *Journal of Machine Learning Research (JMLR)*, 10:281–299, 2009.

[11] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 3rd edition, 2008.

[12] V. Efthymiou, G. Papadakis, G. Papastefanatos, K. Stefanidis, and T. Palpanas. Parallel meta-blocking for scaling entity resolution over big heterogeneous data. *Information Systems*, 65:137–157, 2017.

[13] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[14] C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. W. Shavlik, and X. Zhu. Corleone: hands-off crowdsourcing for entity matching. In *SIGMOD*, pages 601–612, 2014.

[15] S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML*, pages 353–360, 2007.

[16] S. Hanneke. Theory of disagreement-based active learning. *Foundations and Trends in Machine Learning*, 7(2-3):131–309, 2014.

[17] M. Kaariainen. Active learning in the non-realizable case. In *Proceedings of International Conference on Algorithmic Learning Theory (ALT)*, pages 63–77, 2006.

[18] L. Kolb, A. Thor, and E. Rahm. Load balancing for mapreduce-based entity resolution. In *ICDE*, pages 618–629, 2012.

[19] P. Konda, S. Das, P. Suganthan, A. Doan, A. Ardalan, J. R. Ballard, H. Li, F. Panahi, H. Zhang, J. F. Naughton, S. Prasad, G. Krishnan, R. Deep, and V. Raghavendra. Magellan: Toward building entity matching management systems. *PVLDB*, 9(12):1197–1208, 2016.

[20] H. Kopcke and E. Rahm. Frameworks for entity matching: A comparison. *DKE*, 69(2):197–210, 2010.

[21] H. Kopcke, A. Thor, and E. Rahm. Evaluation of entity resolution approaches on real-world match problems. *PVLDB*, 3(1):484–493, 2010.

[22] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *SIGKDD*, pages 269–278, 2002.

[23] B. Settles. Active learning literature survey. *Technical Report, University of Wisconsin-Madison*, 2010.

[24] Y. Tao. Entity matching with active monotone classification. In *PODS*, pages 49–62, 2018.

[25] A. Thor and E. Rahm. MOMA - A mapping-based object matching system. In *CIDR*, pages 247–258, 2007.

[26] L. Wang. Smoothness, disagreement coefficient, and the label complexity of agnostic active learning. *Journal of Machine Learning Research*, 12:2269–2292, 2011.