Data & Knowledge Engineering 69 (2010) 197–210

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Frameworks for entity matching: A comparison

Hanna Köpcke *, Erhard Rahm

Database Group, University of Leipzig, Postfach 100920, 04009 Leipzig, Germany

ARTICLE INFO

Article history:
Received 3 December 2008
Received in revised form 2 October 2009
Accepted 3 October 2009
Available online 14 October 2009

Keywords:
Entity resolution
Entity matching
Matcher combination
Match optimization
Training selection

ABSTRACT

Entity matching is a crucial and difficult task for data integration. Entity matching frameworks provide several methods and their combination to effectively solve different match tasks. In this paper, we comparatively analyze 11 proposed frameworks for entity matching. Our study considers both frameworks which do or do not utilize training data to semi-automatically find an entity matching strategy to solve a given match task. Moreover, we consider support for blocking and the combination of different match algorithms. We further study how the different frameworks have been evaluated. The study aims at exploring the current state of the art in research prototypes of entity matching frameworks and their evaluations. The proposed criteria should be helpful to identify promising framework approaches and enable categorizing and comparatively assessing additional entity matching frameworks and their evaluations.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Entity matching (also referred to as duplicate identification, record linkage, entity resolution or reference reconciliation) is a crucial task for data integration and data cleaning [19,33,47]. It is the task of identifying entities (objects, data instances) referring to the same real-world entity. Entities to be resolved may reside in distributed, typically heterogeneous data sources or in a single data source, e.g., in a database or a search engine store. They may be physically materialized or dynamically be requested from sources, e.g., by database queries or keyword searches.

Entity matching is a challenging task particularly for entities that are highly heterogeneous and of limited data quality, e.g., regarding completeness and consistency of their descriptions. Table 1 illustrates some of the problems for a bibliographic example of three duplicate entries for the same paper. It is assumed that the bibliographic entities have automatically been extracted from fulltext documents and may thus contain numerous quality problems such as misspelled author names, different ordering of authors, and heterogeneous venue denominations.

The entity matching problem was originally defined in 1959 by Newcombe et al. [45] and was formalized by Felligi and Sunter [26] 10 years later. Since then it has been considered under various facets and from different communities, including the AI research community, the DB research community, and industry. Numerous approaches have been proposed for entity matching especially for structured data. For overview surveys and two tutorials see [60,28,3,25,35,31], respectively. Due to the large variety of data sources and entities to match there is no single "best" match algorithm. A single match approach typically performs very differently for different domains and match problems. For example, it has been shown that there is no universally best string similarity measure [29,50]. Instead it is often beneficial and necessary to combine several methods for improved matching quality, e.g., to consider the similarity of several attributes or to take into account relationships between entities. For large datasets, it is also necessary to apply so-called blocking strategies to reduce the search space for

* Corresponding author.
E-mail addresses: koepcke@informatik.uni-leipzig.de (H. Köpcke), rahm@informatik.uni-leipzig.de (E. Rahm).

0169-023X/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2009.10.003

Table 1
Multiple references to the same paper object.

Title	Author	Venue	Year
The merge/purge problem for large databases	M.A. Hernandez, S.J. Stolfo	Proceedings of the ACM SIGMOD international conference	
The Merge/Purge problem for Large Databases	A.H. Mauricio, J.S. Stolfo	Proc. of the 1995 ACM SIGMOD conference on management	1995
andez, SJ Stolfo The merge/purge problem for Large Databases	M. Hern	Proceedings of the 1995 ACM SIGMOD conference on management	1995

entity matching and achieve sufficiently fast execution times. Hence, several entity matching frameworks have recently been developed which support multiple approaches for blocking and matching as well as their combination. A key problem to be addressed either manually or with the help of machine learning techniques is how to find an effective and efficient combination of different techniques.

To investigate the state of the art in this respect we comparatively analyze 11 proposed entity matching frameworks. We focus on research prototypes but do not consider the more general system approaches on data cleaning and data integration, such as AJAX [27] and Potters's Wheel [48]. We also exclude commercial systems such as ChoiceMaker, DataCleanser (EDD), Merge/Purge Library (Sagent/QM Software) or MasterMerge (Pitney Bowes) from our discussion since they are not widely available and their algorithms are not described in the public literature.

In contrast to the previously published surveys [60,28,3,25] we do not focus on specific algorithms but study which spectrum of approaches is provided by different frameworks and how the approaches can be combined. Our study considers both frameworks which do or do not utilize training data to find an effective configuration to solve a given entity matching task. We further study how the different frameworks have been quantitatively evaluated. The study aims at providing an overview of the current state of the art regarding research entity matching frameworks and their evaluations. Our comparison criteria should enable to categorize and compare further frameworks and methods for entity matching as well as their evaluations. Furthermore, they should be helpful in identifying promising frameworks as well as areas where more work is needed.

The remaining part of this paper is organized as follows: in Section 2 we briefly introduce the entity matching problem and specify high-level requirements for an entity matching framework. Section 3 compares the functionality of 11 selected entity matching frameworks based on a common set of criteria. In Section 4 we then compare how the considered frameworks have been evaluated to assess their effectiveness and efficiency. Section 5 concludes the paper.

2. Requirements for entity matching frameworks

The considered frameworks have to solve the entity matching problem which can be stated as follows:

Definition 1 (*Entity matching problem*). Given two sets of entities $A \in S_A$ and $B \in S_B$ of a particular semantic entity type from data sources S_A and S_B, the entity matching (EM) problem is to identify all correspondences between entities in $A \times B$ representing the same real-world object. The definition includes the special case of finding pairs of equivalent entities within a single source ($A = B, S_A = S_B$). The match result is typically represented either by a set of correspondences, sometimes called a mapping, or by a set of clusters. A correspondence $c = (e_i, e_j, s)$ interrelates two entities e_i and e_j from sources S_A and S_B. An optional similarity value $s \in [0, 1]$ indicates the similarity or strength of the correspondence between the two objects. In the alternate result representation, a cluster contains entities that are deemed to represent the same real-world object. Ideally all entities in a cluster refer to the same object, and no two entities from two different clusters refer to the same object.

There are several high-level requirements and desiderata that should be met as far as possible by a suitable entity matching solution and thus by entity matching frameworks.

Effectiveness: The main goal of entity matching is to achieve a high-quality match result with respect to recall and precision, i.e., all real corresponding entities but no others should be included in the result. Achieving this goal for different match tasks typically requires the flexible combination and customization of different match methods. A key concern will thus be which match approaches are supported and how they can be combined.

Efficiency: Entity matching should be fast even for voluminous datasets. For very large datasets this typically prescribes the use of blocking methods to reduce the search space for entity matching (see next section).

Genericity, offline/online matching: The supported entity matching methods should be applicable to different match tasks from various domains (e.g., enterprise data, life science data) and for different data models (e.g., relational, XML). Furthermore, an EM framework should be applicable to offline and online match tasks. Online match tasks arise for interactive data integration steps such as mediated queries or data mashups based on specific user input. Offline entity matching is less time-critical than online matching which can thus better deal with large datasets and may allow for more match algorithms to be applied. Entity matching during the ETL (extract, transform, load) process of data warehouses is a sample case for offline matching.

Low manual effort/self-tuning: The manual effort to employ an EM framework should be as low as possible, in particular for selecting the methods for blocking and matching, their parameters and their combination. Ideally, the framework is able to solve these tasks automatically in a self-tuning manner, e.g., with the help of machine learning methods utilizing training data. On the other hand, selecting and labeling training data may also incur manual effort which should therefore be low.

A key challenge in developing a successful EM framework is that some of the posed requirements are in conflict with each other, e.g., effectiveness and efficiency or genericity and ease-of-use. For example the use of blocking methods improves efficiency by reducing the search space. However, this may eliminate some relevant entity pairs from consideration and thus reduce effectiveness (recall) of entity resolution. On the other hand, the combined use of several match algorithms may improve effectiveness but will typically lead to increased computational overhead and thus reduce efficiency. Successfully resolving entities in diverse domains with the help of a generic entity matching framework is more difficult than for only one domain. Non-generic frameworks may thus incur a reduced manual effort to provide training or to find a suitable combination and customization of algorithms.

3. Functional comparison of EM frameworks

We compare 11 proposed EM frameworks (Table 2). Our selection tries to cover a broad spectrum of different approaches. We have selected three frameworks that need no training and eight training-based frameworks. We focused on those frameworks which allow the combined use of several match algorithms and for which evaluation results have been published. Clearly, our comparison can only cover a current snapshot of existing systems and not all relevant frameworks can be included for space reasons. However, we believe that our methodology to compare different frameworks can be used to evaluate further systems or to update the characteristics when frameworks improve or new evaluations become known. In this section we focus on the functionality of the different frameworks. In Section 4, we analyze published evaluation results for the frameworks. Before we start the comparative discussion for the frameworks we first provide more information of some key comparison criteria influencing the high-level requirements of effectiveness, efficiency and self-tuning: blocking methods, matchers, combination of matchers, and training selection.

3.1. Comparison criteria

3.1.1. Entity type

Methods for relational entity matching assume that each tuple represents an entity and all attribute values describe that entity. Sufficiently similar data values of two tuples imply that they are duplicates. Complex structured and XML data is semi-structured and is organized hierarchically. This complicates entity matching, compared to relational data that is flat and usually well-structured. It is not clear whether a child element represents part of the description of an element (as does a relational attribute), or if it represents a related object (as does a relationship with another table). Furthermore, elements describing the same kind of entity are not necessarily equally structured. These structural differences are due to either different representations of same entities (e.g., persons may be represented as managers or employees), or differences allowed by the schema, e.g., multiplicities of elements (e.g., persons having no, one, or multiple phone numbers).

3.1.2. Blocking methods

Blocking is needed for large inputs to reduce the search space for entity matching from the Cartesian product to a small subset of the most likely matching entity pairs. Numerous blocking algorithms have been proposed in the past years (see [4] for an overview). These techniques typically use a key to partition the entities to be matched into groups (blocks). Matching of an entity can then be restricted to the entities in the same block. The key is typically composed from parts (e.g., first letters) of entity attribute values.

As a criterion for comparing frameworks we distinguish between *disjoint* and *overlapping* blocking methods. *Disjoint* methods build mutually exclusive blocks, i.e., each entity is assigned to one block. Implementations may use sorting or hashing on the key. *Overlapping* methods may result in overlapping blocks of entities; implementations include the (multi-pass) sorted neighborhood approach [33], bi-gram indexing [4], canopy clustering [40] and iterative blocking [59]. These methods can require an entity to be matched against multiple blocks (increased overhead) but may lead to a better recall than disjoint methods. For a recent comparison of several blocking methods see [23].

The definition of the key is a critical issue with all blocking methods. A suboptimal choice may lead to over-selection of many dissimilar entity pairs that impedes efficiency, or, worse, sorting out true matching entity pairs thus decreasing match quality. The key may have to be determined manually or (semi-)automatically based on training data [10,42].

3.1.3. Matchers

Entity matching requires a way to determine whether two entities are alike enough to represent the same real-world entity. A matcher is an algorithm specifying how the similarity between two entities is computed. We distinguish two types of matchers: attribute value matchers and context matchers.

Attribute value matchers use a similarity function and apply it on the values of a pair of corresponding attributes or attribute concatenations of the input datasets. They typically return a value between 0 and 1 indicating the degree of similarity

Table 2
Overview of entity matching frameworks (– means not present, ? means not clear from publication).

	BN [38]	MOMA [55]	SERF [5]	Active Atlas [53,54]	MARLIN [11,12]	Multiple Classifier System [62]	Operator Trees [13]	TAILOR [24]	FEBRL [18,17]	STEM [36]	Context Based Framework [16]
Training-based				Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Entity type	XML	Relational	Relational	Relational	Relational	Relational	Relational	Relational	Relational	Relational	Relational
Blocking	–	–	–								
Key definition				Manual	Manual	Manual	Manual	Manual	Manual	Manual	Manual
Partitioning											
Disjoint						?		Sorting, hashing	Sorting	Threshold	
Overlapping				Hashing	Canopy clustering	?	Canopy clustering	Sorted neighborhood	Sorted neighborhood, q-gram, canopy clustering	Sorted neighborhood	Canopy-like
Matchers	Attribute value, context	Attribute value, context	Attribute value	Attribute value	Attribute value		Attribute value	Attribute value	Attribute value	Attribute value	Attribute value, context
Matcher combination	Numerical	Workflow	Rules	Rules	Numerical, rules	Numerical, rules	Rules	Numerical, rules	Numerical	Numerical, rules	Numerical, rules
Learners				Decision tree	SVM, decision tree	7 (SVM, decision tree, etc.)	Operator Tree algorithm, SVM	Probabilistic, decision tree	SVM	SVM, decision tree, logistic regression, multiple learning	Diverse (SMOreg, Logistic, etc.)
Training selection				Manual, semi-automatic	Manual, semi-automatic	Manual	Manual	Manual	Manual, automatic	Manual, semi-automatic	Manual

between two entities. The previously proposed approaches mostly assume that corresponding attributes from the input datasets have been determined beforehand, either manually or with the help of schema matching. The corresponding attributes and similarity functions used for matching may have to be selected manually or they could be determined (semi-)automatically, similarly to the key selection for blocking. Numerous similarity functions may be employed, in particular generic string similarity measures (see [20,25] for a comprehensive comparison). Similarity computation may also utilize different kinds of auxiliary information, such as dictionaries, thesauri or domain-specific lookup tables [14], e.g., to deal with synonyms, homonyms, abbreviations, acronyms, or geographic name variations. In offline data integration, e.g., for data warehousing, such auxiliary sources are often used by separate data cleaning steps to resolve representational differences before entity matching begins.

Context matchers consider the context or semantic relationships of different entities for similarity computation. Context matchers commonly represent contextual information (e.g., semantic relationships, hierarchies) in a graph structure, see for example [16,34,7,6,15,21,22,51]. The graph structure allows the propagation of similarity information (e.g., represented as edge weights or auxiliary nodes) to related entities. For example to disambiguate several persons with the same name one may additionally consider contextual information such as their affiliations or co-authors.

3.1.4. Combination of matchers

There are many possibilities for combining multiple matchers within a match strategy. In general, the combination may be expressed by a decision function which applies matchers from a given set of matchers to determine for each pair of entities whether or not the entities match.

We distinguish between three kinds of combination approaches: numerical, rule-based and workflow-based combinations. *Numerical approaches* combine the similarity values of entity pairs (e_i, e_j) determined by different matchers m_1, m_2, etc. by a numerical combination function $f : f(m_1(e_i, e_j), m_2(e_i, e_j), \ldots)$, e.g., by taking a weighted sum or weighted average of similarity values. The resulting numerical value is then mapped to a match or non-match decision. Examples for numerical combination approaches can be found in [26,41,52,32].

Rule-based approaches derive the match decision by a logical combination (or predicate) of match conditions. A match condition can be a threshold condition defined on the similarity value computed by a single matcher $m : C = m(e_i, e_j) > t$, or by several matchers (e.g., using a numerical combination function f) or express a hard constraint (e.g., every paper has a single publisher). A *simple match rule* R consists of the logical conjunction of n match conditions: $R = \bigwedge_{i=1}^{n} C_i$. For example, two books may be considered to match if the string similarities of their title and author attributes both exceed certain thresholds. Such simple match rules consisting only of threshold conditions for single attribute value matchers have also been called *similarity joins*; their restricted structure permits an efficient execution in many cases [1]. *Complex match rules* allow the combination of multiple simple match rules, e.g., by disjunction $\bigvee_{i=1}^{n} R_i$. Many approaches, e.g. [33,37,2], require a human expert to specify such match rules declaratively.

Workflow-based combination of matchers such as supported by MOMA [55] allows almost arbitrary complex combinations of matchers, e.g., to apply a sequence of matchers to iteratively refine a match result or to combine the results of independently executed matchers.

Key decisions to be made for the specification of a combination strategy expressed as a numerical combination function, match rules, or a match workflow include selecting and configuring the matchers to be used. The chosen configuration can have a large impact on the overall quality but even experts will find it difficult and time-consuming to determine a good selection. The use of supervised (training-based) approaches or learners aims at automating the process of entity matching to reduce the required manual effort. Training-based approaches, e.g., Naïve Bayes [49], logistic regression [46], Support Vector Machine (SVM) [11,43,49] or decision trees [63,29,49,53,54,56] have so far been used for some subtasks, e.g., determining suitable parameterizations for matchers or adjusting combination functions parameters (weights for matchers, offsets). However, training-based approaches require suitable training data and providing such data typically involves manual effort. Furthermore, some decisions (e.g., selection of the similarity functions and attributes to be evaluated) may still have to be determined manually. Hence it is important to analyze for training-based approaches which tasks still require manual decisions. In the framework comparison we consider whether a framework supports a numerical combination approach, match rules, or match workflows. We further assess whether and for which tasks training is utilized.

3.1.5. Training selection

Training-based EM frameworks promise a reduced manual effort to find suitable match strategies but their effectiveness critically depends on the size and quality of the available training data. For entity matching it is important that the training data is representative for the objects to be matched and exhibit the variety and distribution of errors observed in practice. We thus investigate how training data is selected in the proposed frameworks. Selection of training example may be manual, semi-automatic or automatic. With manual selection entities have to be chosen and labeled by a user. Semi-automatic selection still requires a human for labeling, but entity pairs are automatically chosen for labeling. Automatic selection provides and labels training examples automatically without any inspection by a user.

3.2. Comparison

Table 2 compares the 11 selected frameworks for entity matching within three groups. The first group includes (three) frameworks that do not utilize training data, while the (four) frameworks of the second group depend on training data. The third group includes hybrid approaches which support supervised matcher combinations as well the manual specification of EM strategies without training. For each group, the frameworks are listed in chronological order of their year of publication. All inspected frameworks support only one type of entities either relational (10 frameworks) or XML (one frameworks), i.e., no framework is able to handle both types. All frameworks focus on offline matching, i.e., they do not yet cover online matching. Online matching has been addressed to some extent in [8,9]. Three of the 11 frameworks operate without training; four of the eight training-based frameworks, the Context Based Framework [16], FEBRL [18,17], STEM [36] and TAILOR [24], also support the manual specification of EM strategies without training (hybrid frameworks). Four of the training-based frameworks expect the users to provide suitable training data manually, while Active Atlas [53,54], MARLIN [11,12], STEM [36] and FEBRL [18,17] have also investigated (semi-)automatic training methods.

Most training-based frameworks – but none of the others – provide explicit support for blocking based on disjoint or/and overlapping entity partitioning. This seems to be influenced by the fact that the trained methods to determine matches, such as SVM or decision trees, are computationally expensive so that blocking is mandatory to reduce the search space even for small-sized match tasks. Canopy-like clustering and sorted neighborhood are the most common blocking techniques. The definition of the blocking key is not yet derived from training data but has to be specified manually in all training-based frameworks. This is a serious limitation for the optimization potential of training-based methods.

All frameworks support attribute value matchers utilizing a large variety of string similarity functions. (Table 3 indicates which similarity functions have been used in the evaluations). Two frameworks additionally provide context matching but only one of the training-based frameworks. All frameworks support the combination of multiple matchers. Most frameworks support (complex) match rules, numerical combination functions, or both approaches. Only one framework, MOMA [55], allows the definition of match workflows.

Seven of the eight training-based frameworks (Active Atlas [53,54], Context Based Framework [16], MARLIN [11,12], Multiple Classifier System [62], Operator Trees [13] and TAILOR [24]) utilize training for learning match rules, mostly by employing decision tree algorithms. Training data is utilized to automatically determine the order in which different matchers are applied as well as threshold conditions on the similarity values computed by the matchers. Another approach pursued by seven training-based frameworks (Context Based Framework [16], MARLIN [11,12], Multiple Classifier System [62], Operator Trees [13], FEBRL [18,17], STEM [36] and TAILOR [24]) is to utilize training for automatically determining a numerical combination function *f*. The combination function is determined through the choice of the employed supervised learning algorithm. The most often employed learner for this task is the SVM. For the SVM the combination function *f* is a weighted sum of matcher similarity values s_i of the form $f = sgn\left(\sum_{i=1}^{n} w_i s_i + b\right)$ and training data is utilized to find suitable weights w_i and the offset *b*. Besides the SVM, the Context Based Framework [16], the Multiple Classifier System [62] and FEBRL [18,17] frameworks consider other supervised learners (e.g., logistic regression) for determining a combination function. All training-based frameworks optimize the combination of a manually predetermined set of matchers, i.e., they do not explicitly select which attributes or similarity functions should be used.

In the following three sections we discuss specific features of the 11 frameworks. We start with the three approaches that do not require training. The training-based frameworks are highlighted in Section 3.2.2 and hybrid frameworks supporting both combination strategies are considered in Section 3.2.3.

3.2.1. Frameworks without training

BN (Bayesian Network): Leitão et al. [38] propose a framework for matching XML entities based on a Bayesian network (BN) model. Bayesian networks provide a graph-based formalism to explicitly represent the dependencies among the entities of a domain. This model is derived from the structure of the XML entities to be matched. The approach numerically combines the similarity for direct attributes value of two XML entities as well as the similarity for descendant XML entities. The user has to specify a match probability threshold above which entities are considered matches.

MOMA (Mapping-based Object Matching): MOMA [55] is a domain-independent framework for entity matching providing an extensible library of matchers, both attribute value and context matchers. To solve a particular match problem MOMA allows the specification of a workflow of several matchers and combination operators. Each matcher and workflow step determines a so-called same-mapping (set of corresponding entity pairs) that can be refined by additional matchers and steps. The final mapping determined by a match workflow is stored in a mapping repository and can be re-used in other workflows. A so-called neighborhood matcher implements a context-based match approach and utilizes semantic relationships between different entities, such as publications of authors or publications of a conference. MOMA supports compose and merge operators to combine different mappings for the same match problem. Different combination functions (avg, min, max, weighted and prefer) can be used to derive a combined similarity values for input correspondences to be combined into a merged correspondence. MOMA does not explicitly offer blocking methods. However, a blocking-like reduction of the search space could be implemented by a liberal attribute matching within a first workflow step whose result is then refined by further match steps.

SERF (Stanford Entity Resolution Framework): The SERF project [5] develops a generic EM infrastructure with the focus on improving the efficiency of entity matching. The authors do not study the internal details of matchers (similarity functions)

Table 3
Overview of framework evaluations.

	BN [38]	MOMA [55]	SERF [5]	Active Atlas [53,54]	MARLIN [11,12]	Multiple Classifier System [62]	Operator Trees [13]	TAILOR [24]	FEBRL [18,17]	STEM [36]	Context Based Framework [16]
Type of test problems	Real, artificial	Real	Real	Real	Real	Real	Real	Real, artificial	Real	Real	Real, artificial
# Domains/# Sources/# Tasks	2/3/3	1/3/6	2/1/2	3/6/3	2/4/6	1/1/1	5/5/5	2/2/2	4/4/7	3/5/4	2/2/2
Semantic entity types	CDs, movies	Publications, authors, venues	Products, hotels	Restaurants, companies, weather data	Restaurants, publications	Persons	Companies, persons, publications, restaurants, birds	Products, persons	Persons, restaurants, publications	Publications, restaurants, parks	Publications, authors, departments, organizations, persons
Min/Max # Entities	1000/10,000	258/66,879	5000/14,574	862/12,428	295/1295	?	?	100,000	841/10,000	654/66,879	1085/14,590
Min/Max # Attributes		1/3	3/8	2/3	1/6	18	1/8	7	?/6	1/4	?/?
Used matchers	Edit distance, match probability of descendants	Trigram, neighborhood	(Exact) equality, Jaro–Winkler, numeric difference	TF–IDF variant	Edit distance, TF–IDF	(Exact) equality, soundex, sub-string, edit distance	(Exact) equality, jaccard, (generalized) edit distance	?	Jaro–Winkler, numeric difference	Cosine, EditDistance, Jaccard, Jaro–Winkler, Monge-Elkan, TF/IDF, Trigram	eTFIDF, connection strength
# Training examples				Varying (up to 700)	10/−20/−40/−60/−1000	66%	1000/−5000/−100,000	Varying (1–100%)		Varying (20–10,000)	?
Blocking performance measures								*RR*, *PC*, F_S	*RR*, *PC*, F_S		
Effectiveness measures (achieved max values)	*P*/*R* (1.0/0.99)	*P*/*R*/*F* (?/?/0.988)	*P* (1.0)	*A* (1.0)	*P*/*R*/*F* (?/?/0.966)	*A* (0.998)	*P*/*R* (0.98/1.0)	*A*/*R* (1.0/0.9)	*F* (0.95)	*F* (0.97)	*F* (0.89)
Efficiency measures (achieved values)			ET (<1–15 h), # feature value comparisons			*TT* (0.86–6050.18 s), ET (0.21–3197.25 s)	TT (<1–1000 s), ET (61 s to 10 d)				TT (0.11–53.66 s), ET (1–5 s)

RR, Reduction ratio; *P*, Precision; ET, Execution time; *PC*, Pairs completeness; *R*, Recall; TT, Training time; *Fs*, *F*-score; *F*, *F*-measure; *A*, Accuracy.

but view them as "black boxes" to be invoked by the EM engine. Different algorithms are provided to minimize the number of invocations to these potentially expensive black boxes by keeping track of previously compared values thus avoiding redundant comparisons. Multiple matchers can be combined by a disjunction of manually defined simple match rules.

3.2.2. Training-based frameworks

Active Atlas: The Active Atlas system proposed by Tejada et al. [53,54] allows the training-based determination of match rules by utilizing a combination of several decision tree learners. Training selection is semi-automatic to minimize the number of required training examples. This is achieved by letting the decision tree learners vote on the most informative example for user to classify next ("active learning"). Attribute value matchers use a variant of TF–IDF that allows considering a variety of additional information (e.g., stemming, abbreviations) to determine the common tokens of two attribute values. A disjoint blocking strategy based on hashing is supported.

MARLIN (Multiply Adaptive Record Linkage with Induction): MARLIN [11,12] employs a training-based approach for combining multiple matchers using the SVM at two levels. At the first level attribute value matchers are tuned. At the second level the SVM is applied to determine a combination of the tuned matchers from the previous step. MARLIN utilizes the canopies clustering method using Jaccard similarity for overlapping blocking. Two methods for semi-automatic training selection are supported: static-active and weakly-labeled negative selection. Static-active selection compares the entities to be resolved with some string similarity measure and selects only pairs that are fairly similar according to this measure to find near-duplicate entity pairs for training. For "legacy" datasets with few duplicate entries the weakly-labeled negative training selection is proposed. It randomly selects entity pairs with few shared tokens for training; these pairs are thus likely non-duplicates.

Multiple Classifier System: In Ref. [62] a Multiple Classifier System approach is proposed that employs a variety of supervised learners for combining matchers, including decision trees, 1-rule, Naïve Bayes, linear and logistic regression, back propagation neural network, and k-nearest neighbors. Furthermore, meta-combination approaches for combining several supervised learners are supported, namely cascading, bagging, boosting, and stacking. While bagging and boosting combine multiple supervised learners of the same type, cascading and stacking are used to combine supervised matchers of different types (e.g., logistic regression and decision tree learning). Blocking support is not explicitly mentioned, but the evaluation suggests that some blocking method has been applied.

Operator Trees: Chaudhuri et al. [13] specify EM strategies by operator trees which correspond to the union (disjunction) of multiple similarity joins. Manually labeled training samples are used to construct the Operator Trees by a recursive divide and conquer strategy. The maximum number of similarity joins in an Operator Tree and the maximum number of similarity function predicates per similarity join can be restricted by the user. Blocking is not explicitly supported. However, the authors state that the canopies clustering method using Jaccard similarity is applied in the evaluation. For comparison a numeric matcher combination utilizing the SVM is considered.

3.2.3. Hybrid frameworks

TAILOR: TAILOR [24] is a toolbox for record linkage supporting numerical and rule-based combination approaches for multiple matchers without training as well as training-based. Five similarity functions are provided for attribute value matching, namely hamming distance, edit distance, Jaro's algorithm, q-grams and soundex. For training-based combination the user can choose among three probabilistic approaches for numerical combination and two rule-based approaches utilizing decision tree learning. Training data must be manually provided by the user. Disjoint as well as overlapping blocking methods are supported.

FEBRL (Freely Extensible Biomedical Record Linkage): FEBRL [18,17] is a hybrid framework supporting a training-based numerical combination approach utilizing the SVM as well as numerical approaches without training. FEBRL is the only one of the considered frameworks that is freely available on the web under an open source software license. It was originally developed for entity matching in the biomedical domain (hence the name). A large selection of 26 different similarity measures is available for attribute value matching. FEBRL supports one disjoint as well as three overlapping blocking methods. Besides manual training selection two strategies for automatic training are supported [17]: threshold- and nearest-based. Both methods select entity pairs automatically and do not require manual labeling by a user. To determine matching/non-matching training examples the threshold method selects entity pairs whose similarity values are within a certain distance to exact similarity or total dissimilarity for all considered matchers. The nearest method sorts the similarity vectors of the entity pairs according to their distances from the vectors containing only exact similarities and only total dissimilarities, respectively, and then selects the nearest entity pairs for training.

STEM (*Self-Tuning Entity Matching*): STEM [36] is a hybrid framework supporting the automatic construction of entity matching strategies. The framework addresses the problem of how to automatically configure and combine several matchers using numerical combination approaches. Several training-based numerical combination approaches are supported utilizing the SVM, decision trees, logistic regression and their combination. Different similarity measures are available for attribute value matching. Besides manual training selection two strategies for automatic training are supported: Threshold-Equal and Threshold-Random. With Threshold-Random entity pairs are randomly selected among the ones satisfying a given minimal threshold t applying a similarity measure m, whereas Threshold-Equal selects an equal number $(n/2)$ of matching and non-matching pairs from the ones satisfying a given minimal threshold t applying a similarity measure m.

Context Based Framework: The Context Based Framework [16] is a graph-based hybrid framework supporting a two stage training-based numerical combination approach. The first stage tunes attribute value and context matchers using mainly SMOreg, a support vector regression approach. At the second level a second learner, e.g., logistic regression is applied to determine a combination of the tuned matchers from the previous step. The Context Based Framework utilizes a canopy-like technique for blocking. Methods for training selection are not supported. Manually labeled training samples have to be provided for both stages of the combination approach.

4. Evaluation comparison

In this section we compare the published evaluations of the considered frameworks. Table 3 summarizes the evaluations.

4.1. Comparison criteria

To compare the evaluations of entity matching frameworks we consider the following criteria:

- *Type of test problems:* Test problems may involve real-world data sources or may be artificially generated.
- *# Domains/# Sources/# Tasks:* How many domains, sources and match tasks are considered?
- *Semantic entity types:* What kinds of match problems have been solved?
- *Min/Max # Entities:* What is the minimum/maximum number of entities involved in a match task?
- *Min/Max # Attributes:* What is the minimum/maximum number of attributes used for solving a match tasks?
- *Used Matchers:* Which matchers (similarity functions) have been used?
- *# Training examples:* How many examples were used for training?
- *Blocking performance measures:* For the evaluation of blocking techniques three measures have been proposed: pairs completeness, reduction ratio and *F*-score. Pairs completeness (*PC*) indicates which share of the truly matching entity pairs are preserved after blocking. *PC* thus corresponds to a recall measure and a high value is important for effectiveness. The reduction ratio (*RR*) measure indicates the fraction of all possible entity pairs which is eliminated by blocking; it indicates how far the search space is reduced and thus efficiency is improved. *F*-score combines pairs completeness (*PC*) and reduction ratio (*RR*) via a harmonic mean, $F\text{-}score = \frac{2\times PC\times RR}{PC+RR}$.
- *Effectiveness measures (achieved max. values):* The effectiveness of entity matching is commonly determined with the standard measures precision (*P*), recall (*R*), and *F*-measure with respect to a manually determined "perfect" result. Alternative, an accuracy measure (*A*) is being used. These measures are formally defined as follows. The set of derived correspondences is comprised of True Positives (*TP*) and False Positives (*FP*), i.e., correctly and falsely proposed matches. False Negatives (*FN*) are true correspondences that have not been identified, while True Negatives (*TN*) are false matches which have been correctly discarded. Precision is defined as $P = \frac{|TP|}{|TP|+|FP|}$, recall as $R = \frac{|TP|}{|TP|+|FN|}$ and *F*-measure as $F = \frac{2P\times R}{P+R}$. Accuracy is defined as $A = \frac{|TP|+|TN|}{|TP|+|FP|+|TN|+|FN|}$. The values in parentheses in Table 3 give the maximum reported values for the considered effectiveness measures.
- *Efficiency measures:* The efficiency is commonly determined in terms of the execution time (*ET*). For training-based approaches the training time (*TT*) has to be considered additionally.The values in parentheses in Table 3 give the reported value ranges for the considered efficiency measures.

4.2. Comparison

Table 3 gives a summary about the evaluations of the considered frameworks as reported in the corresponding research papers. All frameworks offer a selection of matchers and combination approaches resulting in a huge number of possible configurations. Since it is impossible to exhaustively explore the configuration space, the reported evaluations cover only a restricted choice of configurations.

The summarized results indicate a substantial diversity between the different studies. The evaluations employ up to seven test problems from one to five domains. Popular domains for evaluation are the bibliographic, E-commerce and personal data domains. Some test problems from the RIDDLE repository (e.g., Cora, Restaurant) are used in the evaluations of five frameworks (Active Atlas, MARLIN, Operator Trees, FEBRL and STEM). Moreover, MOMA and STEM consider the same bibliographic match tasks DBLP–Google Scholar and DBLP–ACM. Most test problems are small and deal only with a few 100 entities; the largest test problem used in the evaluations matches 100,000 entities. All frameworks are evaluated on real-world test problems; the evaluations of BN, FEBRL and TAILOR additionally utilized artificially created test problems. While the use of real datasets is important to test for real-life conditions it is difficult to determine a perfect match result for them, especially for very large datasets. Moreover, real datasets are highly different in match difficulty making it problematic to generalize findings to other datasets. Artificially created match problems allow for more controlled test conditions for arbitrary many entities. However, artificially introduced errors are not domain-specific, do not necessarily cover all error types, and their distribution might be unrealistic.

Up to 18 attributes are considered for solving a given entity matching task. Most commonly the used attribute value matchers are based on the string similarity measures edit distance, (exact) equality, Jaro–Winkler and TF–IDF. From the large selection of 26 different similarity measures available for attribute value matching with FEBRL only two are used in the evaluation. For the evaluation of TAILOR it is unclear which of the five supported similarity measures are applied in the evaluation.

The effectiveness of the frameworks is commonly evaluated in terms of precision, recall, and *F*-measure. The weaker accuracy measure has been used in the evaluations of Active Atlas, the Multiple Classifier System and TAILOR. As expected the reported effectiveness values are typically high. However, these values do not allow a representative comparison of the relative effectiveness of the frameworks due to the high diversity of the test problems and employed evaluation methodology (e.g., different measures, training sizes, etc.).

Efficiency is evaluated for six of the 11 frameworks, mostly by measuring the execution times for matching (SERF, Multiple Classifier System, Operator Trees, Context Based System, FEBRL). For three training-based frameworks (Context Based System, Multiple Classifier System and Operator Trees) the time needed for training is reported.

The evaluations concerning blocking methods are rather unsatisfying so far. Only for FEBRL and TAILOR different blocking methods are compared in terms of reduction ratio, pairs completeness and *F*-score on artificial test problems. Experimental comparisons of blocking algorithms on real-world test problems are missing.

The influence of training selection is an important issue in evaluating training-based frameworks. Unfortunately, some evaluations provide few details on the selection (Context Based System, Multiple Classifier System, Operator Trees and TAILOR) and size of training data (FEBRL). Four of the eight training-based frameworks (Active Atlas, FEBRL, MARLIN and STEM) have investigated the issue of training selection and evaluated (semi-)automatic approaches. The influence of different training-set sizes is investigated for Active Atlas, MARLIN, Operator Trees, STEM and TAILOR. In the evaluation of the Multiple Classifier System a relatively large amount of training data is used thus favoring good match quality at the expense of a high manual effort for labeling.

In the following, we briefly discuss selected evaluation details for each framework. A summary of the effectiveness performance is given for those frameworks where the evaluation reported *F*-measure values.

4.2.1. Frameworks without training

BN: The evaluation considered artificial as well as real-world datasets on movies and CDs (IMDB, IMDB+FilmDienst, FreeDB). The evaluation investigated the impact of the choice of the threshold on effectiveness, the impact of data quality on effectiveness, i.e., how errors (e.g., typos or missing data) and error frequencies affect the result, and a comparison to DogmatiX [57], a previously proposed XML entity matching framework. In the evaluation all attribute values are considered as textual strings and edit distance is used for similarity computation. The framework achieved high precision and recall values in all cases.

MOMA (mapping-based object matching): The authors evaluated the framework on match tasks from the bibliographic domain. The evaluation considered three data sources (Google Scholar, DBLP and ACM Digital Library) and matching tasks for publications, authors, and venues. The combination of several matchers and mappings was shown to compensate weaknesses of individual strategies; the neighborhood matcher proved to be very valuable. Table 4 gives a summary of the maximal effectiveness performance reported for MOMA on five considered match tasks.

SERF (Stanford Entity Resolution Framework): The evaluation regarded comparison shopping and hotel datasets from Yahoo. Two manually defined match strategies were used for the evaluation. The evaluation considered the efficiency in terms of the runtimes of the match strategies. The runtime results mainly depend on the number of attribute value comparisons. The most dominant factor of the runtime for any algorithm compared in the evaluation turns out to be the total time for comparing string values.

4.2.2. Training-based frameworks

Active Atlas: The evaluation considered three datasets involving restaurants, companies and airports. The evaluation compared decision tree learning for rule-based matcher combination with two baseline combination approaches without training. For training selection a random approach and the semi-automatic active learning were compared. The experimental results show that the training-based approaches achieve higher accuracy values than the manually specified baseline EM strategies utilizing no training. Active learning required fewer labeled examples than random training selection.

MARLIN: The evaluation compared the performance of support vector machines (SVM) to decision trees revealing that SVM significantly outperform decision trees when training data is limited. Further experiments demonstrate the comparative

Table 4
Summary of effectiveness performance for MOMA (*F*-measure), number of entities in brackets.

	Venues	Publications	Authors
DBLP–ACM	0.99 (258)	0.99 (4910)	0.97 (6866)
DBLP–GS	–	0.89 (66,879)	–
GS–ACM	–	0.88 (66,557)	–

Table 5
Summary of effectiveness performance for MARLIN.

Match task	# Entities	*F*-measure
Reasoning	514	0.94
Face	349	0.97
Reinforcement	406	0.90
Constraint	295	0.94
Restaurant	864	0.92
Cora	1295	0.87

Table 6
Summary of effectiveness performance for FEBRL.

Match task	# Entities	*F*-measure
Census data	1000	0.8
DS-Gen-A	2500	0.9
DS-Gen-B	5000	0.9
DS-Gen-D	10,000	0.9

utility of static-active, weakly labeled negative and random training selection using TF–IDF and edit distance for attribute value matching and SVM as the learner. The highest performance was achieved when training data is a mix of examples selected using the static-active strategy and randomly chosen entity pairs. In situations where human labeling of negatives is expensive or infeasible (e.g., due to privacy issues), using weakly-labeled non-duplicates is found to be valuable for automatic acquisition of negative examples. Table 5 gives a summary of the maximal effectiveness performance reported for TAILOR on the four considered match tasks.

Multiple Classifier System: The evaluation of the system is rather limited. It is restricted to a single dataset (airline passengers) from a single domain. The dataset consisted of 25,000 passenger pairs of which 5000 where matching pairs. Various experiments compare accuracy, training time and matching time of different training-based combination approaches. For training a relatively large amount of training data is used (66% of the 25,000 examples) thus favoring good match quality however at the expense of a high manual overhead for labeling.

Operator Trees: The evaluation considered several datasets from different domains: organization names and addresses, personal data of hurricane evacuees, and three datasets from the RIDDLE repository (Cora, Restaurant and Bird). The approach is compared with a domain specific address cleansing solution as well as with the SVM. On the evacuees dataset the SVM offers better recall at higher precision. However, at lower precision the recall of Operator Trees is close to that of SVMs. Therefore, the authors propose to use Operator Trees as an efficient filter (with low target precision) before invoking SVMs. On the smaller RIDDLE datasets, the recall values achieved by operator trees are comparable to that of the SVM at the same precision. For the Bird dataset, significantly better recall values are obtained. The efficiency evaluations illustrate that a DBMS-based Operator Tree implementation executes significantly faster than SVM models, even if they employ blocking.

4.2.3. Hybrid frameworks

TAILOR: The evaluation compares various string similarity measures (Bigrams, Trigrams, EditDistance and Jaro) for attribute value matching, disjoint (sorting) and overlapping (sorted neighborhood) blocking, and matcher combination strategies utilizing training (probabilistic, decision tree and hybrid) and without training. The blocking evaluation showed no clear advantage for overlapping approaches with larger window size over disjoint methods with short key sizes. Maximum pairs completeness and reduction ratios of about 0.95 are achieved. The other evaluation results show that (i) attribute value matchers based on Jaro's algorithm perform better than the other attribute value matchers. (ii) training-based approaches outperform approaches without training.

FEBRL: In Ref. [4] two disjoint (sorting and sorted neighborhood) and two overlapping (bigram indexing and canopy clustering) blocking methods are compared in terms of reduction ratio, pairs completeness and *F*-score on artificial test problems with varying entity sizes. Both bigram indexing and canopy clustering outperform the two disjoint blocking methods with the right parameter settings. Pairs completeness with the two disjoint methods had a maximum of about 0.96, whereas the maximum for canopy clustering with TF–IDF is 0.98. Canopy clustering also achieved the best reduction ratio of 0.9 and best *F*-score of over 0.98. However, the method seems highly dependent on the choice of the threshold. With a suboptimal choice pairs completeness drops to 0.8. The evaluation in [17] evaluates the threshold and the nearest-based training selection methods. The evaluations showed that a SVM-based matcher combination with automatic training selection is often better than a numeric matcher combination without training. Table 6 gives a summary of the effectiveness performance reported for FEBRL on four considered match tasks.

STEM: The evaluation compares various string similarity measures (Cosine, EditDistance, Jaccard, Jaro–Winkler, Monge-Elkan, TF-IDF and Trigram), matcher combination strategies utilizing training (decision tree, SVM, logistic regression and

Table 7
Summary of effectiveness performance for STEM.

Match task	# Entities	*F*-measure
DBLP–ACM	4910	0.97
DBLP–GS	66,879	0.92
Parks	654	0.94
Restaurant	864	0.97

multiple learning) and manually configured strategies. The evaluation is performed for different training sizes and four match tasks. The evaluation shows that the automatically constructed EM strategies can significantly outperform manually configured combined strategies. This is especially true for difficult match tasks and often possible for small training sizes, thereby requiring a low labeling effort. The evaluation of several learners revealed that the SVM and the multiple learning approach produce the most stable results even for small training sizes. By contrast decision trees perform well only for large amounts of training data. Table 7 gives a summary of the effectiveness performance reported for STEM on the four considered match tasks. Comparing the results for STEM with MOMA, we observe that the training-based STEM could outperform MOMA for the DBLP–Google Scholar task (*F*-measure 0.92 vs. 0.89) but was less effective for DBLP–ACM (*F*-measure 0.96 vs. 0.99) since it did not utilize context matching such as MOMA. STEM achieved a higher *F*-measure for the restaurant task (from the RIDDLE repository) than Marlin (0.97 vs. 0.92). However, these singular performance observations cannot be generalized to other test cases.

Context Based Framework: The evaluation considered match tasks from two domains, personal webpages and bibliographic references (publications, authors, departments and organizations). The authors compare various attribute value and context matchers (eTFIDF, connection strength), matcher combination strategies utilizing training (mainly SMOreg, Logistic) and manually configured strategies. The evaluation shows that matcher combination strategies considering context and utilizing training can outperform manually configured combined strategies. Effectiveness performance measured in terms of *F*-measure range between 0.599 and 0.89 on the personal webpages task. Training times differ depending on the utilized learner from 0.11 s to 53.66 s. The application times range from less then 1 s to less than 5 s depending on the size of the dataset.

5. Summary and discussion

Entity matching frameworks allow the combined use of blocking and multiple matcher algorithms to effectively solve diverse match tasks. They have to meet challenging and partly contradicting requirements, in particular high effectiveness, efficiency, genericity and low manual effort. To assess the current state of art we comparatively analyzed the functionality and published evaluation results of 11 proposed research prototypes for entity matching. The comparisons are based on a common set of criteria. Criteria for the functional comparison include the supported choices for blocking methods, matchers, combination of matchers, and training selection. Criteria for the evaluation comparison consider the used test problems, applied match strategies and the achieved effectiveness and efficiency performance. The proposed criteria have value beyond the systems considered here but should enable to categorize and comparatively assess further entity matching frameworks and their evaluations.

Our study indicates a research trend towards using supervised (training-based) and hybrid approaches to semi-automatically determine an EM strategy for a given match task. Most of the considered frameworks including the most recent ones, Operator Trees and the Context Based Framework, employ supervised learners to derive numerical combination functions or match rules specifying how multiple matchers should be combined for deriving a match decision. Hybrid frameworks provide the largest scope of methods for solving entity matching tasks, in particular for blocking and the combined use of several matchers. Among the considered hybrid frameworks, FEBRL offers the largest selection of different blocking strategies and attribute value matchers, while the Context Based Framework supports the largest number of learners as well as context matching. Unfortunately, the flexibility of the hybrid frameworks comes at the price of an increased complexity for users to choose an appropriate method (selection of matchers to be considered, size and selection of training data) – despite the use of supervised machine learning approaches.

The functional comparison reveals a number of further research directions. All frameworks focus on offline matching, i.e., they do not yet cover online matching. The definition of the blocking key is not yet derived (semi-)automatically from training data but has to be specified manually in all considered frameworks. While attribute value matchers are well supported the combination of context and attribute matchers is not and should be further studied. Training-based EM frameworks should provide more support for (semi-)automatic selection of suitable training data with low labeling effort. So far training-based approaches only helped to optimize some decisions, e.g., determining parameters for matchers (e.g., similarity thresholds) and combination functions (e.g., weights for matchers) while other decisions (e.g., selection of the similarity functions and attributes to be evaluated) still have to be determined manually.

The published framework evaluations used diverse methodologies, measures, and test problems making it difficult to assess the effectiveness and efficiency of each single system. While the reported evaluation results are usually very positive, the tests so far mostly dealt with small match problems so that the scalability of most approaches is unclear. Hence,

scalability to large test cases needs to be better addressed in future frameworks. Some recent work regarding scalability has focused on computational aspects of string similarity computation [50,1,30,61] and time-completeness trade-offs [39]. Furthermore, we see a strong need for comparative performance evaluations of different frameworks and EM strategies. Standardized benchmarks for entity matching are needed for comparative investigations; first proposals exist [44,58] but have not yet been implemented or applied. Published evaluation results should also be reproducible by other researchers, ideally by providing the prototype implementations and test data.

References

[1] A. Arasu, V. Ganti, R. Kaushik, Efficient exact set-similarity joins, in: Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB '06), 2006, pp. 918–929.
[2] A. Arasu, C. Ré, D. Suciu, Large-scale deduplication with constraints using dedupalog, in: Proceedings of the 25th International Conference on Data Engineering (ICDE '09), 2009, pp. 952–963.
[3] C. Batini, M. Scannapieco, Data Quality: Concepts, Methodologies and Techniques, Data-Centric Systems and Applications, Springer, 2006.
[4] R. Baxter, P. Christen, T. Churches, A comparison of fast blocking methods for record linkage, in: Proceedings of the Ninth ACM SIGKDD Workshop on Data Cleaning, Record Linkage and Object Consolidation, 2003, pp. 25–27.
[5] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S.E. Whang, J. Widom, Swoosh: a generic approach to entity resolution, VLDB J. 18 (1) (2009) 255–276.
[6] I. Bhattacharya, L. Getoor, A latent dirichlet model for unsupervised entity resolution, in: Proceedings of the Sixth SIAM International Conference on Data Mining (SDM '06), 2006, pp. 47–58.
[7] I. Bhattacharya, L. Getoor, Collective entity resolution in relational data, ACM Trans. Knowl. Discov. Data 1 (1) (2007) 5.
[8] I. Bhattacharya, L. Getoor, Query-time entity resolution, J. Artif. Intell. Res. (JAIR) 30 (2007) 621–657.
[9] M. Bilenko, S. Basu, M. Sahami, Adaptive product normalization: using online learning for record linkage in comparison shopping, in: Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM '05), 2005, pp. 58–65.
[10] M. Bilenko, B. Kamath, R.J. Mooney, Adaptive blocking: learning to scale up record linkage, in: Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM '06), 2006, pp. 87–96.
[11] M. Bilenko, R.J. Mooney, Adaptive duplicate detection using learnable string similarity measures, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03), 2003, pp. 39–48.
[12] M. Bilenko, R.J. Mooney, On evaluation and training-set construction for duplicate detection, in: Proceedings of the KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation, 2003, pp. 7–12.
[13] S. Chaudhuri, B.-C. Chen, V. Ganti, R. Kaushik, Example-driven design of efficient record matching queries, in: Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB '07), 2007, pp. 327–338.
[14] S. Chaudhuri, V. Ganti, D. Xin, Mining document collections to facilitate accurate approximate entity matching, PVLDB 2 (1) (2009) 395–406.
[15] Z. Chen, D.V. Kalashnikov, S. Mehrotra, Exploiting relationships for object consolidation, in: Proceedings of the International Workshop on Information Quality in Information Systems (IQIS '05), 2005, pp. 47–58.
[16] Z. Chen, D.V. Kalashnikov, S. Mehrotra, Exploiting context analysis for combining multiple entity resolution systems, in: Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09), 2009, pp. 207–218.
[17] P. Christen, Automatic training example selection for scalable unsupervised record linkage, in: Proceedings of the 12th Pacific-Asia on Conference on Knowledge Discovery and Data Mining (PAKDD '08), 2008, pp. 511–518.
[18] P. Christen, FEBRL: a freely available record linkage system with a graphical user interface, in: Proceedings of the Second Australasian workshop on Health Data and Knowledge Management (HDKM '08), Australian Computer Society Inc., Darlinghurst, Australia, Australia, 2008, pp. 17–25.
[19] W.W. Cohen, H.A. Kautz, D.A. McAllester, Hardening soft information sources, in: Proceedings of the Sixth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '00), 2000, pp. 255–259.
[20] W.W. Cohen, P. Ravikumar, S.E. Fienberg, A comparison of string distance metrics for name-matching tasks, in: Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb '03), 2003, pp. 73–78.
[21] A. Culotta, A. McCallum, Joint deduplication of multiple record types in relational data, in: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management (CIKM '05), 2005, pp. 257–258.
[22] X. Dong, A.Y. Halevy, J. Madhavan, Reference reconciliation in complex information spaces, in: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD '05), 2005, pp. 85–96.
[23] U. Draisbach, F. Naumann, A comparison and generalization of blocking and windowing algorithms for duplicate detection, in: Proceedings of QDB 2009 Workshop at VLDB, 2009.
[24] M.G. Elfeky, A.K. Elmagarmid, V.S. Verykios, TAILOR: a record linkage tool box, in: Proceedings of the 18th International Conference on Data Engineering (ICDE '02), 2002, pp. 17–28.
[25] A.K. Elmagarmid, P.G. Ipeirotis, V.S. Verykios, Duplicate record detection: a survey, IEEE Trans. Knowl. Data Eng. 19 (1) (2007) 1–16.
[26] I.P. Fellegi, A.B. Sunter, A theory for record linkage, J. Am. Stat. Assoc. 64 (328) (1969) 1183–1210.
[27] H. Galhardas, D. Florescu, D. Shasha, E. Simon, AJAX: an extensible data cleaning tool, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD '00), 2000, p. 590.
[28] L. Gu, R. Baxter, D. Vickers, C. Rainsford, Record linkage: current practice and future directions, Tech. Rep., CSIRO Mathematical and Information Sciences, 2003.
[29] S. Guha, N. Koudas, A. Marathe, D. Srivastava, Merging the results of approximate match operations, in: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB '04), 2004, pp. 636–647.
[30] M. Hadjieleftheriou, A. Chandel, N. Koudas, D. Srivastava, Fast indexes and algorithms for set similarity selection queries, in: Proceedings of the 24th International Conference on Data Engineering (ICDE '08), 2008, pp. 267–276.
[31] M. Hadjieleftheriou, C. Li, Efficient approximate search on string collections, Seminar Given at ICDE, 2009.
[32] O. Hassanzadeh, F. Chiang, R.J. Miller, H.C. Lee, Framework for evaluating clustering algorithms in duplicate detection, PVLDB 2 (1) (2009) 1282–1293.
[33] M.A. Hernández, S.J. Stolfo, The merge/purge problem for large databases, in: Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD '95), 1995, pp. 127–138.
[34] H. Kang, L. Getoor, B. Shneiderman, M. Bilgic, L. Licamele, Interactive entity resolution in relational data: a visual analytic tool and its evaluation, IEEE Trans. Vis. Comput. Graph. 14 (5) (2008) 999–1014.
[35] N. Koudas, S. Sarawagi, D. Srivastava, Record linkage: similarity measures and algorithms, in: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD '06), 2006, pp. 802–803.
[36] H. Köpcke, E. Rahm, Training selection for tuning entity matching, in: Proceedings of the Sixth International Workshop on Quality in Databases and Management of Uncertain Data (QDB/MUD '08), 2008, pp. 3–12.
[37] M.-L. Lee, T.W. Ling, W.L. Low, Intelliclean: a knowledge-based intelligent data cleaner, in: Proceedings of the Sixth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '00), 2000, pp. 290–294.
[38] L. Leitão, P. Calado, M. Weis, Structure-based inference of XML similarity for fuzzy duplicate detection, in: Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM '07), 2007, pp. 293–302.

[39] R. Lengu, P. Missier, A.A.A. Fernandes, G. Guerrini, M. Mesiti, Time-completeness trade-offs in record linkage using adaptive query processing, in: Proceedings of the 12th International Conference on Extending Database Technology (EDBT '09), ACM, New York, NY, USA, 2009, pp. 851–861.
[40] A. McCallum, K. Nigam, L.H. Ungar, Efficient clustering of high-dimensional data sets with application to reference matching, in: Proceedings of the Sixth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '00), 2000, pp. 169–178.
[41] A. McCallum, B. Wellner, Conditional models of identity uncertainty with application to noun coreference, in: Advances in Neural Information Processing Systems, vol. 17. 2004, pp. 905–912.
[42] M. Michelson, C.A. Knoblock, Learning blocking schemes for record linkage, in: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI '06), 2006.
[43] S. Minton, C. Nanjo, C.A. Knoblock, M. Michalowski, M. Michelson, A heterogeneous field matching method for record linkage, in: Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM '05), 2005, pp. 314–321.
[44] M. Neiling, S. Jurk, H.J. Lenz, F. Naumann, Object identification quality, in: Proceedings of the International Workshop on Data Quality in Cooperative Information Systsems (DQCIS '03), 2003, pp. 187–198.
[45] H.B. Newcombe, J.M. Kennedy, S.J. Axford, A.P. James, Automatic linkage of vital records, Science 130 (1959) 954–959.
[46] J.C. Pinheiro, D.X. Sun, Methods for linking and mining massive heterogeneous databases, in: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD '98), 1998, pp. 309–313.
[47] E. Rahm, H.H. Do, Data cleaning: problems and current approaches, IEEE Data Eng. Bull. 23 (4) (2000) 3–13.
[48] V. Raman, J.M. Hellerstein, Potter's wheel: an interactive data cleaning system, in: Proceedings of the 27th International Conference on Very Large Data Bases (VLDB '01), 2001, pp. 381–390.
[49] S. Sarawagi, A. Bhamidipaty, Interactive deduplication using active learning, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02), 2002, pp. 269–278.
[50] S. Sarawagi, A. Kirpal, Efficient set joins on similarity predicates, in: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data (SIGMOD '04), 2004, pp. 743–754.
[51] P. Singla, P. Domingos, Multi-relational record linkage, in: Proceedings of the KDD-2004 Workshop on Multi-Relational Data Mining, 2004, pp. 31–48.
[52] P. Singla, P. Domingos, Entity resolution with markov logic, in: Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM '06), 2006, pp. 572–582.
[53] S. Tejada, C.A. Knoblock, S. Minton, Learning object identification rules for information integration, Inf. Syst. 26 (8) (2001) 607–633.
[54] S. Tejada, C.A. Knoblock, S. Minton, Learning domain-independent string transformation weights for high accuracy object identification, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02), 2002, pp. 350–359.
[55] A. Thor, E. Rahm, MOMA – a mapping-based object matching system, in: Proceedings of the Third Biennial Conference on Innovative Data Systems Research (CIDR '07), 2007, pp. 247–258.
[56] V.S. Verykios, G.V. Moustakides, M.G. Elfeky, A Bayesian decision model for cost optimal record matching, VLDB J. 12 (1) (2003) 28–40.
[57] M. Weis, F. Naumann, Dogmatix tracks down duplicates in XML, in: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD '05), 2005, pp. 431–442.
[58] M. Weis, F. Naumann, F. Brosy, A duplicate detection benchmark for XML (and relational) data, in: SIGMOD 2006 Workshop on Information Quality for Information Systems (IQIS '06), 2006.
[59] S.E. Whang, D. Menestrina, G. Koutrika, M. Theobald, H. Garcia-Molina, Entity resolution with iterative blocking, in: Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09), 2009, pp. 219–232.
[60] W.E. Winkler, Overview of record linkage and current research directions, Tech. Rep., US Bureau of the Census, Washington, DC, 2006.
[61] C. Xiao, W. Wang, X. Lin, Ed-join: an efficient algorithm for similarity joins with edit distance constraints, PVLDB 1 (1) (2008) 933–944.
[62] H. Zhao, S. Ram, Entity identification for heterogeneous database integration – a Multiple Classifier System approach and empirical evaluation, Inf. Syst. 30 (2) (2005) 119–132.
[63] H. Zhao, S. Ram, Entity matching across heterogeneous data sources: an approach based on constrained cascade generalization, Data Knowl. Eng. 66 (3) (2008) 368–381.

Hanna Köpcke is a Ph.D. student of Prof. Rahm at the database research group at the University of Leipzig, Germany. She received her diploma degree in Computer Science at the University of Dortmund Germany. Her research interests include data integration, entity resolution and data mining.

Erhard Rahm has been the Chair for Databases at the Institute of Computer Science at the University of Leipzig since 1994 (http://dbs.uni-leipzig.de). His current research areas are data integration, metadata and ontology management, and bio databases. He supervises the WDI lab at the University of Leipzig, a third-party funded innovation lab on web data integration. He has published several books and more than 100 peer-reviewed research papers. In 1988 he received his Ph.D. in Computer Science from the University of Kaiserslautern, and in 1993 his Postdoctoral Lecture Qualification. He was a visiting researcher at both the IBM Research Center in Hawthorne, NY, as well as Microsoft Research in Redmond, WA.