PoWareMatch: **Matching-as-a-Process and the Case of a Human Matcher – Technical Report**

Roee Shraga, Avigdor Gal
Technion – Israel Institute of Technology, Haifa, Israel
{shraga89@campus.,avigal@}technion.ac.il

ABSTRACT

Schema matching is a core task of any data integration process. Being investigated in the fields of databases, AI, Semantic Web and data mining for many years, the main challenge remains the ability to generate quality matches among data concepts (*e.g.,* database attributes). In this work, we analyze a novel angle on the behavior of humans as matchers, studying match creation as a process. We analyze the dynamics of common evaluation measures (precision, recall, and f-measure), and using a new concept of unbiased matching, design PoWareMatch that makes use of a deep learning mechanism to calibrate human matching, which is then combined with algorithmic matching to generate better match results. We provide an empirical evidence that PoWareMatch predicts well the benefit of extending the match with an additional correspondence and generates high quality matches. We also show that, tested on multiple common benchmarks, the proposed solution performs better than state-of-the-art matching algorithms.

PVLDB Reference Format:
Roee Shraga, Avigdor Gal. PoWareMatch: Matching-as-a-Process and the Case of a Human Matcher – Technical Report. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

1 INTRODUCTION

Schema matching is a core task of data integration for structured and semi-structured data. Matching revolves around providing correspondences between concepts describing the meaning of data in various heterogeneous, distributed data sources, such as SQL and XML schemata, entity-relationship diagrams, ontology descriptions, interface definitions, *etc.*. The need for schema matching arises in a variety of domains including linking datasets and entities for data discovery [21, 26, 46, 57, 59], finding related tables in data lakes [64], data enrichment [60, 61], aligning ontologies and relational databases for the Semantic Web [23, 48, 58], and document format merging (*e.g.*, orders and invoices in e-commerce) [50]. As an example, a shopping comparison app that supports queries such as “the cheapest computer among retailers” or “the best rate for a flight to Denmark in August” requires integrating and matching several data sources of product orders and airfare forms.

Schema matching research originated in the database community [50] and has been a focus for other disciplines as well, from artificial intelligence [30], to semantic web [23], to data mining [28, 32]. Schema matching research has been going on for more than 30 years now, focusing on designing high quality matchers, automatic tools for identifying correspondences among database attributes. Initial heuristic attempts (*e.g.*, COMA [18] and Similarity Flooding [39]) were followed by theoretical grounding (*e.g.*, see [12, 19, 27]).

Human schema and ontology matching, the holy grail of matching, requires domain expertise [20, 36]. Zhang *et al.* stated that users that match schemata are typically non experts, and may not even know what is a schema [63]. Others, *e.g.*, [53, 55], have observed the diversity among human inputs. Recently, human matching was challenged by the information explosion (a.k.a Big Data) that provided many novel sources for data and with it the need to efficiently and effectively integrate them. So far, challenges raised by human matching were answered by pulling further on human resources, using crowdsourcing (*e.g.*, [25, 44, 54, 62, 63]) and pay-as-you-go frameworks (*e.g.*, [38, 43, 49]). However, recent research have challenged both traditional and new methods for human-in-the-loop matching, showing that humans have cognitive biases decreasing their ability to perform matching tasks effectively [8]. For example, the study shows that over time, human matchers are willing to determine that an element pair matches despite their low confidence in the match, possibly leading to poor performance.

Faced with the challenges raised by human matching, we offer a novel angle on the behavior of humans as matchers, analyzing *matching as a process*. We now motivate our proposed analysis using an example and then outline the paper’s contribution.

Motivating Example: When it comes to humans performing a matching task, decisions regarding correspondences between data sources are made sequentially. To illustrate, consider Figure 1, presenting two simplified purchase order schemata adopted from [18]. PO_1 has four attributes (foreign keys are ignored for simplicity): purchase order’s number (poCode), timestamp (poDay and poTime) and shipment city (city). PO_2 has three attributes: order issuing date (orderDate), order number (orderNumber), and shipment city (city). A human matching sequence is given by the orderly annotated double-arrow edges. For example, a matching decision that poDay in PO_1 corresponds to orderNumber in PO_2 is the second.

The traditional view on human matching accepts human decisions as a ground truth (possibly subject to validation by additional human matchers) and thus the outcome match is composed of all the correspondences selected (or validated) by the human matcher. Figure 2 illustrates this view using the decision making process of Figure 1. The x-axis represents the ordering according to which correspondences were selected. The dashed line at the top illustrates

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

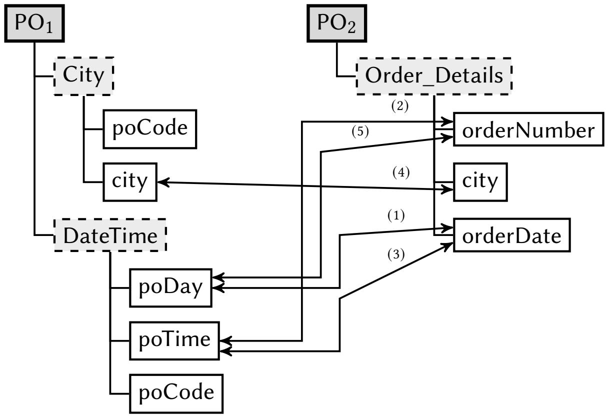


Figure 1: Matching-as-a-Process example over two schemata. Decision ordering is annotated on the double-arrow edges.

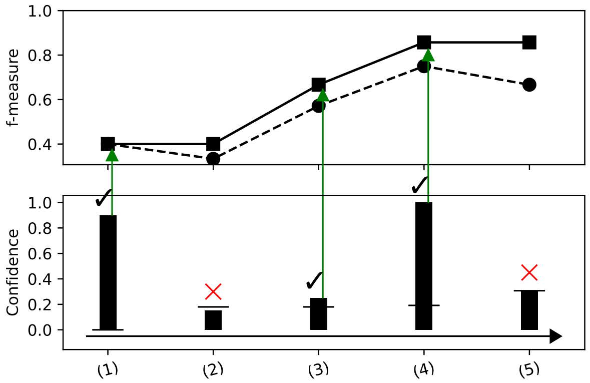


Figure 2: Matching as a Process performance. The x-axis represents the ordered decision of Figure 1. The bars at the bottom represent the confidence associated with each decision and the horizontal lines represent a dynamic threshold (see Section 4.1 for details) to determine acceptance/rejection of human matching decisions. At the top, the circle markers represent the performance of the traditional approach, unconditionally accepting human decisions, and the square markers represent a process-aware inference over the decisions based on the thresholds at the bottom.

the changes to the f-measure (see Section 2.1 for f-measure's definition) as more decisions are added, starting with an f-measure of 0.4, dropping slightly and then rising to a maximum of 0.75 before dropping again as a result of the final human matcher decision.

In this work we offer an alternative approach that analyzes human decision making by utilizing the sequential nature of the process and confidence values humans share to monitor performance and modify decisions accordingly. In particular, we set a dynamic threshold (marked as horizontal lines at the bottom of Figure 2), based on previous decisions. Comparing the threshold to decision confidence (marked as bars in the figure), the algorithm determines whether a human decision is included in the match (marked using a ✓sign and an arrow to indicate inclusion) or not (marked as a red X). A process-aware approach, for example, accepts the third decision that poTime in PO$_1$ corresponds with orderDate in PO$_2$ made with a confidence of 0.25 while rejecting the fifth decision that poDay in PO$_1$ corresponds with orderNumber in PO$_2$, despite the higher confidence of 0.3. The f-measure of this approach, as illustrated by the solid line at the top of the figure, demonstrates a monotonic non-decreasing behavior with a final high value of 0.86.

More on the method, assumptions and how the acceptance threshold is set is given in Section 4.1. Section 4.2 further introduces a methodology to calibrate matching decisions, and the respective confidence, on-the-go, using a deep learning technique.

Contribution: We characterize the dynamics of matching as a process by defining a *monotonic evaluation measure* and its probabilistic derivative. We show conditions under which precision, recall and f-measure are monotonic and accordingly identify correspondences that their addition improves on a match evaluation. Using a novel concept of *unbiased matching*, we design a step-wise matching algorithm that takes into account human confidence when constructing a match in a piecemeal fashion. To overcome human biases, we propose PoWareMatch (**Process** a**Ware Match**er) that uses deep learning to calibrate human matching, which is then combined with algorithmic matching to provide better match results. We provide an empirical evidence that PoWareMatch generates high quality matches. The paper offers the following four specific contributions:

(1) A formal framework for evaluating matching as a process using the well known evaluation measures of precision, recall, and f-measure (Section 3).
(2) A matching algorithm that uses confidence to generate a process-aware match (Section 4.1).
(3) PoWareMatch (Section 4.2), a matching algorithm that uses a deep learning model to calibrate human matching decisions (Section 4.3) and algorithmic matchers to complement human matching (Section 4.4).
(4) An empirical evaluation showing the superiority of our proposed solution over state-of-the-art in schema matching using known benchmarks (Section 5).

Section 2 provides a matching model and discusses algorithmic and human matching. The paper is concluded with related work (Section 6), concluding remarks and future work (Section 7).

2 MODEL

In this section we present the foundations for this work, starting with a matching model, and introducing the matching problem (Section 2.1). We then position algorithmic (Section 2.2) and human matching (Section 2.3) within this setting.

2.1 Schema Matching Model

Let S, S' be two schemata with attributes $\{a_1, a_2, \dots, a_n\}$ and $\{b_1, b_2, \dots, b_m\}$, respectively. A matching model matches S and S' by aligning their attributes using *matchers* that utilize matching cues such as attribute names, instance data, schema structure, *etc.* (see surveys, *e.g.*, [13] and books, *e.g.*, [27]).

A matcher's output is conceptualized as a matching matrix $M(S, S')$ (or simply M), as follows.

Definition 2.1. *$M(S, S')$ is a* matching matrix, *having entry M_{ij} (typically a real number in $[0,1]$) represent a measure of fit (possibly a similarity or a confidence measure) between $a_i \in S$ and $b_j \in S'$. M is* binary *if for all $1 \leq i \leq n$ and $1 \leq j \leq m$, $M_{ij} \in \{0,1\}$. A* match, *denoted σ, between S and S' is a subset of M's entries. Each entry in a match is denoted a* correspondence. *$\Sigma = \mathcal{P}(S \times S')$ is the set of all possible matches.*

Let M^* be a *reference matrix*. M^* is a binary matrix, such that $M^*_{ij} = 1$ whenever $a_i \in S$ and $b_j \in S'$ correspond and $M^*_{ij} = 0$ otherwise. A *reference match*, denoted σ^*, is given by $\sigma^* = \{M^*_{ij} | M^*_{ij} = 1\}$. $G_{\sigma^*} : \Sigma \rightarrow [0,1]$ is an evaluation measure, assigning scores to matches according to their ability to identify correspondences in the reference match. Whenever the reference match is clear from the context, we shall refer to G_{σ^*} simply as G. We define the precision (P) and recall (R) evaluation measures [11], as follows:

$$P(\sigma) = \frac{|\ \sigma \cap \sigma^*\ |}{|\ \sigma\ |}, R(\sigma) = \frac{|\ \sigma \cap \sigma^*\ |}{|\ \sigma^*\ |} \tag{1}$$

recalling that σ is a subset of M's entries. The f-measure (F_1 score), $F(\sigma)$, is calculated as the harmonic mean of $P(\sigma)$ and $R(\sigma)$.

The schema matching problem can be expressed as follows:

Problem 1 (Matching). *Let S, S' be two schemata and G_{σ^*} be an evaluation measure wrt a reference match σ^*. We seek a match $\sigma \in \Sigma$, aligning attributes of S and S', which maximizes G_{σ^*}.*

2.2 Algorithmic Schema Matching

Matching is often a stepped procedure applying algorithms, rules, and constraints. Algorithmic matchers can be classified into those that are applied directly to the problem (first-line matchers – 1LMs) and those that are applied to the outcome of other matchers (second-line matchers – 2LMs). 1LMs receive (typically two) schemata and return a matching matrix, in which each entry M_{ij} captures the similarity between attributes a_i and b_j. 2LMs receive (one or more) matching matrices and return a matching matrix using some function $f(M)$ [27]. Among the 2LMs, we term decision makers those that return a binary matrix as an output, from which a match σ is derived, by maximizing $f(M)$, as a solution to Problem 1.

To illustrate the algorithmic matchers in the literature, consider three 1LMs, namely Term, WordNet, and Token Path and three 2LMs, namely Dominants, Threshold(ν), and Max-Delta(δ), as follows. Term [27] compares attribute names to identify syntactically similar attributes (*e.g.*, using edit distance and soundex). WordNet uses abbreviation expansion and tokenization methods to generate a set of related words for matching attribute names [29, 51, 52]. Token Path [47] integrates node-wise similarity with structural information by comparing the syntactic similarity of full paths from root to a node. Dominants [27] selects correspondences that dominate all matching matrix entries in their row and column. Threshold(ν) and Max-Delta(δ) are selection rules, prevalent in many matching systems [18]. Threshold(ν) selects those entries (i, j) having $M_{ij} \geq \nu$. Max-Delta(δ) selects those entries that satisfy: $M_{ij} + \delta \geq \max_i$, where $\max_i$ denotes the maximum match value in the i'th row.

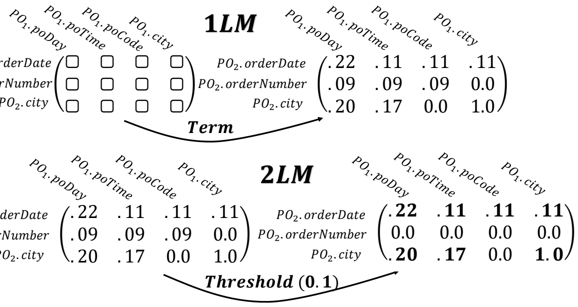


Figure 3: Algorithmic matching example.

Example 1. *Figure 3 provides an example of algorithmic matching over the two purchase order schemata from Figure 1. The top right (and bottom left) matching matrix is the outcome of* Term *and the bottom right is the outcome of* Threshold(0.1). *The projected match is $\sigma_{alg} = \{M_{11}, M_{12}, M_{13}, M_{14}, M_{31}, M_{32}, M_{34}\}$.*[1] *The reference match for this example is given by $\{M_{11}, M_{12}, M_{23}, M_{34}\}$ and accordingly $P(\sigma_{alg}) = 0.43$, $R(\sigma_{alg}) = 0.75$, and $F(\sigma_{alg}) = 0.55$.*

2.3 Human Schema Matching

In this work, we examine human schema matching as a decision making process. Different from a binary crowdsourcing setting (*e.g.*, [62]), in which the matching task is usually reducted into a set of (binary) judgments regarding specific correspondences (see Section 6), we look at human matchers who perform a matching task in its entirety, as illustrated in Figure 1.

Human schema matching is a complex decision making process, which involves sequential interrelated decisions. Each attribute in one schema is examined to decide whether and which attributes from the other schema correspond. Humans either validate an algorithmic result or locate a candidate attribute unassisted. In doing so, human matchers may choose to rely upon superficial information such as string similarity of attribute names or explore additional information such as data-types, instances, and position within the schema tree. The decision whether to explore additional information relies upon self-monitoring of confidence.

Human schema matching has been recently analyzed using metacognitive psychology [8], a discipline that investigates factors impacting humans when performing knowledge intensive tasks [10]. The metacognitive approach [14], traditionally applied for learning and answering knowledge questions, highlights the role of *subjective confidence* in regulating efforts while performing tasks. Metacognitive research shows that subjective judgments (*e.g.*, confidence) regulate the cognitive effort invested in each decision (*e.g.*, identifying a correspondence) [9, 40]. In what follows, we model human matching as a sequence of decisions regarding element pairs, each assigned with a confidence level. We can directly query human matchers' (subjective) confidence level regarding a correspondence, reflecting the ongoing monitoring and final subjective assessment of chances of success [9, 14].

The dynamics of human matching decision making process is modeled using a *decision history H*, as follows.

Definition 2.2. *Given two schemata S, S' with attributes $\{a_1, a_2, \ldots, a_n\}$ and $\{b_1, b_2, \ldots, b_m\}$, respectively, a history H =*

[1] Recall the M_{ij} represents a correspondence between the i'th element in S and the j'th element in S', *e.g.*, M_{11} means that PO$_1$.poDay and PO$_2$.orderDate correspond.

$\langle h_t \rangle_{t=1}^{T}$ is a sequence of triplets of the form $\langle e, c, t \rangle$, where $e = (a_i, b_j)$ such that $a_i \in S$, $b_j \in S'$, $c \in [0,1]$ is a confidence value assigned to the correspondence of a_i and b_j, and $t \in \mathbb{R}$ is a timestamp, recording the time the decision was taken.

Each decision $h_t \in H$ records a matching decision confidence ($h_t.c$) concerning an element pair ($h_t.e$) at time t ($h_t.t$). Timestamps induce a total order over H's elements.

A matching matrix, which may serve as a solution of the human matcher to Problem 1, can be created from a matching history by assigning the latest confidence to the respective matrix entry. Given an element pair $e = (a_i, b_j)$, we denote by h^e_{max} the latest decision making in H that refers to e and compute a matrix entry as follows:

$$M_{ij} = \begin{cases} h^e_{max}.c & \text{if } \exists h_t \in H | h_t.e = (a_i, b_j) \\ \varnothing & \text{otherwise} \end{cases} \tag{2}$$

where $\varnothing$ denotes an empty entry, *i.e.*, $M_{ij} = \varnothing$ means that the matcher did not assign a confidence value for M_{ij}. Whenever clear from the context, we refer to a confidence value ($h_t.c$) assigned to an element pair (a_i, b_j), simply as M_{ij}.

H

	e	c	t
h_1	$PO_1.poDayP$, $O_2.orderDate$	.9	5.0
h_2	$PO_1.poTime$, $PO_2.orderNumber$	.15	15.0
h_3	$PO_1.poTime$, $PO_2.orderDate$	.25	21.0
h_4	$PO_1.city$, $PO_2.city$	1.0	24.0
h_5	$PO_1.poDay$, $PO_2.orderNumber$	.3	35.0

$$M \quad \begin{array}{r|cccc} & PO_1.poDay & PO_1.poTime & PO_1.poCode & PO_1.city \\ \hline PO_2.orderDate & \mathbf{.9} & \mathbf{.25} & \emptyset & \emptyset \\ PO_2.orderNumber & \mathbf{.3} & \mathbf{.15} & \emptyset & \emptyset \\ PO_2.city & \emptyset & \emptyset & \emptyset & \mathbf{1.0} \end{array}$$

Figure 4: Human matching example.

Example 1 (cont.). *Figure 4 (left) provides the decision history that corresponds to the matching process of Figure 1 and the respective matching matrix (applying Eq. 2) is given on the right. The projected match is $\sigma_{hum} = \{M_{11}, M_{22}, M_{12}, M_{34}, M_{21}\}$. Recalling the reference match, $\{M_{11}, M_{12}, M_{23}, M_{34}\}$, the projected match obtains $P(\sigma_{hum}) = 0.6$, $R(\sigma_{hum}) = 0.75$, and $F(\sigma_{hum}) = 0.67$.*

In this work we seek a solution to Problem 1 that takes into account both an evaluation measure G and a decision history H. Therefore, we next analyze the well-known measures, namely precision, recall, and f-measure in the context of matching-as-a-process.

3 (HUMAN) MATCHING-AS-A-PROCESS

Equipped with a matching matrix (Definition 2.1) and a decision history (Definition 2.2), we next analyze the properties of matching evaluation measures (precision, recall, and f-measure, see Eq. 1) in the setting of matching-as-a-process. We start by analyzing evaluation measure monotonicity (Section 3.1), showing that recall is always monotonic, while precision and f-measure are monotonic only under strict conditions. Then, we show a probabilistic analysis, identifying which correspondence should be added to an existing partial match (Section 3.2). Finally, Section 3.3 ties the analysis with human matching and discusses the idea of unbiased matching.

3.1 Monotonic Evaluation

Given two matches (Definition 2.1) σ and σ' that are a result of some sequential decision making such that $\sigma \subseteq \sigma'$, we define their interrelationship using their monotonic behavior with respect to an evaluation measure G (P, R, or F, see Eq. 1). For the remainder of the section, we denote by $\Sigma^{\subseteq}$ the set of all match pairs in $\Sigma \times \Sigma$ such that the first match is a subset of the second:

$$\Sigma^{\subseteq} = \{(\sigma, \sigma') \in \Sigma \times \Sigma : \sigma \subseteq \sigma'\}$$

We use $\Delta_{\sigma,\sigma'} = \sigma' \setminus \sigma$ (Δ, when clear from the context) to denote the set of correspondences that were added to σ to generate σ'.

Definition 3.1 (Monotonic Evaluation Measure). *Let G be an evaluation measure and $\Sigma^2 \subseteq \Sigma^{\subseteq}$ a set of match pairs in $\Sigma^{\subseteq}$. G is a* monotonically increasing evaluation measure (MIEM) over Σ^2 *if for all match pairs $(\sigma, \sigma') \in \Sigma^2$, $G(\sigma) \leq G(\sigma')$.*

According to Definition 3.1, an evaluation measure G is monotonically increasing (MIEM) if by adding correspondences to a match, we do not reduce its value. We next show that recall is a MIEM over all match pairs in $\Sigma^{\subseteq}$.[*]

Lemma 3.1. *Recall (R) is a MIEM over $\Sigma^{\subseteq}$.*

Proof 3.1. *Let $(\sigma, \sigma') \in \Sigma^{\subseteq}$ be a match pair in $\Sigma^{\subseteq}$. Using Eq. 1, one can compute recall of σ and σ', as follows.*

$$R(\sigma) = \frac{|\ \sigma \cap \sigma^*\ |}{|\ \sigma^*\ |}, R(\sigma') = \frac{|\ \sigma' \cap \sigma^*\ |}{|\ \sigma^*\ |}$$

$\sigma \subseteq \sigma'$ and thus $(\sigma \cap \sigma^*) \subseteq (\sigma' \cap \sigma^*)$ and $|\sigma \cap \sigma^*| \leq |\sigma' \cap \sigma^*|$. Noting that the denominator is not affected by the addition, we obtain:

$$R(\sigma) = \frac{|\ \sigma \cap \sigma^*\ |}{|\ \sigma^*\ |} \leq \frac{|\ \sigma' \cap \sigma^*\ |}{|\ \sigma^*\ |} = R(\sigma')$$

and thus $R(\sigma) \leq R(\sigma')$.

Unlike recall, precision and f-measure are not monotonic over the full set of pairs in $\Sigma^{\subseteq}$. We next define the conditions under which monotonicity can be guaranteed for both measures.

Lemma 3.2. *For $(\sigma, \sigma') \in \Sigma^{\subseteq}$:*

- *$P(\sigma) \leq P(\sigma')$ iff $P(\sigma) \leq P(\Delta)$*
- *$F(\sigma) \leq F(\sigma')$ iff $0.5 \cdot F(\sigma) \leq P(\Delta)$*

Proof 3.2. *Let $(\sigma, \sigma') \in \Sigma^{\subseteq}$ be a match pair in $\Sigma^{\subseteq}$. We shall begin with two extreme cases in which the denominator of a precision calculation is zero. First, in case $\sigma = \emptyset$, $P(\sigma)$ is undefined. Accordingly, for this work, we shall define $P(\sigma) = P(\Delta)$,* i.e., *the prior precision value does not change. Second, in the case $\sigma' = \sigma$, we obtain that $\sigma' \setminus \sigma = \emptyset$ resulting in an undefined $P(\Delta)$. Similarly, we define $P(\Delta) = P(\sigma)$,* i.e., *expanding a match with nothing does not "harm" its precision. In both cases we obtain $P(\sigma) = P(\Delta) = P(\sigma')$ and the first statement of the lemma holds.*

Next, we refer to the general case where $\sigma \neq \emptyset$ and $\Delta \neq \emptyset$ and note that $|\sigma'| = |\sigma \cup \Delta|$. Assuming $\sigma \neq \sigma'$ and recalling that $\sigma \subseteq \sigma'$ ($(\sigma, \sigma') \in \Sigma^{\subseteq}$), we obtain that $\sigma \cap \Delta = \emptyset$ and

$$|\sigma'| = |\sigma \cup \Delta| = |\sigma| + |\Delta| \tag{3}$$

[*] Due to space considerations, we refrain from presenting some of the proofs. The interested reader is referred to a technical report [6].

Similarly, we obtain

$$|\sigma' \cap \sigma^*| = |(\sigma \cap \sigma^*) \cup (\Delta \cap \sigma^*)| \tag{4}$$

Recalling that $\sigma \subseteq \sigma'$, we have $(\sigma \cap \sigma^*) \cap (\Delta \cap \sigma^*) = \emptyset$ and

$$|\sigma' \cap \sigma^*| = |\sigma \cap \sigma^*| + |\Delta \cap \sigma^*| \tag{5}$$

- $\mathbf{P}(\sigma) \leq \mathbf{P}(\sigma')$ ***iff*** $\mathbf{P}(\sigma) \leq \mathbf{P}(\Delta)$*:*
  Using Eq. 1, the precision values of σ, σ', and Δ are computed as follows:
  $$P(\sigma) = \frac{|\sigma \cap \sigma^*|}{|\sigma|}, P(\sigma') = \frac{|\sigma' \cap \sigma^*|}{|\sigma'|}, P(\Delta) = \frac{|\Delta \cap \sigma^*|}{|\Delta|} \tag{6}$$
  $\Rightarrow$: Let $P(\sigma) \leq P(\sigma')$. Using Eq. 6, we obtain that
  $$\frac{|\sigma \cap \sigma^*|}{|\sigma|} \leq \frac{|\sigma' \cap \sigma^*|}{|\sigma'|}$$
  and accordingly (Eq. 5 and Eq. 3),
  $$\frac{|\sigma \cap \sigma^*|}{|\sigma|} \leq \frac{|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|}{|\sigma| + |\Delta|}$$
  Then, we obtain
  $$|\sigma \cap \sigma^*|(|\sigma| + |\Delta|) \leq |\sigma|(|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|)$$
  and following,
  $$|\sigma \cap \sigma^*| \cdot |\sigma| + |\sigma \cap \sigma^*| \cdot |\Delta| \leq |\sigma| \cdot |\sigma \cap \sigma^*| + |\sigma| \cdot |\Delta \cap \sigma^*|$$
  Finally, we get
  $$|\sigma \cap \sigma^*| \cdot |\Delta| \leq |\sigma| \cdot |\Delta \cap \sigma^*|$$
  and conclude
  $$P(\sigma) = \frac{|\sigma \cap \sigma^*|}{|\sigma|} \leq \frac{|\Delta \cap \sigma^*|}{|\Delta|} = P(\Delta)$$
  $\Leftarrow$: Let $P(\sigma) \leq P(\Delta)$. Using Eq. 6, we obtain that
  $$\frac{|\sigma \cap \sigma^*|}{|\sigma|} \leq \frac{|\Delta \cap \sigma^*|}{|\Delta|}$$
  and following
  $$|\sigma \cap \sigma^*| \cdot |\Delta| \leq |\sigma| \cdot |\Delta \cap \sigma^*|$$
  By adding $|\sigma \cap \sigma^*| \cdot |\sigma|$ on both sides we get
  $$|\sigma \cap \sigma^*| \cdot |\Delta| + |\sigma \cap \sigma^*| \cdot |\sigma| \leq |\sigma| \cdot |\Delta \cap \sigma^*| + |\sigma \cap \sigma^*| \cdot |\sigma|$$
  After rewriting, we obtain
  $$|\sigma \cap \sigma^*| \cdot (|\Delta| + |\sigma|) \leq |\sigma| \cdot (|\Delta \cap \sigma^*| + |\sigma \cap \sigma^*|)$$
  and in what follows (Eq. 5 and Eq. 3):
  $$P(\sigma) = \frac{|\sigma \cap \sigma^*|}{|\sigma|} \leq \frac{|\sigma' \cap \sigma^*|}{|\sigma'|} = P(\sigma')$$
- $\mathbf{F}(\sigma) \leq \mathbf{F}(\sigma')$ ***iff*** $\mathbf{0.5F}(\sigma) \leq \mathbf{P}(\Delta)$*:*
  The F1 measure value of σ is given by
  $$F(\sigma) = 2 \cdot \frac{P(\sigma) \cdot R(\sigma)}{P(\sigma) + R(\sigma)}$$
  Then, using Eq. 1, we have
  $$F(\sigma) = 2 \cdot \frac{\frac{|\sigma \cap \sigma^*|}{|\sigma|} \cdot \frac{|\sigma \cap \sigma^*|}{|\sigma^*|}}{\frac{|\sigma \cap \sigma^*|}{|\sigma|} + \frac{|\sigma \cap \sigma^*|}{|\sigma^*|}}$$
  and after rewriting we obtain
  $$F(\sigma) = \frac{2 \cdot |\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \tag{7}$$
  Similarly, we can compute
  $$F(\sigma') = \frac{2 \cdot |\sigma' \cap \sigma^*|}{|\sigma'| + |\sigma^*|}$$
  Using Eq. 5 and Eq. 3 , we further obtain
  $$F(\sigma') = \frac{2 \cdot (|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|)}{(|\sigma| + |\Delta|) + |\sigma^*|} \tag{8}$$
  $\Rightarrow$: Let $F(\sigma) \leq F(\sigma')$. Using Eq. 7 and Eq. 8, we obtain
  $$\frac{2 \cdot |\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \leq \frac{2 \cdot (|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|)}{(|\sigma| + |\Delta|) + |\sigma^*|}$$
  after rewriting we get
  $$|\sigma \cap \sigma^*| \cdot (|\sigma| + |\sigma^*| + |\Delta|) \leq (|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|) \cdot (|\sigma| + |\sigma^*|)$$
  and following
  $$|\sigma \cap \sigma^*| \cdot |\Delta| \leq |\Delta \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|) \rightarrow \frac{|\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \leq \frac{|\Delta \cap \sigma^*|}{|\Delta|}$$
  Recalling that $P(\Delta) = \frac{|\Delta \cap \sigma^*|}{|\Delta|}$ (Eq. 6) we conclude
  $$0.5F(\sigma) = \frac{|\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \leq \frac{|\Delta \cap \sigma^*|}{|\Delta|} = P(\Delta)$$
  $\Leftarrow$: Let $0.5F(\sigma) \leq P(\Delta)$. Using Eq. 7 and Eq. 6, we obtain
  $$0.5\frac{2|\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \leq \frac{|\Delta \cap \sigma^*|}{|\Delta|}$$
  and accordingly,
  $$|\sigma \cap \sigma^*| \cdot |\Delta| \leq |\Delta \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|)$$
  By adding $|\sigma \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|)$ on both sides we get
  $$|\sigma \cap \sigma^*| \cdot |\Delta| + |\sigma \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|) \leq |\Delta \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|) + |\sigma \cap \sigma^*| \cdot (|\sigma| + |\sigma^*|)$$
  After rewriting, we obtain
  $$|\sigma \cap \sigma^*| \cdot (|\sigma| + |\sigma^*| + |\Delta|) \leq (|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|) \cdot (|\sigma| + |\sigma^*|)$$
  Then, dividing both sides by $0.5 \cdot (|\sigma| + |\sigma^*| + |\Delta|) \cdot (|\sigma| + |\sigma^*|)$, we get
  $$F(\sigma) = \frac{2 \cdot |\sigma \cap \sigma^*|}{|\sigma| + |\sigma^*|} \leq \frac{2 \cdot (|\sigma \cap \sigma^*| + |\Delta \cap \sigma^*|)}{(|\sigma| + |\Delta|) + |\sigma^*|} = F(\sigma')$$

which concludes the proof.

We define two subsets, $\Sigma^P = \{(\sigma, \sigma') \in \Sigma^{\subseteq} : P(\sigma) \leq P(\Delta)\}$ and $\Sigma^F = \{(\sigma, \sigma') \in \Sigma^{\subseteq} : 0.5 \cdot F(\sigma) \leq P(\Delta)\}$ and use them in the following theorem to summarize the main dynamic properties of the evaluation measures of precision, recall, and f-measure.

Theorem 3.1. *Recall (R) is a MIEM over $\Sigma^{\subseteq}$, Precision (P) is a MIEM over Σ^P, and f-measure (F) is a MIEM over Σ^F.*

Proof 3.3. *The first part of the theorem follows directly from Lemma 3.1. For the reminder of the proof we rely on Lemma 3.2.*

Let $(\sigma, \sigma') \in \Sigma^P$ be a match pair in Σ^P. By definition, since $(\sigma, \sigma') \in \Sigma^P$, then $P(\sigma) \leq P(\Delta)$ and by Lemma 3.2, we can conclude that $P(\sigma) \leq P(\sigma')$.

Similarly, let $(\sigma, \sigma') \in \Sigma^F$ be a match pair in Σ^F. By definition, since $(\sigma, \sigma') \in \Sigma^F$, then $0.5 \cdot F(\sigma) \leq P(\Delta)$ and by Lemma 3.2, we can infer that $F(\sigma) \leq F(\sigma')$, which concludes the proof.

3.2 Local Match Annealing

The analysis of Section 3.1 lays the groundwork for a principled matching process that continuously improves on the evaluation measure of choice. The conditions set forward use knowledge of, first, the evaluation outcome of the match performed thus far ($G(\sigma)$), and second, the evaluation score of the additional correspondences ($G(\Delta)$). While such knowledge can be extremely useful, it is rarely available during the matching process. Therefore, we next provide a relaxed setting, where $G(\sigma)$ and $G(\Delta)$ are probabilistically known (Section 3.3 provides an approximation the using human confidence). For simplicity, we restrict our analysis to matching processes where a single correspondence is added at a time. We denote by $\Sigma^{\subseteq_1} = \{(\sigma, \sigma') \in \Sigma^{\subseteq} : |\sigma'| - |\sigma| = 1\}$ the set of all match pairs (σ, σ') in $\Sigma^{\subseteq}$ where σ' is generated by adding a single correspondence to σ. The discussion below can be extended (beyond the scope of this work) to adding multiple correspondences at a time.

We start with a characterization of correspondences whose addition to a match improves on the match evaluation. Recall that Δ represents the marginal set of correspondences that were added to the match. In what follows, Δ represents a single correspondence ($|\Delta| = 1$), which may be the result of multiple match pairs $(\sigma, \sigma') \in \Sigma^{\subseteq_1}$ such that $\Delta = \sigma' \setminus \sigma$.

Definition 3.2 (Local Match Annealing). *Let G be an evaluation measure and Δ be a singleton correspondence set ($|\Delta| = 1$). Δ is a* local annealer with respect to G over $\Sigma^2 \subseteq \Sigma^{\subseteq_1}$ *if for every $(\sigma, \sigma') \in \Sigma^2$ s.t. $\Delta = \Delta_{(\sigma,\sigma')}$: $G(\sigma) \leq G(\sigma')$.*

We now connect the MIEM property of an evaluation measure G (Definition 3.1) with the annealing property of a match delta Δ (Definition 3.2) with respect to a specific match σ'.

Proposition 3.1. *Let G be an evaluation measure. If G is a MIEM over $\Sigma^2 \subseteq \Sigma^{\subseteq_1}$, then $\forall (\sigma, \sigma') \in \Sigma^2 : \Delta = \sigma' \setminus \sigma$ is a local annealer with respect to G over $\Sigma^2 \subseteq \Sigma^{\subseteq_1}$.*

Proof 3.4. *Let G be an MIEM over $\Sigma^2 \subseteq \Sigma^{\subseteq_1}$. By Definition 3.1, $G(\sigma) \leq G(\sigma')$ holds for every match pair $(\sigma, \sigma') \in \Sigma^2$.*

Assume, by way of contradiction, that exists some Δ that is not a local annealer with respect to G over Σ^2. Thus, there exists some match pair $(\sigma, \sigma') \in \Sigma^2$ such that $\Delta = \Delta_{(\sigma,\sigma')}$ and $G(\sigma) > G(\sigma')$, in contradiction to the fact that G is a MIEM over Σ^2.

Using Theorem 3.1 and Proposition 3.1, one can deduce the following immediate corollary.

Corollary 3.1. *Any singleton correspondence set Δ ($|\Delta| = 1$) is a local annealer with respect to 1) R over $\Sigma^{\subseteq_1}$, 2) P over $\Sigma^P \cap \Sigma^{\subseteq_1}$, and 3) F over $\Sigma^F \cap \Sigma^{\subseteq_1}$.*

Proof 3.5. *Note that $\Sigma^{\subseteq_1} \subseteq \Sigma^{\subseteq}$, and accordingly also $\Sigma^P \cap \Sigma^{\subseteq_1} \subseteq \Sigma^P$ and $\Sigma^F \cap \Sigma^{\subseteq_1} \subseteq \Sigma^F$. Using Theorem 3.1 we can therefore say that recall (R) is a MIEM over $\Sigma^{\subseteq_1}$, precision (P) is a MIEM over $\Sigma^P \cap \Sigma^{\subseteq_1}$, and F1 measure (F) is a MIEM over $\Sigma^F \cap \Sigma^{\subseteq_1}$.*

Then using Proposition 3.1, we can conclude that for all $\Delta_{\sigma,\sigma'}$ s.t. $(\sigma, \sigma') \in \Sigma^{\subseteq_1}/\Sigma^P \cap \Sigma^{\subseteq_1}/\Sigma^F \cap \Sigma^{\subseteq_1}$, $\Delta_{\sigma,\sigma'}$ is a local annealer with respect to R over $\Sigma^{\subseteq_1}$/P over $\Sigma^P \cap \Sigma^{\subseteq_1}$/$F$ over $\Sigma^F \cap \Sigma^{\subseteq_1}$. For any other singleton Δ, the claim is vacuously satisfied.

Assume now that the value of G, applied to a match σ, is not deterministically known. Rather, $G(\sigma)$ is a random variable with an expected value of $E(G(\sigma))$. We extend Definition 3.2 as follows.

Definition 3.3 (Probabilistic Local Match Annealing). *Let G be a random variable, whose values are taken from the domain of an evaluation measure, and Δ be a singleton correspondence set ($|\Delta| = 1$). Δ is a* probabilistic local annealer with respect to G over $\Sigma^2 \subseteq \Sigma^{\subseteq_1}$ *if for every $(\sigma, \sigma') \in \Sigma^2$ s.t. $\Delta = \Delta_{(\sigma,\sigma')}$: $E(G(\sigma)) \leq E(G(\sigma'))$.*

Similar to the analysis in Section 3.1, we now define conditions under which a correspondence is a probabilistic local annealer for recall (R), precision (P), and f-measure (F). We define $\mathbb{I}_{\{\Delta \in \sigma^*\}}$ to be an indicator function, returning a value of 1 whenever Δ is a part of the reference match and 0 otherwise.

$$\mathbb{I}_{\{\Delta \in \sigma^*\}} = \begin{cases} 1 & \text{if } \Delta \in \sigma^* \\ 0 & \text{otherwise} \end{cases}$$

Using $\mathbb{I}_{\{\Delta \in \sigma^*\}}$, we define the probability that Δ is correct:

$$Pr\{\Delta \in \sigma^*\} = Pr\{\mathbb{I}_{\{\Delta \in \sigma^*\}} = 1\}$$

Lemma 3.3 is the probabilistic counterpart of Lemma 3.2.

Lemma 3.3. *For $(\sigma, \sigma') \in \Sigma^{\subseteq_1}$:*

- *$E(P(\sigma)) \leq E(P(\sigma'))$ iff $E(P(\sigma)) \leq Pr\{\Delta \in \sigma^*\}$*
- *$E(F(\sigma)) \leq E(F(\sigma'))$ iff $0.5 \cdot E(F(\sigma)) \leq Pr\{\Delta \in \sigma^*\}$*

Proof 3.6. *Let $(\sigma, \sigma') \in \Sigma^{\subseteq_1}$ be a match pair in $\Sigma^{\subseteq_1}$.*

We first address an extreme case, where the denominator of a precision calculation is zero. Here, this case occurs only when $\sigma = \sigma' = \emptyset$. For this work, we shall define $E(P(\sigma)) = E(P(\sigma')) = -1$, ensuring the validity of the first statement of the lemma.

Next, we analyze the expected values of the evaluation measures. We shall assume that the size of the current match σ is deterministically known and therefore $E(|\sigma|) = |\sigma|$. Let $|\sigma^*|$ and $|\sigma \cap \sigma^*|$ be random variables with expected values of $E(|\sigma^*|)$ and $E(|\sigma \cap \sigma^*|)$, respectively.

- **$\mathbf{E(P(\sigma)) \leq E(P(\sigma'))}$ *iff* $\mathbf{E(P(\sigma)) \leq Pr\{\Delta \in \sigma^*\}}$**: *Similar to Eq. 1, we compute the expected precision value of σ as follows:*

$$E(P(\sigma)) = \frac{E(|\sigma \cap \sigma^*|)}{|\sigma|} \tag{9}$$

Now, we are ready to compute the expected precision value of σ'. Note that the value of the denominator is deterministic, $E(|\sigma'|) = |\sigma| + 1$ since $(\sigma, \sigma') \in \Sigma^{\subseteq_1}$. Then, using $Pr\{\Delta \in \sigma^*\}$ and Eq. 9, we obtain

$$E(P(\sigma')) = Pr\{\Delta \in \sigma^*\} \cdot \frac{E(|\sigma \cap \sigma^*|) + 1}{|\sigma| + 1} + (1 - Pr\{\Delta \in \sigma^*\}) \cdot \frac{E(|\sigma \cap \sigma^*|)}{|\sigma| + 1}$$

While the denominator remains unchanged, the numerator increases by one only if Δ is part of the reference match. After rewriting we obtain

$$E(P(\sigma')) = \frac{Pr\{\Delta \in \sigma^*\} \cdot E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\} + E(|\sigma \cap \sigma^*|) - Pr\{\Delta \in \sigma^*\} \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + 1}$$

and conclude that

$$E(P(\sigma')) = \frac{E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\}}{|\sigma| + 1} \tag{10}$$

$\Rightarrow$: Let $E(P(\sigma)) \leq E(P(\sigma'))$. Using Eq. 9 and Eq. 10 we obtain

$$\frac{E(|\sigma \cap \sigma^*|)}{|\sigma|} \leq \frac{E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\}}{|\sigma| + 1}$$

Multiplying by $|\sigma| \cdot (|\sigma| + 1)$, we get

$$E(|\sigma \cap \sigma^*|) \cdot (|\sigma| + 1) \leq (E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\}) \cdot |\sigma|$$

Using rewriting we obtain

$$E(|\sigma \cap \sigma^*|) \cdot |\sigma| + E(|\sigma \cap \sigma^*|) \leq E(|\sigma \cap \sigma^*|) \cdot |\sigma| + Pr\{\Delta \in \sigma^*\} \cdot |\sigma| \rightarrow$$
$$E(|\sigma \cap \sigma^*|) \leq Pr\{\Delta \in \sigma^*\} \cdot |\sigma|$$

and conclude that

$$E(P(\sigma)) = \frac{E(|\sigma \cap \sigma^*|)}{|\sigma|} \leq Pr\{\Delta \in \sigma^*\}$$

$\Leftarrow$*: Let* $E(P(\sigma)) \leq Pr\{\Delta \in \sigma^*\}$*. Using Eq. 9 we obtain*

$$\frac{E(|\sigma \cap \sigma^*|)}{|\sigma|} \leq Pr\{\Delta \in \sigma^*\} \rightarrow E(|\sigma \cap \sigma^*|) \leq Pr\{\Delta \in \sigma^*\} \cdot |\sigma|$$

by adding $E(|\sigma \cap \sigma^*|) \cdot |\sigma|$ *on both sides and rewriting we conclude that*

$$E(P(\sigma)) = \frac{E(|\sigma \cap \sigma^*|)}{|\sigma|} \leq \frac{E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\}}{|\sigma| + 1} = E(P(\sigma'))$$

- $\mathbf{E(F(\sigma)) \leq E(F(\sigma'))}$ ***iff*** $\mathbf{0.5 \cdot E(F(\sigma)) \leq Pr\{\Delta \in \sigma^*\}}$*:* *The expected F1 measure value of* σ *is given by*

$$E(F(\sigma)) = \frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + E(|\sigma^*|)} \tag{11}$$

Similar to the computation of the precision value, since the size of the match is deterministic, $E(|\sigma'|) = |\sigma| + 1$ *and*

$$E(F(\sigma')) = \frac{2 \cdot E(|\sigma' \cap \sigma^*|)}{E(|\sigma'|) + E(|\sigma^*|)} = \frac{2 \cdot E(|\sigma' \cap \sigma^*|)}{|\sigma| + 1 + E(|\sigma^*|)}$$

The denominator value is deterministically affected by the addition of Δ *(and increased by one). However, the nominator depends on* $Pr\{\Delta \in \sigma^*\}$ *and accordingly, we can rewrite it as*

$$Pr\{\Delta \in \sigma^*\} \cdot \frac{2 \cdot (E(|\sigma \cap \sigma^*|) + 1)}{|\sigma| + 1 + E(|\sigma^*|)} + (1 - Pr\{\Delta \in \sigma^*\}) \cdot \frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + 1 + E(|\sigma^*|)}$$

after rewriting we obtain

$$E(F(\sigma')) = \frac{2 \cdot (E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\})}{|\sigma| + 1 + E(|\sigma^*|} \tag{12}$$

$\Rightarrow$*: Let* $E(F(\sigma)) \leq E(F(\sigma'))$*. Using Eq. 11 and Eq. 12 we obtain*

$$\frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + E(|\sigma^*|)} \leq \frac{2 \cdot (E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\})}{|\sigma| + 1 + E(|\sigma^*|)}$$

multiplying by $(|\sigma| + E(|\sigma^*|)) \cdot (|\sigma| + E(|\sigma^*|) + 1)$ *yields*

$$2 \cdot E(|\sigma \cap \sigma^*|) \cdot (|\sigma| + E(|\sigma^*|) + 1) \leq 2 \cdot (E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\}) \cdot (|\sigma| + E(|\sigma^*|))$$

and by rewriting we get

$$E(|\sigma \cap \sigma^*|) \leq Pr\{\Delta \in \sigma^*\}) \cdot |\sigma| + Pr\{\Delta \in \sigma^*\}) \cdot E(|\sigma^*|)$$

and conclude

$$0.5 \cdot E(F(\sigma)) = 0.5 \cdot \frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + E(|\sigma^*|)} \leq Pr\{\Delta \in \sigma^*\}$$

$\Leftarrow$*: Let* $0.5 \cdot E(F(\sigma)) \leq Pr\{\Delta \in \sigma^*\}$*. Using Eq. 11 we obtain*

$$0.5 \cdot \frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + E(|\sigma^*|)} \leq Pr\{\Delta \in \sigma^*\}$$

by multiplying by $|\sigma| + E(|\sigma^*|)$ *we get*

$$E(|\sigma \cap \sigma^*|) \leq Pr\{\Delta \in \sigma^*\} \cdot |\sigma| + Pr\{\Delta \in \sigma^*\} \cdot E(|\sigma^*|)$$

Similar to above, we add $E(|\sigma \cap \sigma^*|) \cdot (|\sigma| + E(|\sigma^*|))$ *on both sides and get*

$$E(|\sigma \cap \sigma^*|) + E(|\sigma \cap \sigma^*|) \cdot (|\sigma| + E(|\sigma^*|)) \leq$$
$$Pr\{\Delta \in \sigma^*\} \cdot |\sigma| + Pr\{\Delta \in \sigma^*\} \cdot E(|\sigma^*|) + E(|\sigma \cap \sigma^*|) \cdot (|\sigma| + E(|\sigma^*|))$$

by rewriting and multiplying by $\frac{2}{(|\sigma|+E(|\sigma^*|)\cdot(|\sigma|+E(|\sigma^*|)+1)}$ *we get*

$$\frac{2 \cdot E(|\sigma \cap \sigma^*|)}{|\sigma| + E(|\sigma^*|)} \leq \frac{2 \cdot (E(|\sigma \cap \sigma^*|) + Pr\{\Delta \in \sigma^*\})}{|\sigma| + E(|\sigma^*|) + 1}$$

and conclude that

$$E(F(\sigma)) \leq E(F(\sigma'))$$

which finalizes the proof.

The following two subsets, $\Sigma^{E(P)} = \{(\sigma, \sigma') \in \Sigma^{\subseteq_1} : E(P(\sigma)) \leq Pr\{\Delta \in \sigma^*\}\}$ and $\Sigma^{E(F)} = \{(\sigma, \sigma') \in \Sigma^{\subseteq_1} : 0.5 \cdot E(F(\sigma)) \leq Pr\{\Delta \in \sigma^*\}\}$, serve in extending Theorem 3.1 to the probabilistic setting.

Theorem 3.2. *Let* $R/P/F$ *be a random variable, whose values are taken from the domain of* $[0, 1]$*, and* Δ *be a singleton correspondence set (*$|\Delta| = 1$*).* Δ *is a* probabilistic local annealer *with respect to* $R/P/F$ over $\Sigma^{\subseteq_1}/\Sigma^{E(P)}/\Sigma^{E(F)}$.

Proof 3.7. *Let* Δ *be a singleton correspondence set (*$|\Delta| = 1$*). Let* R *be a random variable and let* $\Delta = \Delta_{(\sigma,\sigma')}$ *such that* $(\sigma, \sigma') \in \Sigma^{\subseteq_1}$*.*

According to Corollary 3.1, Δ *is a local annealer with respect to* R *over* $\Sigma^{\subseteq_1}$ *and therefore* $R(\sigma) \leq R(\sigma')$*, regardless of* $Pr\{\Delta \in \sigma^*\}$*. Therefore, for any* $p = Pr\{\Delta \in \sigma^*\}, p \cdot R(\sigma) \leq p \cdot R(\sigma')$ *and by definition of expectation,* $E(R(\sigma)) \leq E(R(\sigma'))$*.*

Let P *be a random variable and let* $\Delta = \Delta_{(\sigma,\sigma')}$ *such that* $(\sigma, \sigma') \in \Sigma^{E(P)}$*. By definition of* $\Sigma^{E(P)}$*,* $E(P(\sigma)) \leq Pr\{\Delta \in \sigma^*\}$ *and using Lemma 3.3 we obtain* $E(P(\sigma)) \leq E(P(\sigma'))$*.*

Let F *be a random variable and let* $\Delta = \Delta_{(\sigma,\sigma')}$ *such that* $(\sigma, \sigma') \in \Sigma^{E(F)}$*. By definition of* $\Sigma^{E(F)}$*,* $E(F(\sigma)) \leq 0.5 \cdot Pr\{\Delta \in \sigma^*\}$ *and using Lemma 3.3 we obtain* $E(F(\sigma)) \leq E(F(\sigma'))$*.*

We can therefore conclude, by Definition 3.3, that Δ *is a probabilistic local annealer with respect to* $R/P/F$ *over* $\Sigma^{\subseteq_1}/\Sigma^{E(P)}/\Sigma^{E(F)}$*.*

3.3 Evaluation Approximation: the Case of a Human Matcher

Theorem 3.2 offers a relaxation to the demands of Theorem 3.1 by defining $\Sigma^{E(P)}$ and $\Sigma^{E(F)}$ over which matches are probabilistic local annealers. A key component to generating these subsets is the computation of $Pr\{\Delta \in \sigma^*\}$ that, in most real-world scenarios, is likely unavailable during the matching process or even after it concludes. To overcome this hurdle, we discuss next the possibility to judicially make use of human matching to assign a probability to the inclusion of a correspondence in the reference match.

The traditional view of human matchers in schema matching is that they offer a reliable assessment on the inclusion of a correspondence in a match. Given a matching decision by a human matcher, we formulate this view, as follows.

Definition 3.4 (Unbiased Matching). *Let* M_{ij} *be a confidence value assigned to an element pair* (a_i, b_j) *and* σ^* *a reference match.* M_{ij} *is* unbiased *(with respect to* σ^**) if* $M_{ij} = Pr\{M_{ij} \in \sigma^*\}$

Unbiased matching allows the use of a matching confidence to assess the probability of a correspondence to be part of a reference match. Using Definition 3.4, we define an unbiased matching matrix M such that $\forall M_{ij} \in M : M_{ij} = Pr\{M_{ij} \in \sigma^*\}$ and an unbiased matching history H such that $\forall h_t \in H : h_t.c = Pr\{M_{ij} \in \sigma^*\}$.

Given an unbiased matching matrix M, a reference match σ^*, a match $\sigma \subseteq M$ and a candidate correspondence $M_{ij} \in M$, we can, using Definition 3.4 and the definition of expectation, compute

$$Pr\{M_{ij} \in \sigma^*\} = M_{ij},$$
$$E(P(\sigma)) = \frac{\sum_{M_{ij}\in\sigma} M_{ij}}{|\sigma|}, E(F(\sigma)) = \frac{2 \cdot \sum_{M_{ij}\in\sigma} M_{ij}}{|\sigma| + |\sigma^*|} \tag{13}$$

and check whether $(\sigma, \sigma \cup \{M_{ij}\}) \in \Sigma^{E(P)}$ and $(\sigma, \sigma \cup \{M_{ij}\}) \in \Sigma^{E(F)}$. In case the size of the reference match $|\sigma^*|$ is unknown, it needs to be estimated, *e.g.*, using 1:1 matching, $|\sigma^*| = min(S, S')$.[2]

The details of the computation of Eq. 13 are as follows:

Computation 1. *We first look into the main component in both expressions $E(|\sigma \cap \sigma^*|)$, that is, the expected number of correct correspondences in a match σ.*

By rewriting we get

$$E(|\sigma \cap \sigma^*|) = E\left(\sum_{M_{ij}\in\sigma} \mathbb{I}_{\{M_{ij}\in\sigma^*\}}\right)$$

and based on the linearity of expectation, we obtain

$$E(|\sigma \cap \sigma^*|) = \sum_{M_{ij}\in\sigma} E(\mathbb{I}_{\{M_{ij}\in\sigma^*\}})$$

The expected value of an indicator equals the probability of an event and thus,

$$\sum_{M_{ij}\in\sigma} E(\mathbb{I}_{\{M_{ij}\in\sigma^*\}}) = \sum_{M_{ij}\in\sigma} Pr\{M_{ij} \in \sigma^*\}$$

Assuming unbiased matching (Definition 3.4), we conclude

$$E(|\sigma \cap \sigma^*|) = \sum_{M_{ij}\in\sigma} M_{ij}$$

Using Eq.1 and recalling that $|\sigma|$ is deterministic, we obtain

$$E(P(\sigma)) = \frac{\sum_{M_{ij}\in\sigma} M_{ij}}{|\sigma|}$$

Similarly, using Eq. 7 and assuming that $|\sigma^*|$ is also deterministic, we obtain

$$E(F(\sigma)) = \frac{2 \cdot \sum_{M_{ij}\in\sigma} M_{ij}}{|\sigma| + |\sigma^*|}$$

3.3.1 Biased Human Matching.

In Section 2.3 we presented a human matching decision process as a history, from which a matching matrix may be derived. The decisions in the history represent corresponding element pairs chosen by the human matcher and their assigned confidence level (see Definition 2.2). Assuming unbiased human matching (Definition 3.4), the assigned confidence can be used to determine which of the selected correspondences should be added to the current match, given an evaluation measure of choice (recall, precision, or f-measure), using Eq. 13.

An immediate question that comes to mind is whether human matching is indeed unbiased. Figure 5 illustrates of the relationship between human confidence in matching and two derivations of an empirical probability distribution estimation of $Pr\{M_{ij} \in \sigma^*\}$. The results are based on our experiments (see Section 5 for details). We partitioned the confidence levels into 10 buckets (x-axis) to allow an estimation of an expected value (blue dots) and standard deviation (vertical lines) of the accuracy results of decisions within a bucket of confidence. Each bucket includes at least 500 examples and the estimation within each bucket was calculated as the proportion of correct correspondences out of all correspondences in the bucket. The red dotted line represents theoretical unbiased matching. It is clearly illustrated that human matching **is** biased. Therefore, human subjective confidence is unlikely to serve as a (single) good predictor to matching correctness. This understanding, combined with the analysis of matching as a process serves us next in offering an algorithmic solution to create quality matches.

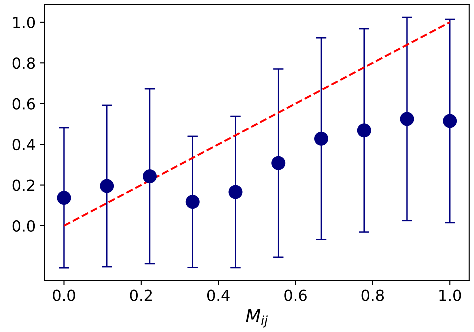


Figure 5: Is human matching biased? Confidence by correctness partitioned to 0.1 buckets.

4 PROCESS-AWARE MATCHING COLLABORATION WITH POWAREMATCH

We begin this section by providing in Section 4.1 a decision history processing strategy under the (utopian) unbiased matching assumption (Definition 3.4) following the observations of Section 3. Then, we describe PoWareMatch, our proposed solution to handle biased matching (Section 4.2) and detail its components (sections 4.3-4.4).

4.1 History Processing with Unbiased Matching

Ideally, human matching is unbiased. That is, each matching decision $h_t \in H$ in the decision history (Definition 2.2) is accompanied by an unbiased (accurate) confidence value $h_t.c = Pr\{M_{ij} \in \sigma^*\}$ (Definition 3.4). Accordingly, we can use the observations from Section 3 to produce a better match as a solution to Problem 1 (with respect to some evaluation measure) out of a decision history.

Let $h_t \in H$ be a matching decision made at time t, such that $h_t.e = (a_i, b_j)$. Aiming to generate a match, we have two options, either adding the correspondence M_{ij} to the match or not. Targeting recall, an MIEM over $\Sigma^{\subseteq}$ (Theorem 3.1), the choice is clear. Since $\{M_{ij}\}$ is a local annealer with respect to recall over $\Sigma^{\subseteq_1}$ (Corollary 3.1), we always benefit by adding it. Focusing on precision or f-measure, the decision depends on the current match (termed σ_{t-1}). With unbiased matching, we can simply utilize Eq. 13 to estimate the values of $Pr\{M_{ij} \in \sigma^*\}$, $E(P(\sigma_{t-1}))$, and $E(F(\sigma_{t-1}))$ and reach a decision using Theorem 3.2. Specifically, if we target precision and $E(P(\sigma_{t-1})) \leq M_{ij}$, then $\{M_{ij}\}$ is a probabilistic local annealer with respect to P, and we can increase precision by adding M_{ij} to the match ($\sigma_t = \sigma_{t-1} \cup \{M_{ij}\}$). Similarly, targeting f-measure, if

[2] The computation Eq. 13 is given, due to space consideration, in a technical report [6].

$0.5 \cdot E(F(\sigma_{t-1})) \leq M_{ij}$, then $\{M_{ij}\}$ is a probabilistic local annealer with respect to F, and adding M_{ij} to the match does not decrease it.

Example 2. *Figure 6 illustrates history processing over the history from Figure 4. The left column presents the targeted evaluation measure and at the bottom, a not-to-scale timeline lays out the decision history. Along each row,* ✓ *and* ✗ *represent whether* M_{ij} *is added to the match or not, respectively. Targeting recall (top row), all decisions are accepted and therefore a high final recall value of* 0.75 *is obtained.*

For precision and f-measure, each arrow is annotated with a decision threshold, which is set by $E(P(\sigma_{t-1}))$ *and* $0.5 \cdot E(F(\sigma_{t-1}))$*, whose computation is given in Eq. 13. At the beginning of the process,* $\sigma_0 = \emptyset$ *and* $E(P(\sigma_0))$ *is set to 0 by definition.* $E(F(\sigma_0)) = 0$ *since there are no correspondences in* σ_0*. To illustrate the process consider, for example, the second threshold, computed based on the match* $\{M_{11}\}$*, is* $\frac{0.9}{1.0} = 0.9$ *in the second row and* $0.5 \cdot \frac{2 \cdot 0.9}{1+4} = 0.18$ *in the third row.*

4.1.1 Setting a Static (Global) Threshold.

The decision making process above assumes the availability of (an unlabeled) σ_{t-1}. Whenever we do not know σ_{t-1}, *e.g.*, when partitioning the matching task over multiple matchers [54], we can set a static (global) threshold. Adding M_{ij} is always guaranteed to improve recall and, thus, the strategy targeting recall remains the same, *i.e.*, the static threshold will be set to 0. For precision and f-measure, by setting $E(P(\sigma_{t-1}))$ and $E(F(\sigma_{t-1}))$ to their upper bound, 1, we obtain a global condition to add M_{ij} if $1 \leq M_{ij}$ and $0.5 \leq M_{ij}$, respectively. To sum, targeting $R/F/P$ with a static threshold is done by adding a correspondence to a match if its confidence exceeds $0/0.5/1$, respectively.

When setting a static threshold, ongoing decisions are not taken into account. Recalling Example 2, using a static threshold for f-measure will reject the third decision ($0.5 > M_{12} = 0.25$) (unlike when using σ_{t-1}), resulting in a lower final f-measure of 0.67.

4.2 PoWareMatch Architecture Overview

In Section 3.3.1 we demonstrated that human matching may be biased, in which case Eq. 13 cannot be used as is. Therefore, we next present PoWareMatch, aiming to calibrate biased matching decisions and to predict the values of P and F. PoWareMatch is a matching algorithm that enriches the representation of each matching decision with cognitive aspects, algorithmic assessment and neural encoding of prior decisions using LSTM. Compensating for (possible) lack of evaluation regarding former decisions, PoWareMatch repeatedly predicts missing precision and f-measure values (learned during a training phase) that are used to monitor the decision making process (see Section 3). Finally, to reach a complete match and boost recall, PoWareMatch uses algorithmic matching to complete missing values that were not inserted by human matchers.

The flow of the PoWareMatch algorithm is illustrated in Figure 7. Its input is a schema pair (S, S') and a decision history H (Definition 2.2) and its output is a match $\hat{\sigma}$. PoWareMatch is composed of two components, *HP*, aiming to calibrate matching decisions (Section 4.3) and *RB*, focusing on recall boosting (Section 4.4).

4.3 Calibrating Matching Decisions

The main component of PoWareMatch calibrates matching decisions by history processing (*HP*). Matching decisions are processed in the order they were assigned according to the history. The goal of *HP* is to improve the estimation of $Pr\{M_{ij} \in \sigma^*\}$ beyond the M_{ij} value, which is an accurate assessment only for unbiased matching. We use $Pr\{e_t\}$ here as a shorthand writing for the probability of an element pair assigned at time t to be correct ($Pr\{h_t.e \in \sigma^*\}$).

We first propose a feature representation of matching history decisions (Section 4.3.1) to be processed using a recurrent neural network that is trained in a supervised manner to capture latent temporal properties of the matching process. Once trained, the network predicts a set of labels $\langle \hat{Pr}\{e_t\}, \hat{P}(\sigma_{t-1}), \hat{F}(\sigma_{t-1}) \rangle$ regarding each decision h_t to produce a match σ_{HP} (Section 4.3.2).

4.3.1 Turning Biases into Features.

We propose a feature encoding of a human matcher decision h_t using a 4-dimensional feature vector, composed of the reported confidence, allocated time, consensual agreement, and an algorithmic similarity score. Allocated time and consensual agreement (along with control, the extent of which the matcher was assisted by an algorithmic solution, which we do not use here since it was shown to be less predictive in our experiments) are human biases studied by Ackerman *et al.* [8]. Allocated time is measured directly using history timestamps. As for consensual agreement, we use an $n \times m$ matrix A such that $a_{ij} \in A$ represents the number of other human matchers that determine $a_i \in S$ and $b_j \in S'$ to correspond. An algorithmic matching result is given as an $n \times m$ similarity matrix $\tilde{M}$.

Let $h_t \in H$ be a matching decision at time t (Definition 2.2) regarding entry $h_t.e = (a_i, b_j)$. We create a feature encoding $v_t \in \mathbb{R}^4$ given by $v_t = \langle h_t.c, \delta_t, a_e, \tilde{M}_e \rangle$, where

- $h_t.c$ is the confidence value associated with h_t,
- $\delta_t = h_t.t - h_{t-1}.t$ is the time spent until determining h_t,
- $A_e = A_{ij}$ is the consensus regarding the entry assigned in the decision h_t,
- $\tilde{M}_e = \tilde{M}_{ij}$, an algorithmic similarity score regarding (a_i, b_j).

v_t enriches the reported confidence ($h_t.c$) with additional properties of a matching decision including possible biases and an algorithmic opinion. A simple solution, which we analyze as a baseline in our experiments (*ML*, Section 5) applies an out-of-the-box machine learning method to predict $\langle \hat{Pr}\{e_t\}, \hat{P}(\sigma_{t-1}), \hat{F}(\sigma_{t-1}) \rangle$. Yet, to encode the sequential nature of a (human) matching process, we next propose the use of recurrent neural networks (LSTM) to process the expended representation of a decision v_t.

4.3.2 History Processing with Biased Matching.

Using decision encoding v_t, we now turn our effort to improve biased matching decisions. For biased matching we train a neural network to estimate $Pr\{M_{ij} \in \sigma^*\}$, $E(P(\sigma_{t-1}))$ and $E(F(\sigma_{t-1}))$ instead of using Eq. 13. Therefore, the *HP* component consists three (supervised) neural models, a classifier predicting $\hat{Pr}\{e_t\}$ and two regressors predicting $\hat{P}(\sigma_{t-1})$ and $\hat{F}(\sigma_{t-1})$ (enlarged component in Figure 7).

With deep learning, it is common to expect a large training dataset. While with human matching, training data is scarce, the sequential decision making process (formalized as a decision history) that is unique to human matchers (as opposed to algorithmic matchers) offers an opportunity for collecting a reasonable size training set that we use to train recurrent neural networks, and specifically long short-term memory (LSTMs). This is a natural choice when processing a sequential decision making process [35]. LSTMs use a gating scheme to control the amount of information

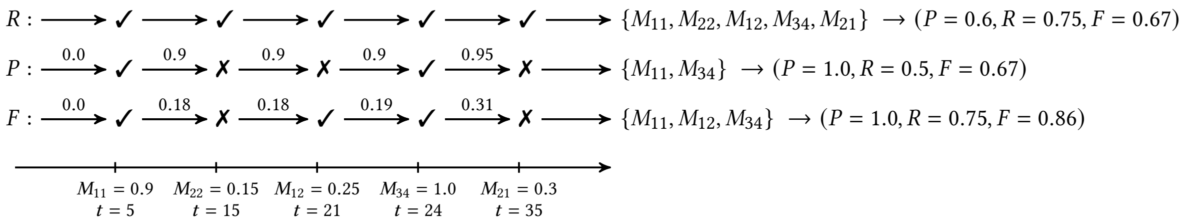

R :
✓
✓
✓
✓
✓
$\{M_{11}, M_{22}, M_{12}, M_{34}, M_{21}\}$ → $(P = 0.6, R = 0.75, F = 0.67)$
P :
0.0
✓
0.9
✗
0.9
✗
0.9
✓
0.95
✗
$\{M_{11}, M_{34}\}$ → $(P = 1.0, R = 0.5, F = 0.67)$
F :
0.0
✓
0.18
✗
0.18
✓
0.19
✓
0.31
✗
$\{M_{11}, M_{12}, M_{34}\}$ → $(P = 1.0, R = 0.75, F = 0.86)$
$M_{11} = 0.9$
$t = 5$
$M_{22} = 0.15$
$t = 15$
$M_{12} = 0.25$
$t = 21$
$M_{34} = 1.0$
$t = 24$
$M_{21} = 0.3$
$t = 35$

Figure 6: History Processing Example.

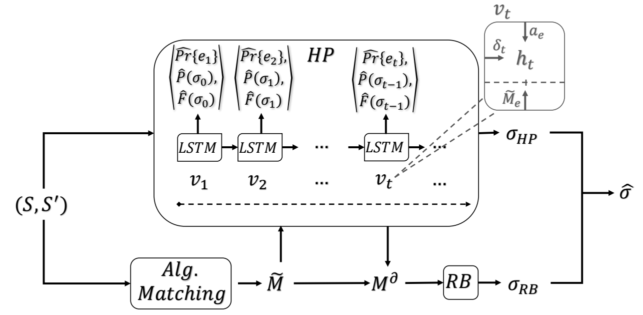

v_t
a_e
δ_t
h_t
$\widetilde{M}_e$
HP
$\widehat{Pr}\{e_1\}$, $\hat{P}(\sigma_0)$, $\hat{F}(\sigma_0)$
$\widehat{Pr}\{e_2\}$, $\hat{P}(\sigma_1)$, $\hat{F}(\sigma_1)$
$\widehat{Pr}\{e_t\}$, $\hat{P}(\sigma_{t-1})$, $\hat{F}(\sigma_{t-1})$
LSTM
LSTM
LSTM
...
v_1
v_2
...
v_t
...
σ_{HP}
$\hat{\sigma}$
(S, S')
Alg. Matching
$\widetilde{M}$
M^{∂}
RB
σ_{RB}

Figure 7: PoWareMatch **Framework.**

to preserve at each timestamp using a hidden state. Using LSTM, we can train a model that, when assessing a matching decision h_t, takes advantage of matcher's previous decisions ($\{h_{t'} \in H | t' < t\}$).

Applying LSTM for a decision vector v_t yields a recurrent representation of the decision, after which, we apply $tanh$ activation to obtain a hidden representation h_t^{LSTM}. Using h_t^{LSTM}, the classifier applies a $softmax(\cdot)$ function and the regressors apply a $sigmoid(\cdot)$ function after reducing the dimension to the label dimension (2 and 1, respectively). The softmax function returns a two-dimensional probabilities vector, *i.e.,* the i'th entry represents a probability that the classification is i, for which we take the entry representing 1 as $\hat{Pr}\{e_t\}$. At each time t we obtain $\hat{Pr}\{e_t\}$, $\hat{P}(\sigma_{t-1}) = sigmoid(h_t^{LSTM})$ and $\hat{F}(\sigma_{t-1}) = sigmoid(h_t^{LSTM})$ whose labels during training are $\mathbb{I}_{\{\{M_{ij}:e_t=(a_i,b_j)\}\in\sigma^*\}}$ (an indicator stating whether the decision is a part of the reference match, see Section 3.2), $P(\{M_{ij}|e_{t'} = (a_i, b_j) \wedge h_{t'} \in H \wedge t' < t\})$ and $F(\{M_{ij}|e_{t'} = (a_i, b_j) \wedge h_{t'} \in H \wedge t' < t\})$ (computed according to Eq. 1 over a match composed of prior decisions), respectively. Note that h_t^{LSTM} encodes latent information regarding timestamps $\{t'|t' < t\}$ and thus assists in predicting σ_{t-1}.

Once trained, the history processing of H is similar to the one described in Section 4.1. Let $h_t \in H$ be a matching decision made at time t and σ_{t-1} the match **generated until** time t. When targeting recall we accept every decision, to produce σ_{HP}. Targeting precision (f-measure), we predict $\hat{Pr}\{e_t\}$ and $\hat{P}(\sigma_{t-1})$ ($\hat{F}(\sigma_{t-1})$) and add $\{M_{ij}|e_t = (a_i, b_j)\}$ to the match ($\sigma_t = \sigma_{t-1} \cup \{M_{ij}\}$) if $\hat{P}(\sigma_{t-1}) \leq \hat{Pr}\{e_t\}$ ($0.5 \cdot \hat{F}(\sigma_{t-1}) \leq \hat{Pr}\{e_t\}$). A static threshold may also be applied here (see Section 4.1.1). Table 1 summarizes history processing. Finally, HP returns the match $\sigma_{HP} = \sigma_T$.

Table 1: History processing. given target and threshold.

target→ ↓**threshold**	R	P	F
dynamic	0.0	$\hat{P}(\sigma_{t-1})$	$0.5 \cdot \hat{P}(\sigma_{t-1})$
static	0.0	1.0	0.5

4.4 Recall Boosting

The HP component improves on the precision of the human matcher and is limited to correspondences that were chosen by a human matcher, which may result in a match with low recall. To better cater to recall, we present next a complementary component, RB, to boost recall using adaptation of existing 2LMs (see Section 2.2).

RB uses algorithmic match results for correspondences that were not considered by a human matcher, formally stated as $M^{\partial} = \{M_{ij} | \forall h \in H, h.e \neq (a_i, b_j)\}$. When a human matcher does not offer an opinion on a correspondence, it may be intentional or caused inadvertently and the two are indistinguishable. RB complements human matching by analyzing all unassigned correspondences.

We start by presenting RB as a general threshold-based 2LM, from which the 2LMs can be derived. Note that unlike traditional 2LMs, which are applied over a whole similarity matrix, RB is applied to M^{∂} only, letting HP handle human decisions as discussed in Section 4.3. Given a partial matrix M^{∂} and a set of thresholds $\nu = \{\nu_{ij}\}$, RB is defined as follows:

$$RB(M^{\partial}, \nu) = \{M_{ij}^{\partial} | M_{ij}^{\partial} \geq \nu_{ij}\} \tag{14}$$

RB forms a natural implementation of Threshold [18] by defining ν^{th} to assign a single threshold value to all entries such that $\nu_{i,j} = \nu^{th}$ for all matrix entries (i, j). To adapt Max-Delta to work with M^{∂}, we separate the decision-making process by cases. Let M_{ij} be the entry under consideration. In the case that a human matcher did not choose any value in row i, RB adds M_{ij} to σ_{RB} if $M_{ij} > 0$. There are two possible scenarios when a human matcher chose an entry in row i. If none of the entries a human matcher chose in row i are included in σ_{HP}, then we are back to the first case, adding M_{ij} to σ_{RB} if $M_{ij} > 0$. Otherwise, some entry in row i has been included in σ_{HP} and by using a hyper-parameter θ,[3] M_{ij} is added to σ_{RB} if it satisfies the Max-Delta revised condition: $M_{ij} > 1 - \theta$.

[3] we use θ here to avoid confusion with a δ function notation.

In what follows, each threshold in the set $v^{md}(\theta)$ is formally defined as follows:[4]

$$v_{ij}^{md}(\theta) = \mathrm{I}_{\{\exists j' \in \{1,2,\ldots,m\}: M_{ij'} \in \sigma_{HP}\}} \cdot \varepsilon - \theta \cdot (1 - \mathrm{I}_{\{\nexists j' \in \{1,2,\ldots,m\}: M_{ij'} \in \sigma_{HP}\}}) \tag{15}$$

A straightforward variation of Max-Delta can be derived by looking at the columns instead of the rows of each entry, as initially suggested by Do *et al.* [18].

The implementation of such variant of Eq. 15 is given by

$$v_{ij}^{md(col)}(\theta) = \mathrm{I}_{\{\exists i' \in \{1,2,\ldots,n\}: M_{ij'} \in \sigma_{HP}\}} \cdot \varepsilon - \theta \cdot (1 - \mathrm{I}_{\{\nexists i' \in \{1,2,\ldots,n\}: M_{ij'} \in \sigma_{HP}\}}) \tag{16}$$

Finally, we offer an extension to the original implementation of Dominants by introducing a tuning window (distance from maximal value) similar to the one defined for Max-Delta. The inference is similar to the one applied for $v^{md}(\theta)$ and each threshold in the set $v^{dom}(\theta)$ can be defined as follows:

$$v_{ij}^{dom}(\theta) = \mathrm{I}_{\{\exists i' \in \{1,\ldots,n\}: M_{ij'} \in \sigma_{HP} \vee \exists j' \in \{1,\ldots,m\}: M_{ij'} \in \sigma_{HP}\}} \cdot \varepsilon - \theta \cdot (1 - \mathrm{I}_{\{\nexists i' \in \{1,\ldots,n\}: M_{ij'} \in \sigma_{HP} \wedge \exists j' \in \{1,\ldots,m\}: M_{ij'} \in \sigma_{HP}\}}) \tag{17}$$

The *RB* methods introduced above are all accompanied by a hyper-parameter (a uniform v^{th} for Threshold and θ for the Max-Delta's variations and Dominants) controlling the threshold. In this work we focus on a uniform threshold, which also yielded the best results in our empirical evaluation (see Section 5.4).

Finally, PoWareMatch combines the two generated matches σ_{HP} and σ_{RB} to return $\hat{\sigma} = \sigma_{HP} \cup \sigma_{RB}$.

5 EMPIRICAL EVALUATION

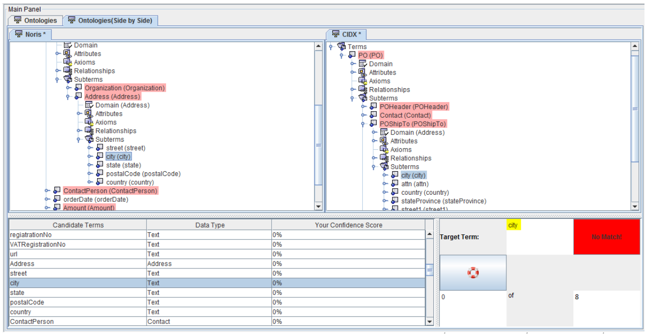
In this work we focus on human matching as a decision making process. Accordingly, different from a typical crowdsourcing setting, the human matchers in our experiments are free to choose their own order of matching elements and to decide on which correspondences to report (as illustrated in Figure 1). Section 5.1 describes human matching datasets using two matching tasks and Section 5.2 details the experimental setup. Our analysis shows that

- Simple process-aware inference, using self-reported confidence, suffices to improve matching outcome (Section 5.3).
- PoWareMatch effectively calibrates human matching to provide high precision values (Section 5.3) and, by recall boosting, produces improved overall matching outcome (Section 5.4).
- PoWareMatch generalizes (without training a new model) beyond the domain of schema matching to the domain of ontology alignment (Section 5.5).

5.1 Human Matching Datasets

The datasets were gathered via a controlled experiment, where Science/Engineering undergraduates who studied database management course were used as human matchers. The study was approved by the institutional review board and four pilot participants completed the task prior to the study to ensure its coherence and instruction legibility. Participants were briefed in matching prior to the task, after which they were trained on a pair of short schemata from the *Thalia* dataset [31] prior to performing the main matching tasks. A subset of the dataset is available at [1].[5]

The main matching tasks were chosen from two domains, one of a schema matching task and the other of an ontology alignment task (which is used to demonstrate generalizability, see Section 5.5). The schema matching task was taken from the *Purchase Order* (PO) dataset [18] with schemata of medium size, having 142 and 46 attributes, and with high information content (labels, data types, and instance examples). A total of 7, 618 matching decisions from 175 human matchers were gathered for the PO dataset. The ontology alignment [23] task was taken from the OAEI 2011 and 2016 competitions [3], containing ontologies with 121 and 109 elements with high information content as well. A total of 1, 562 matching decisions from 34 human matchers were gathered for the OAEI dataset. Schema matching and ontology alignment offer different challenges, where ontology elements differ in their characteristics from schemata attributes. Element pairs vary in their difficulty level, introducing a mix of both easy and complex matches.



Domain
Attributes
Axioms
Relationships
Subterms
Organization (Organization)
Address (Address)
Domain (Address)
Attributes
Axioms
Relationships
Subterms
street (street)
city (city)
state (state)
postalCode (postalCode)
country (country)
ContactPerson (ContactPerson)
orderDate (orderDate)
Amount (Amount)

Terms
PO (PO)
Domain
Attributes
Axioms
Relationships
Subterms
POHeader (POHeader)
Contact (Contact)
POShipTo (POShipTo)
Domain (Address)
Attributes
Axioms
Relationships
Subterms
city (city)
attn (attn)
country (country)
stateProvince (stateProvince)
street1 (street1)

Candidate Terms	Data Type	Your Confidence Score
regiatrationNo	Text	0%
VATRegistrationNo	Text	0%
url	Text	0%
Address	Address	0%
street	Text	0%
city	Text	0%
state	Text	0%
postalCode	Text	0%
country	Text	0%
ContactPerson	Contact	0%

Figure 8: User Interface Example.

The interface that was used in the experiments is an upgraded version of the Ontobuilder research environment [41], an open source prototype [4]. An illustration of the user interface is given in Figure 8. Schemata are presented as foldable trees of terms (attributes). When selecting an attribute from the target schema, the match table presents a list of candidate attributes synchronized with the candidate schema tree. Selecting an element reveals additional information about it in a properties box. When a matcher selects an attribute, time until reaching a decision is recorded. Match confidence is inserted by participants as a value in $[0, 1]$ and timestamped to construct a history.

5.2 Experimental Setup

Evaluation was performed on a server with 2 Nvidia gtx 2080 Ti and a CentOS 6.4 operating system. Networks were implemented using PyTorch [7] and the code repository is available online [5]. The *HP* component of PoWareMatch was implemented according to Section 4.3.2, using an LSTM hidden layer of 64 nodes and a 128 nodes fully connected layer. We used Adam optimizer with default configuration ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) with cross entropy and mean squared error loss functions for classifiers ($\hat{Pr}\{e_t\}$, Section 4.3) and regressors ($\hat{P}(\sigma_{t-1})$ and $\hat{F}(\sigma_{t-1})$), respectively, during training.

[4] $\varepsilon \sim 0$ is a very small number assuring a strict inequality.

[5] We intend to make the full datasets public upon acceptance.

5.2.1 Evaluation Measures. We evaluate the matching quality using precision (P), recall (R), and f-measure (F) (see Section 2.1). In addition, we introduce four measures to account for biased matching (Section 3.3), which we use for evaluation in Section 5.3.2. Two of the proposed measures compute the correlation between the estimated values (*e.g.,* M_{ij}) and the true values (*e.g.,* $Pr\{M_{ij} \in \sigma^*\}$) and two compute the accumulated error. For the former we use Pearson correlation coefficient (r) measuring linear correlation and Kendall rank correlation coefficient (τ) measuring the ordinal association. When measuring the bias of M_{ij}, the r and τ values for a match σ are given by:

$$r = \frac{\sum_{M_{ij} \in \sigma}(M_{ij} - \bar{\sigma}) \cdot (\mathbb{I}_{\{M_{ij} \in \sigma^*\}} - P(\sigma))}{\sqrt{\sum_{M_{ij} \in \sigma}(M_{ij} - \bar{\sigma})^2} \cdot \sqrt{\sum_{M_{ij} \in \sigma}(\mathbb{I}_{\{M_{ij} \in \sigma^*\}} - P(\sigma))^2}} \tag{18}$$

where $\bar{\sigma} = \frac{\sum_{M_{ij} \in \sigma} M_{ij}}{|\sigma|}$ represents the average of a match σ and $P(\sigma)$ is the precision of σ (see Eq. 1).

$$\tau = \frac{C - D}{C + D} \tag{19}$$

where C and D represent the number of concordant and discordant pairs.

When measuring the bias of P and F (for convenience we use G in the formulas), r is given by:

$$r = \frac{\sum_{k=1}^{K}(\hat{G}(\sigma_k) - \bar{\hat{G}}(\sigma)) \cdot (G(\sigma_k) - \bar{G}(\sigma_k))}{\sqrt{\sum_{k=1}^{K}(\hat{G}(\sigma_k) - \bar{\hat{G}}(\sigma))^2} \cdot \sqrt{\sum_{k=1}^{K}(G(\sigma_k) - \bar{G}(\sigma_k))^2}} \tag{20}$$

where $\hat{G}$ is the estimated value of G and $\bar{G}(\sigma) = \frac{\sum_{k=1}^{K} G(\sigma_k)}{K}$ is the average of G.

For the latter, we use root mean squared error (RMSE) and mean absolute error (MAE), computed as follows (for the match σ):

$$RMSE = \sqrt{\frac{1}{|\sigma|} \sum_{M_{ij} \in \sigma} (\mathbb{I}_{\{M_{ij} \in \sigma^*\}} - M_{ij})^2}, MAE = \frac{1}{|\sigma|} \sum_{M_{ij} \in \sigma} |\mathbb{I}_{\{M_{ij} \in \sigma^*\}} - M_{ij}| \tag{21}$$

and for an evaluation measure G, as follows:

$$RMSE = \sqrt{\frac{1}{K} \sum_{k=1}^{K}(G(\sigma_k) - \hat{G}(\sigma_k))^2}, MAE = \frac{1}{K} \sum_{k=1}^{K} |G(\sigma_k) - \hat{G}(\sigma_k)| \tag{22}$$

5.2.2 Methodology. Sections 5.3-5.5 provide an analysis of PoWareMatch's performance. We analyze the ability of PoWareMatch to improve on decisions taken by human matchers (Section 5.3) and to improve the overall final matching (Section 5.4). Finally, we analyze PoWareMatch's generalizability to the domain of ontology alignment (Section 5.5). The experiments were conducted as follows:

Human Matching Improvement (Section 5.3): Using 5-fold cross validation over the human schema matching dataset (PO task, see Section 5.1), we randomly split the data into 5 folds and repeat an experiment 5 times with 4 folds for training (140 matchers) and the remaining fold (35 matchers) for testing. We report on average performance over all human matchers from the 5 experiments. Matches for each human matcher are created according to Section 4.3. We report on PoWareMatch's ability to calibrate biased matching (Section 5.3.2). An ablation study is reported in Section 5.3.3, for which we trained and tested 6 additional *HP* implementations using a 5-fold cross validation as before. In this analysis we explore the feature representation of a matching decision (see Section 4.3.1) by either solely using or discarding 1) confidence ($h_t.c$), 2) cognitive aspects (δ_t and a_e), or 3) algorithmic input ($\tilde{M}_e$).

PoWareMatch**'s Overall Performance (Section 5.4):** We assess the overall performance of PoWareMatch. In Section 5.4.1, we also report on two subgroups of human matchers representing top 10% (Top-10) and bottom 10% (Bottom-10) performing human matchers. Threshold selection during **training** is analyzed in Section 5.4.2.

Generalizing to Ontology Alignment (Section 5.5): This experiment demonstrates the generalizability of PoWareMatch. We train a model over the full set of human schema matchers that performed the PO task (175 matchers) and tested it over human matchers that performed ontology alignment over the OAEI task (34 matchers).

We produce $\tilde{M}$ (an algorithmic match, see Section 4) using the matchers presented in Section 2.2 and ADnEV, a state-of-the-art aggregated matcher [56]. A comparison of the algorithmic matchers performance over several thresholds in terms of precision, recall and F1 measure is given in Figure 9. The comparison shows the superiority of ADnEV in high threshold levels. Thus, we present the results of PoWareMatch using ADnEV for recall boosting.

Statistically significant differences in performance are tested using a paired two-tailed t-test with Bonferroni correction for 95% confidence level, and marked with an asterisk.

5.2.3 Baselines. We next describe two types of baselines that correspond to the experiment types we conducted, as follows:

Human analysis and improvement (Section 5.3): First, when PoWareMatch targets recall, it accepts all human judgments (see Theorems 3.1 and 3.2). This also represents traditional methods using human input as ground truth (in this work referred as "unbiased matching assumption"). We use two additional types of baselines to evaluate PoWareMatch performance improving human matching:

1) **raw**: human confidence with threshold filtering that follows Section 4.1. For example, targeting F with static threshold (0.5, see Table 1) represents a likelihood-based baseline that accepts decisions assigned with a confidence level greater than 0.5.

2) **ML**: non process-aware machine learning using common classifiers (*e.g.,* SVM) and regressors (*e.g.,* Lasso), selected during training, to replace the neural process-aware classifier/regressor of *HP*.

Matching improvement for schema matching (Section 5.4) and ontology alignment (Section 5.5): We use four algorithmic matching baselines, namely, Term, WordNet, and Token Path, and ADnEV, the state-of-the-art deep learning algorithmic matching [56]. For each, we applied several thresholds (see Figure 9) during training and report the top performing threshold. Since ADnEV shows the best overall performance ($F = 0.73$), we combine it with human matching (*raw*) to create a human-algorithm baseline (*raw*-ADnEV).

5.3 Improving Human Matching

We analyze the ability of PoWareMatch's *HP* component to improve the precision of human matching decisions (Section 5.3.1)

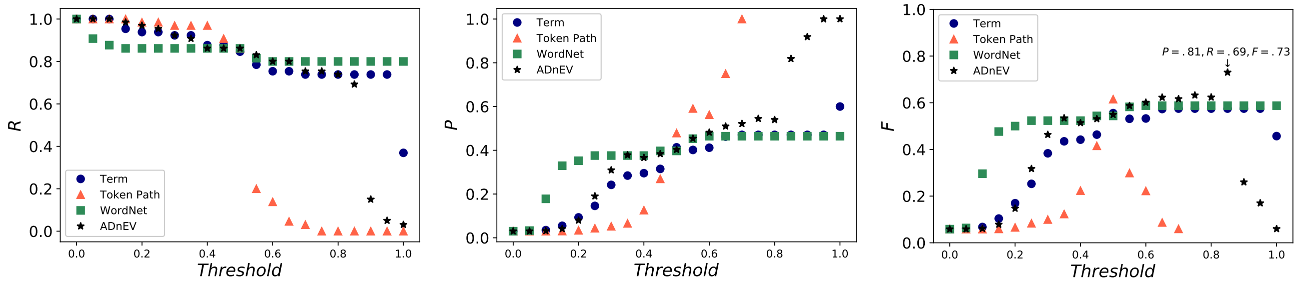


Figure 9: Precision (P), recall (R), and the f-measure (F) of the algorithmic matchers used in the experiments.

and to calibrate human confidence, potentially achieving unbiased matching (Section 5.3.2). We then provide feature analysis via an ablation study in Section 5.3.3.

5.3.1 Precision. Table 2 provides a comparison of results in terms of precision (P) and its computational components ($|\sigma \cap \sigma^*|$ and $|\sigma|$, see Eq. 1) when applying PoWareMatch to two baselines, namely *raw* (assuming unbiased matching) and *ML* (applying non-process aware learning), see details in Section 5.2.3. We split the comparison by target measure (see Section 4), namely targeting 1) recall (R),[6] 2) precision (P) with static (1.0) and dynamic ($\hat{P}(\sigma_{t-1})$) thresholds, and 3) f-measure (F) with static (0.5) and dynamic ($0.5 \cdot \hat{F}(\sigma_{t-1})$) thresholds (as illustrated in Table 1). The best results within each target measure (+threshold) are marked in bold.

Table 2: True positive ($|\sigma \cap \sigma^*|$), match size ($|\sigma|$), and precision (P) by target measure, applying history processing (HP)

Target	(threshold)	*HP*	$\mid \sigma \cap \sigma^* \mid$	$\mid \sigma \mid$	*P*
R	0.0	-	19.02	43.53	0.549
P	1.0	*raw*	10.79	19.91	0.656
		ML	14.01*	14.02	0.999*
		PoWareMatch	**15.80***	15.81	**0.999***
	$\hat{P}(\sigma_{t-1})$	*raw*	11.34	21.69	0.612
		ML	16.21*	18.63	0.858*
		PoWareMatch	**16.50***	16.62	**0.987***
F	0.5	*raw*	**18.19**	36.90	0.574
		ML	16.16	17.97	0.890*
		PoWareMatch	16.65	16.73	**0.993***
	$0.5 \cdot \hat{F}(\sigma_{t-1})$	*raw*	**18.05**	39.65	0.552
		ML	16.95	22.16	0.759
		PoWareMatch	16.97	17.32	**0.968***

The first row of Table 2 represents the decision making when targeting recall, *i.e.*, accepting **all** human decisions.[6] Comparing this baseline with *raw* results (using reported confidence as in Section 4.1 by target measure), we see a clear benefit treating matching as a process. Even when assuming unbiased matching (*raw*), as in Section 4.1, we achieve average precision improvement of 9%.

PoWareMatch achieves a statistically significant precision improvement over *raw*, with an average improvement of 65%, targeting both P and F with both static and dynamic thresholds. PoWareMatch also outperforms the learning-based baseline (*ML*), achieving 13.6% higher precision on average. Note that even when the precision is similar, PoWareMatch generates larger matches ($|\sigma|$), which can improve recall and f-measure, as discussed in Section 5.4. Since PoWareMatch performs process-aware learning (LSTM), its improvement over *ML* supports the use of matching as a process.

PoWareMatch achieves highest precision when targeting P with static threshold, forcing the algorithm to accept only decisions for which PoWareMatch is fully confident ($\hat{Pr}\{e_t\} = 1$). Alas, it comes at the cost match size (less than 16 correspondences on average). This indicates that being conservative is beneficial for precision, yet it has a disadvantage when it comes to recall (and accordingly f-measure). Targeting F, especially with dynamic thresholds, balances match size and precision. This observation becomes essential when analyzing the full performance of PoWareMatch (Section 5.4).

5.3.2 Calibration. In the final analysis of the *HP* component, we quantify its ability to calibrate human confidence to potentially achieve unbiased matching. Table 3 compares the results of PoWareMatch to three baselines, namely *raw* (assuming unbiased matching), ADnEV (representing state-of-the-art algorithmic matching) and *ML* (applying non-process aware learning), in terms of correlation (r and τ) and error (RMSE and MAE), see Section 5.2.1, with respect to the three estimated quantities, $\hat{Pr}\{e_t\}$, $\hat{P}(\sigma_{t-1})$ and $\hat{F}(\sigma_{t-1})$. For *raw* and ADnEV the quantities are calculated by Eq. 13 and for *ML* and PoWareMatch they are predicted using the *HP* component. The best results within each quantity are marked in bold.

Human (and algorithmic) matching is biased and PoWareMatch performs well in calibrating it. Specifically, PoWareMatch improves *raw* human decision confidence correlation with decision correctness by 169% and 146% in terms of r and τ, respectively, and lowers the respective error by 0.36 and 0.39 in terms of *RMSE* and *MAE*, respectively. Algorithmic matching (ADnEV) exhibits similar low correlation, yet the error is fairly low. A possible explanation involves the significant number of non-corresponding element pairs (non-matches). While human matchers avoid assigning confidence values to non-matches (and therefore such element pairs are not included in the error computation), algorithmic matchers assign (very) low similarity scores to non-corresponding element pairs. For example, none of the human matchers assigned the correspondence

[6] Since recall is always an MIEM (Lemma 3.1), a dominating strategy adds all correspondences (first row in Table 2), regardless of the history processing method.

Table 3: Correlation (r and τ) and error (RMSE and MAE) in estimating $\hat{Pr}\{M_{ij} \in \sigma^*\}$, $\hat{P}(\sigma_{t-1})$ and $\hat{F}(\sigma_{t-1})$

Measure	Method	r	τ	RMSE	MAE
$\hat{Pr}\{M_{ij} \in \sigma^*\}$	*raw*	0.29	0.26	0.59	0.45
	ADnEV	0.27	0.22	0.24	0.21
	ML	0.76	0.62	0.30	0.13
	PoWareMatch	**0.78**	**0.64**	**0.23**	**0.06**
$\hat{P}(\sigma_{t-1})$	*raw*	0.23	0.17	0.99	0.50
	ADnEV	-	-	0.19	0.19
	ML	0.00	-0.01	0.34	0.29
	PoWareMatch	**0.90**	**0.72**	**0.13**	**0.10**
$\hat{F}(\sigma_{t-1})$	*raw*	0.21	0.15	0.77	0.39
	ADnEV	-	-	0.37	0.37
	ML	-0.06	-0.04	0.27	0.23
	PoWareMatch	**0.80**	**0.60**	**0.12**	**0.09**

between Contact.e-mail and POBillTo.city, while all algorithmic matchers assigned a similarity score of less than 0.05 to this correspondence. This observation may also serve as an explanation to the proximity between the RMSE and MAE values of ADnEV. Finally, when compared to non-process aware learning (*ML*), PoWareMatch achieves only a slight correlation (r and τ) improvement, yet *ML*'s error values (*RMSE* and *MAE*) are significantly higher. These higher error values demonstrate that process aware-learning, as applied by PoWareMatch, is better in accurately predicting the probability of a decision in the history to be correct ($\hat{Pr}\{e_t\}$) and explains the superiority of PoWareMatch in providing precise results (Table 2).

5.3.3 Ablation Study. After empirically validating that applying a process-aware learning (as in PoWareMatch) is better than assuming unbiased matching (*raw*) and unordered learning (*ML*), we next analyze the suggested representation of matching decisions (Section 4.3.1), namely, 1) **conf**idence ($h_t.c$), 2) **cognitive** aspects (δ_t and a_e), or 3) **alg**orithmic input ($\tilde{M}_e$). Similar to Table 2, Table 4 presents precision (P) and its computational components ($|\sigma \cap \sigma^*|$ and $|\sigma|$, see Eq. 1, with respect to a target measure (without recall[6]). Table 4 compares PoWareMatch to: 1) using each decision representation element by itself (*only*) and 2) removing it one at a time (*w/o*). Boldface entries indicate the higher importance (for *only* higher quality and for *w/o* lower quality).

Examining Table 4, we observe that two aspects of the decision representation are predominant, namely confidence (conf) and cognitive aspects (cognitive). Using confidence features only yields the highest proportion of correct correspondences ($|\sigma \cap \sigma^*|$) and using only cognitive features offers the best precision values. The ablation study shows that when self-reported confidence is absent from the decision representation (only cognitive, only alg, and w/o conf, Table 4), PoWareMatch can provide precise results using other aspects of the decision. For example, using a cognitive representation of decisions (decision times and consensus), the process-aware learning of PoWareMatch (targeting F) achieves a precision of 0.94.

Cognitive and confidence together, without an algorithmic similarity (bottom row of Table 4), achieves comparable results to the ones reported in Table 2, while eliminating either confidence or cognitive features reduces performance. This observation may indicate that the algorithmic matcher is not as important to correspondences that are assigned by human matchers. Recalling that *HP* is designed to consider only the set of correspondences that were originally assigned by the human matcher during the decision history, in the following section we show the importance of algorithmic results in complementing human decisions.

5.4 Improving Matching Outcome

We now examine the overall performance of PoWareMatch and the ability of *RB* to boost recall. The *RB* thresholds (see Section 4.4) for PoWareMatch(*HP*+*RB*) and *raw*-ADnEV were set to the top performing thresholds **during training** (0.9 and 0.85, respectively). Table 5 compares, for each target measure, results of PoWareMatch with and without recall boosting (PoWareMatch(*HP*+*RB*) and PoWareMatch(*HP*), respectively) and *raw*-ADnEV (see Section 5.2.3). In addition, the four last rows of Table 5 exhibit the results of algorithmic matchers, for which we present the threshold yielding the best performance in terms of F. Best results for each quality measure are marked in bold.

Evidently, *RB* improves (mostly in a statistical significance manner) recall and F1 measure over PoWareMatch's *HP*. On average, the recall boosting phase improves recall by 214% and the F1 measure by 125%. Compared to the baselines, PoWareMatch outperforms *raw*-ADnEV by 23%, 8%, and 17% in terms of P, R, and F on average, respectively, and performs better than ADnEV, Term, Token Path, WordNet by 19%, 51%, 41%, and 49% in terms of F, on average.

Compared to the baselines, PoWareMatch outperforms *raw*-ADnEV by 23%, 8%, and 17% in terms of P, R, and F on average, respectively, and performs better than ADnEV, Term, Token Path, WordNet by 19%, 51%, 41%, and 49% in terms of F, on average. We next dive into more detailed analysis, namely skill-based performance (Section 5.4.1) and precision-recall tradeoff (Section 5.4.2).

5.4.1 RB's Effect via Skill-based Analysis. We note again that poor recall is a feature of human matching and while raw human matching oftentimes also suffers from low precision, Section 5.3 shows that PoWareMatch can boost the precision to obtain reliable human matching results. We now analyze the *RB*'s ability to boost recall. Since human matchers vary in their abilities to perform high quality matching, in addition to all human matchers (All), we also investigate high-quality (Top-10) and low-quality matchers (Bottom-10). On average, using Eq. 2, all human matchers yielded matches with P=.55, R=.29, F=.38, the Top-10 group matches with P=.91, R=.74, F=.81, and the Bottom-10 group matches with P=.14, R=.06, F=.08.

Table 6 compares the three groups in terms of history size, *i.e.,* average number of human decisions, match size (and number of true positive) of the *HP* phase, the average number of (correct) correspondences added in the *RB* phase, and the improvement in terms of P, R, and F of the recall boosting using the *RB* component.

RB significantly improves, over all human matchers, recall (211% on average) and f-measure (122% on average) and slightly improves precision. When it comes to low-quality matchers (note that we refrain from screening low quality human matchers) *RB* has a considerable role (bottom row, Table 6) while for high-quality matchers, RB only provides a slight recall boost (middle row, Table 6).

Table 4: Decision representation ablation study. *only* refers to training using only one decision representation element while *w/o* refers to the exclusion of a decision representation element at a time.

Target →		P						F					
(threshold)		1.0			$\hat{P}(\sigma_{t-1})$			0.5			$0.5 \cdot \hat{F}(\sigma_{t-1})$		
↓ Decision Rep.		$\mid \sigma \cap \sigma^* \mid$	$\mid \sigma \mid$	P	$\mid \sigma \cap \sigma^* \mid$	$\mid \sigma \mid$	P	$\mid \sigma \cap \sigma^* \mid$	$\mid \sigma \mid$	P	$\mid \sigma \cap \sigma^* \mid$	$\mid \sigma \mid$	P
only	conf	**18.78**	19.94	0.94	**19.02**	20.53	0.93	**18.72**	20.03	0.93	**17.92**	19.53	0.92
	cognitive	15.29	15.80	**0.96**	15.35	16.53	**0.93**	16.11	16.91	**0.95**	16.27	17.31	**0.94**
	alg	14.15	16.93	0.82	16.86	21.17	0.75	17.24	21.71	0.76	18.35	24.48	0.70
w/o	conf	**13.36**	15.76	0.85	**16.14**	20.43	0.79	**16.39**	20.57	0.797	**13.53**	18.23	0.74
	cognitive	14.10	16.86	**0.81**	16.79	20.89	**0.77**	17.05	21.18	**0.77**	18.18	23.66	**0.72**
	alg	15.46	15.66	0.98	16.42	16.54	0.98	16.58	16.66	0.98	16.68	17.28	0.95

Table 5: Precision (P), recall (R), f-measure (F) of PoWareMatch **by target measure compared to baselines (PO task)**

Target	(threshold)	Method	P	R	F
R	0.0	PoWareMatch(*HP*)	0.549	0.293	0.380
		PoWareMatch(*HP+RB*)	0.776	0.844	0.797
		raw-ADnEV	0.731	0.807	0.756
P	1.0	PoWareMatch(*HP*)	**0.999**	0.230	0.364
		PoWareMatch(*HP+RB*)	**0.999***	0.782	0.876*
		raw-ADnEV	0.824	0.680	0.729
	$\hat{P}(\sigma_{t-1})$	PoWareMatch(*HP*)	0.987	0.254	0.391
		PoWareMatch(*HP+RB*)	0.998*	0.805*	0.889*
		raw-ADnEV	0.808	0.688	0.730
F	0.5	PoWareMatch(*HP*)	0.993	0.256	0.393
		PoWareMatch(*HP+RB*)	0.998*	0.807	0.891*
		raw-ADnEV	0.754	0.794	0.764
	$0.5 \cdot \hat{F}(\sigma_{t-1})$	PoWareMatch(*HP*)	0.968	0.261	0.398
		PoWareMatch(*HP+RB*)	0.993*	0.812	**0.892***
		raw-ADnEV	0.741	0.792	0.754
-		ADnEV [56]	0.810	0.692	0.730
		Term [27]	0.471	0.738	0.575
		Token Path [47]	0.479	**0.862**	0.615
		WordNet [29]	0.453	0.815	0.582

Table 6: History sizes ($|H|$), match sizes ($|\sigma|$), true positive number ($|\sigma \cap \sigma^*|$), and Precision (P), recall (R), and f-measure (F) improvement achieved by *RB* by matchers subgroup

	$\lvert H \rvert$	$\lvert\sigma\rvert$ ($\lvert\sigma \cap \sigma^*\rvert$)		*RB* % Improvement		
↓ Group		*HP*	*RB*	P	R	F
All	43.5	17.3 (17)	36.2 (35.5)	2%	211%	122%
Top-10	54.5	43 (43)	6.9 (6.9)	0%	16%	12%
Bottom-10	26.2	2.7 (2.3)	46.6 (43.2)	4%	3,230%	1,900%

PoWareMatch is judicious even when calibrating the results of the high quality matchers. While on average, 49.8 of the 54.5 raw decisions of high-quality human matchers are correct, PoWareMatch only uses an average of 43 (correct) correspondences when processing history, omitting, on average, 6.8 correct correspondences from the final match (recall that *RB* considers only M^{∂}, see Section 4.4). However, a state-of-the-art algorithmic matcher enables recall boosting, adding an average of 6.9 (other) correct correspondences to the final match, improving both recall and f-measure.

5.4.2 PoWareMatch *Precision - Recall Tradeoff.* Our analysis thus far used a threshold of 0.9, which yielded the best performance during training. We next turn to examine how the tradeoff between precision and recall changes with the threshold. Figure 10 illustrates precision (P), recall (R), and f-measure (F), targeting recall (Figure 10a), precision (with dynamic thresholds,[7] Figure 10b), and f-measure (with dynamic thresholds, Figure 10c). The far right values in each graph represent using *HP* only, allowing no algorithmic results into the output match $\hat{\sigma}$. Values at the far left, setting *RB* threshold to 0, includes all algorithmic results in $\hat{\sigma}$.

Overall, the three graphs demonstrate similar trend. Primarily, regardless of the target measure, a 0.9 threshold yields the best results in terms of f-measure (as was set during training). Recall is at its peak when adding all correspondences human matchers did not assign (threshold = 0). A conservative approach of adding only correspondences the algorithmic matcher is fully confident about (threshold = 1) results in a very low recall.

5.5 PoWareMatch Generalizability

In our final analysis we use the ontology alignment domain to demonstrate the power of PoWareMatch in generalizing beyond schema matching. Table 7 presents results on human matching dataset of OAEI (see Section 5.1) in a similar fashion to Table 5.

Matching results are slightly lower than in the PO task. However, the tendency is the same, demonstrating that a PoWareMatch trained on the domain of schema matching can achieve align ontologies well. The main difference between the performance of PoWareMatch on the PO task and the OAEI task is in terms of precision, where the results of the latter failed to reach the 0.999 precision value of the former (when targeting precision). This may not come as a surprise since the trained model affects only the *HP* component, which is also in charge of providing high precision.

6 RELATED WORK

Human-in-the-loop in schema matching typically uses one of two techniques to reduce the demanding cognitive load of this task, namely crowdsourcing [25, 44, 54, 62, 63] and pay-as-you-go [38, 43, 49]. The former slices the task into smaller sized tasks and spread the load over multiple matchers. The latter partitions the task over time, aiming at minimizing the matching task effort at each point in time. From a usability point-of-view, Noy, Lanbrix, and Falconer investigated ways to assist humans in validating results of computerized matching systems [24, 34, 45]. In this work, we provide an

[7] Static thresholds yielded similar results and can be found in an online repository [2].

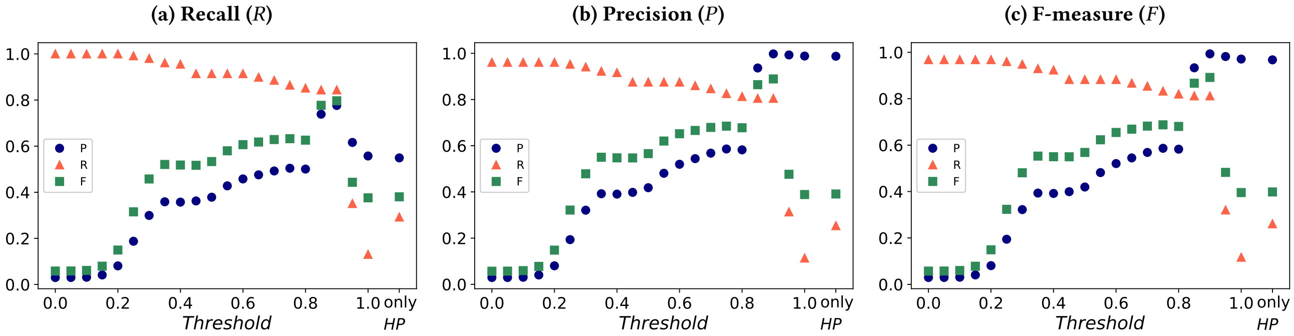


Figure 10: R, P, and F of PoWareMatch **as a function of RB threshold by target measure (P and F use dynamic thresholds).**[7]

Table 7: Precision (P), recall (R), f-measure (F) of PoWareMatch **by target measure compared to baselines (OAEI task)**

Target	(threshold)	Method	P	R	F
R	0.0	PoWareMatch(HP)	0.616	0.348	0.447
		PoWareMatch(HP+RB)	0.554	0.896	0.749*
		raw-ADnEV	0.484	0.861	0.638
P	1.0	PoWareMatch(HP)	0.878	0.315	0.432
		PoWareMatch(HP+RB)	**0.912**	0.746*	0.792*
		raw-ADnEV	0.812	0.582	0.688
	$\hat{P}(\sigma_{t-1})$	PoWareMatch(HP)	0.867	0.318	0.436
		PoWareMatch(HP+RB)	0.889	0.776*	0.810*
		raw-ADnEV	0.785	0.567	0.674
F	0.5	PoWareMatch(HP)	0.824	0.327	0.442
		PoWareMatch(HP+RB)	0.896*	0.747*	0.809*
		raw-ADnEV	0.745	0.582	0.681
	$0.5 \cdot \hat{F}(\sigma_{t-1})$	PoWareMatch(HP)	0.811	0.319	0.437
		PoWareMatch(HP+RB)	0.892*	0.772*	**0.825***
		raw-ADnEV	0.743	0.586	0.682
-		ADnEV [56]	0.677	0.656	0.667
		Term [27]	0.266	0.750	0.393
		Token Path [47]	0.400	0.250	0.307
		WordNet [29]	0.462	**0.937**	0.618

alternative approach, offering an algorithmic solution that is shown to improve on human matching performance. Our approach takes a human matcher's input and boosts its performance by analyzing it as a process and complementing it with an algorithmic matcher.

Also using a crowdsourcing technique, Bozovic and Vassalos proposed a combined human-algorithm matching system where limited user feedback is used to weigh the algorithmic matching [15]. In our work we offer an opposite approach, according to which once a human match is provided, it is evaluated and modified, and then extended with algorithmic solutions.

Human matching performance was analyzed in both schema matching and the related field of ontology alignment [8, 20, 37, 62], acknowledging that humans can err while matching due to biases. Our work turns such bugs into features, improving human matching performance by taking into account possible biases.

The use of deep learning for solving data integration problems becomes widespread [22, 33, 42, 59]. Chen *et al.* [17] use instance data to apply supervised learning for schema matching. Fernandez [26] *et al.* used embeddings to identify relationships between attributes, which was further extended by Cappuzzo *et al.* [16] by considering instances to create local embeddings. Shraga *et al.* [56] use a neural network to improve an algorithmic schema matching result. In our work, we use an LSTM to capture the time-dependent decision making involved in human matching and complement it with a state-of-the-art deep-learning-based algorithmic matching [56].

7 CONCLUSIONS AND FUTURE WORK

This work offers a novel approach to address matching, analyzing it as a process. We define a matching sequential process using matching history (Definition 2.2) and monotonic evaluation of the matching process (Section 3.1). We show conditions under which precision, recall and f-measure are monotonic (Theorem 3.1). Then, we tie the monotonicity of these measures to the ability of a correspondence to improve on a match evaluation and characterize such correspondences in probabilistic terms (Theorem 3.2).

Realizing that human matching is biased (Section 3.3.1) we offered PoWareMatch to calibrate human matching decisions and compensate for correspondences that were left out using algorithmic matching. Our empirical evaluation showed a clear benefit in treating matching as a process and confirmed that PoWareMatch improves on human and algorithmic matching and generalizes well to the closely domain of ontology alignment.

In future work, we aim to extend PoWareMatch to additional platforms, specifically, the one of crowdsourcing, where several additional aspects, such as the heterogeneity of crowd workers [53], need to be considered. Another interesting direction involves experimenting with additional matching tools.

REFERENCES

[1] 2020. Data. https://github.com/shraga89/PoWareMatch/tree/master/DataFiles.
[2] 2020. Graphs. https://github.com/shraga89/PoWareMatch/tree/master/Eval_graphs.
[3] 2020. OAEI benchmark. http://oaei.ontologymatching.org/2011/benchmarks/.
[4] 2020. Ontobuilder research environment. https://github.com/shraga89/Ontobuilder-Research-Environment.
[5] 2020. PoWareMatch repository. https://github.com/shraga89/PoWareMatch.
[6] 2020. PoWareMatch Technical Report. https://github.com/shraga89/PoWareMatch/blob/master/PoWareMatch_Tech.pdf.
[7] 2020. PyTorch. https://pytorch.org/.
[8] Rakefet Ackerman, Avigdor Gal, Tomer Sagi, and Roee Shraga. 2019. A cognitive model of human bias in matching. In *Pacific Rim International Conference on Artificial Intelligence*. Springer, 632–646.

[9] Rakefet. Ackerman and Valerie Thompson. 2017. Meta-Reasoning: Monitoring and control of thinking and reasoning. *Trends in Cognitive Sciences* 21, 8 (2017), 607–617.

[10] Lawrence W Barsalou. 2014. *Cognitive psychology: An overview for cognitive scientists*. Psychology Press.

[11] Zohra Bellahsene, Angela Bonifati, Fabien Duchateau, and Yannis Velegrakis. 2011. On Evaluating Schema Matching and Mapping. In *Schema Matching and Mapping*, Zohra Bellahsene, Angela Bonifati, and Erhard Rahm (Eds.). Springer Berlin Heidelberg, 253–291. http://dx.doi.org/10.1007/978-3-642-16518-4_9

[12] Zohra Bellahsene, Angela Bonifati, and Erhard Rahm (Eds.). 2011. *Schema Matching and Mapping*. Springer. https://doi.org/10.1007/978-3-642-16518-4

[13] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. 2011. Generic Schema Matching, Ten Years Later. *PVLDB* 4, 11 (2011), 695–701. http://www.vldb.org/pvldb/vol4/p695-bernstein_madhavan_rahm.pdf

[14] Robert A Bjork, John Dunlosky, and Nate Kornell. 2013. Self-regulated learning: Beliefs, techniques, and illusions. *Annual Review of Psychology* 64 (2013), 417–444.

[15] Nikolaos Bozovic and Vasilis Vassalos. 2015. Two Phase User Driven Schema Matching. In *Advances in Databases and Information Systems - 19th East European Conference, ADBIS 2015, Poitiers, France, September 8-11, 2015, Proceedings*. 49–62. https://doi.org/10.1007/978-3-319-23135-8_4

[16] Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating embeddings of heterogeneous relational datasets for data integration tasks. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1335–1349.

[17] Chen Chen, Behzad Golshan, Alon Y Halevy, Wang-Chiew Tan, and AnHai Doan. 2018. BigGorilla: An Open-Source Ecosystem for Data Preparation and Integration. *IEEE Data Eng. Bull.* 41, 2 (2018), 10–22.

[18] Hong-Hai Do and Erhard Rahm. 2002. COMA—a system for flexible combination of schema matching approaches. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 610–621.

[19] Xin Dong, Alon Halevy, and Cong Yu. 2009. Data integration with uncertainty. *The VLDB Journal* 18 (2009), 469–500. Issue 2. http://dx.doi.org/10.1007/s00778-008-0119-9

[20] Zlatan Dragisic, Valentina Ivanova, Patrick Lambrix, Daniel Faria, Ernesto Jiménez-Ruiz, and Catia Pesquita. 2016. User validation in ontology alignment. In *International Semantic Web Conference*. Springer, 200–217.

[21] Eduard C Dragut, Mourad Ouzzani, Ahmed K Elmagarmid, Walid G Aref, et al. 2016. ORLF: A flexible framework for online record linkage and fusion. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 1378–1381.

[22] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2018. Distributed Representations of Tuples for Entity Resolution. *PVLDB* 11, 11 (2018).

[23] Jérôme Euzenat, Pavel Shvaiko, et al. 2007. *Ontology matching*. Vol. 18. Springer.

[24] Sean M. Falconer and Margaret-Anne D. Storey. 2007. A Cognitive Support Framework for Ontology Mapping. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*. Lecture Notes in Computer Science, Vol. 4825. Springer Berlin Heidelberg, 114–127. https://doi.org/10.1007/978-3-540-76298-0_9

[25] Ju Fan, Meiyu Lu, Beng Chin Ooi, Wang-Chiew Tan, and Meihui Zhang. 2014. A hybrid machine-crowdsourcing system for matching web tables. In *2014 IEEE 30th International Conference on Data Engineering*. IEEE, 976–987.

[26] Raul Castro Fernandez, Essam Mansour, Abdulhakim A Qahtan, Ahmed Elmagarmid, Ihab Ilyas, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. 2018. Seeping semantics: Linking datasets using word embeddings for data discovery. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 989–1000.

[27] Avigdor Gal. 2011. *Uncertain Schema Matching*. Morgan & Claypool Publishers.

[28] Avigdor Gal, Haggai Roitman, and Roee Shraga. 2019. Learning to Rerank Schema Matches. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* PrePrint https://ieeexplore.ieee.org/document/8944172 (2019). https://doi.org/10.1109/TKDE.2019.2962124

[29] Maciej Gawinecki. 2009. Abbreviation expansion in lexical annotation of schema. *Camogli (Genova), Italy June 25th, 2009 Co-located with SEBD* (2009), 61.

[30] Alon Y Halevy and Jayant Madhavan. 2003. Corpus-based knowledge representation. In *IJCAI*, Vol. 3. 1567–1572.

[31] Joachim Hammer, Michael Stonebraker, and Oguzhan Topsakal. 2005. THALIA: Test Harness for the Assessment of Legacy Information Integration Approaches. In *Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5-8 April 2005, Tokyo, Japan*. 485–486. https://doi.org/10.1109/ICDE.2005.140

[32] Bin He and Kevin Chen-Chuan Chang. 2005. Making holistic schema matching robust: an ensemble approach. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. 429–438.

[33] Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2018. Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 787–798.

[34] Patrick Lambrix and Anna Edberg. 2003. Evaluation of Ontology Merging Tools in Bioinformatics. In *Proceedings of the 8th Pacific Symposium on Biocomputing, PSB 2003, Lihue, Hawaii, USA, January 3-7, 2003*, Vol. 8. 589–600.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[36] Guoliang Li. 2017. Human-in-the-loop data integration. *Proceedings of the VLDB Endowment* 10, 12 (2017), 2006–2017.

[37] Huanyu Li, Zlatan Dragisic, Daniel Faria, Valentina Ivanova, Ernesto Jiménez-Ruiz, Patrick Lambrix, and Catia Pesquita. 2019. User validation in ontology alignment: functional assessment and impact. *The Knowledge Engineering Review* 34 (2019).

[38] Robert McCann, Warren Shen, and AnHai Doan. 2008. Matching schemas in online communities: A web 2.0 approach. In *2008 IEEE 24th international conference on data engineering*. IEEE, 110–119.

[39] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. 2002. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings 18th International Conference on Data Engineering*. IEEE, 117–128.

[40] Janet Metcalfe and Bridgid Finn. 2008. Evidence that judgments of learning are causally related to study choice. *Psychonomic Bulletin & Review* 15, 1 (2008), 174–179.

[41] Giovanni Modica, Avigdor Gal, and Hasan M Jamil. 2001. The use of machine-generated ontologies in dynamic information seeking. In *International Conference on Cooperative Information Systems*. Springer, 433–447.

[42] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data*. ACM, 19–34.

[43] Quoc Viet Hung Nguyen, Thanh Tam Nguyen, Zoltán Miklós, Karl Aberer, Avigdor Gal, and Matthias Weidlich. 2014. Pay-as-you-go reconciliation in schema matching networks. In *2014 IEEE 30th International Conference on Data Engineering*. IEEE, 220–231.

[44] Natalya Fridman Noy, Jonathan Mortensen, Mark A. Musen, and Paul R. Alexander. 2013. Mechanical turk as an ontology engineer?: using microtasks as a component of an ontology-engineering workflow. In *Web Science 2013 (co-located with ECRC), WebSci '13, Paris, France, May 2-4, 2013*, Hugh C. Davis, Harry Halpin, Alex Pentland, Mark Bernstein, and Lada A. Adamic (Eds.). ACM, 262–271. https://doi.org/10.1145/2464464.2464482

[45] Natalya F Noy and Mark A Musen. 2002. Evaluating ontology-mapping tools: Requirements and experience. In *Workshop on Evaluation of Ontology Tools at EKAW*, Vol. 2. p1–14.

[46] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2020. Blocking and Filtering Techniques for Entity Resolution: A Survey. *ACM Computing Surveys (CSUR)* 53, 2 (2020), 1–42.

[47] Eric Peukert, Julian Eberius, and Erhard Rahm. 2011. AMC-A framework for modelling and comparing matching systems as matching processes. In *2011 IEEE 27th International Conference on Data Engineering*. IEEE, 1304–1307.

[48] Christoph Pinkel, Carsten Binnig, Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Wolfgang May, Andriy Nikolov, Ana Sasa Bastinos, Martin G Skjæveland, Alessandro Solimando, Mohsen Taheriyan, et al. 2018. RODI: Benchmarking relational-to-ontology mapping generation quality. *Semantic Web* 9, 1 (2018), 25–52.

[49] Christoph Pinkel, Carsten Binnig, Evgeny Kharlamov, and Peter Haase. 2013. IncMap: pay as you go matching of relational schemata to OWL ontologies.. In *OM*. Citeseer, 37–48.

[50] Erhard Rahm and Philip A Bernstein. 2001. A survey of approaches to automatic schema matching. *the VLDB Journal* 10, 4 (2001), 334–350.

[51] L Ratinov and Ehud Gudes. 2004. Abbreviation expansion in schema matching and web integration. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society, 485–489.

[52] Patricia Rodríguez-Gianolli and John Mylopoulos. 2001. A semantic approach to XML-based data integration. In *International Conference on Conceptual Modeling*. Springer, 117–132.

[53] Joel Ross, Lilly Irani, M Silberman, Andrew Zaldivar, and Bill Tomlinson. 2010. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2863–2872.

[54] C. Sarasua, E. Simperl, and N. F Noy. 2012. Crowdmap: Crowdsourcing ontology alignment with microtasks. In *ISWC*.

[55] Roee Shraga, Avigdor Gal, and Haggai Roitman. 2018. What Type of a Matcher Are You?: Coordination of Human and Algorithmic Matchers. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics, HILDA@SIGMOD 2018, Houston, TX, USA, June 10, 2018*. 12:1–12:7. https://doi.org/10.1145/3209900.3209905

[56] Roee Shraga, Avigdor Gal, and Haggai Roitman. 2020. ADnEV: Cross-Domain Schema Matching using Deep Similarity Matrix Adjustment and Evaluation. *Proceedings of the VLDB Endowment* 13, 9 (2020), 1401–1415.

[57] Rohit Singh, Venkata Vamsikrishna Meduri, Ahmed Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang. 2017. Synthesizing entity matching rules by examples. *Proceedings of the VLDB Endowment* 11, 2 (2017), 189–202.

[58] Radu Stoica, George HL Fletcher, and Juan F Sequeda. 2019. On Directly Mapping Relational Databases to Property Graphs.. In *AMW*.

[59] Saravanan Thirumuruganathan, Nan Tang, Mourad Ouzzani, and AnHai Doan. 2020. Data Curation with Deep Learning. In *EDBT*. 277–286.

[60] Pei Wang, Yongjun He, Ryan Shea, Jiannan Wang, and Eugene Wu. 2018. Deeper: A data enrichment system powered by deep web. In *Proceedings of the 2018 International Conference on Management of Data*. 1801–1804.

[61] Pei Wang, Ryan Shea, Jiannan Wang, and Eugene Wu. 2019. Progressive Deep Web Crawling Through Keyword Queries For Data Enrichment. In *Proceedings of the 2019 International Conference on Management of Data*. 229–246.

[62] Chen Zhang, Lei Chen, HV Jagadish, Mengchen Zhang, and Yongxin Tong. 2018. Reducing Uncertainty of Schema Matching via Crowdsourcing with Accuracy Rates. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[63] Chen Jason Zhang, Lei Chen, H. V. Jagadish, and Caleb Chen Cao. 2013. Reducing Uncertainty of Schema Matching via Crowdsourcing. *PVLDB* 6, 9 (2013), 757–768. http://www.vldb.org/pvldb/vol6/p757-zhang.pdf

[64] Yi Zhang and Zachary G Ives. 2020. Finding Related Tables in Data Lakes for Interactive Data Science. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1951–1966.