An Automatic Approach for Duplicate Bibliographic Metadata Identification Using Classification

Eduardo N. Borges, Karin Becker, Carlos A. Heuser, Renata Galante
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
{*enborges, karin.becker, heuser, galante*}*@inf.ufrgs.br*

***Abstract*—References are the main descriptive metadata used by digital libraries of scientific articles. These references can be represented by several formats and styles. Although considerable content variations can also occur in some metadata fields such as title, author names and publication venue. Duplicate records influence the quality of digital library services once they need to be appropriately identified and treated. This paper presents an approach to identifying duplicated bibliographic metadata. We extend our previous work so that instead of setting thresholds based on the scores returned by similarity functions, we use the scores to train classification algorithms which automatically identify duplicated references. The experiments show that the classifiers increases up to 11% the quality of results when compared to our unsupervised heuristic-based approach.**

***Keywords*-classification algorithms; information representation; information management;**

I. Introduction

Bibliographic references are the most important metadata stored in digital libraries of scientific articles. References can be represented using several metadata standards and different styles. The metadata catalog is usually organized as metadata records, which describe how articles are represented, including their style. The catalog also specifies how digital objects can be manipulated and retrieved.

A style is a set of rules for formatting references and citations. American Psychological Association (APA), Chicago, Institute of Electrical and Electronics Engineers (IEEE), Harvard and Vancouver are well-know styles. The following examples present four different bibliographic references retrieved from distinct digital libraries:

1) Y.-I. Chang, J.-R. Chen, and M.-T. Hsu, *Next-Generation Applied Intelligence*, LNCS. Springer, 2009, vol. 5579, ch. A Hash Trie Filter Approach to Approximate String Matching for Genomic Databases, pp. 816-825.
2) Chang, Y. et al. A Hash Trie Filter Approach to Approximate String Matching for Genomic Databases. In Proceedings of IEA/AIE '09. Springer-Verlag, Berlin, Heidelberg, 816-825.
3) Ye-In Chang, Jiun-Rung Chen and Min-Tze Hsu. A Hash Trie Filter Method for Approximate String Matching in Genomic Databases. Applied Intelligence, 33(1), 2010.
4) Chang, Y.-I. et al., 2010. A hash trie filter method for approximate string matching in genomic databases. *Applied Intelligence*, Vol. 33, No. 1, pp. 21-38.

Despite the differences, references 1 and 2 describe the same real-word article, and so do references 3 and 4. It can be noticed that in addition to style differences, content representation (title, authors' names, publication venue, among others) can also vary considerably. Recognizing duplicate references can be a hard task: heterogeneous references can represent the same article (e.g. references 1 and 2) and similar references can represent distinct papers (e.g. references 2 and 3).

The identification of duplicated records that refer to same real-world object can be a very difficult task. Metadata records can contain variations in spelling, omission of words, different styles, use of different standards and even spelling errors [1]. The task of finding matching records in one or more data sources is called deduplication, record linkage or instance matching [2].

Several approaches to record deduplication have been proposed in recent years. Most of these works focus on the deduplication task in the context of relational databases integration [1], [3], [4], [5], [6]. Few automatic approaches have been specifically developed for digital libraries [7], [8]. In the digital libraries domain, deduplication is generally based on the semantics of specific metadata fields. For example, fields that specify the authors of a digital object are among the most discriminative fields of a record, and hence this information can be used as a strong evidence of similarity for the deduplication process. There may exist several references with similar titles, but if the authors do not have similar names, most probably they are different real-world objects [7]. For instance, Baeza-Yates and Ribeiro-Neto [9] as well as Manning et al. [10] have published books with similar titles: Modern Information Retrieval and Introduction to Information Retrieval. Another specific problem of digital libraries is the variation in the representation of author names in bibliographic references and citations. Variations include abbreviations, inversions of names ordering and omission of suffixes such as Jr [11].

This paper presents an approach that combines similarity functions and classification algorithms for identifying duplicated bibliographic metadata. We extend our previous work [7], which proposed a set of similarity functions specially designed for the metadata content and a composition function to identify a replica. The main difference is that instead of setting similarity thresholds based on the scores returned by the similarity functions, we use the scores to train classification algorithms to automatically identify duplicated references. We have evaluated the classification models by the quality of deduplication process. Our experiments show that the combination of specific-purpose similarity functions and classification algorithms identified up to 98.6% of the duplicate references. These results represent an improvement of 11% when compared to the experiments using our original approach [7].

The rest of this paper is organized as follows. In Section II, we define deduplication and discuss related work. In Section III, we present our approach to deduplicate bibliographic metadata. We give details on the performed experiments and discuss the obtained results in Section IV. We also present a comparative analysis of algorithms. Finally, in Section V, we draw our conclusions and point out some future work directions.

II. Related Work

Several approaches to record deduplication have been proposed in recent years. Chaudhuri et al. [12] propose a probabilistic algorithm for retrieving the K records closest to a input record, according to a fuzzy match similarity function that considers the weight of words using the Inverse Document Frequency (IDF) [10]. Carvalho and da Silva [13] also use the vector space model to calculate the similarity between objects from multiple sources. Their approach can be used to deduplicate objects with complex structures such as XML documents.

The problem of bibliographic references deduplication is explicitly discussed by Lawrence et al. [8]. The authors propose algorithms for matching references from different sources based on metrics like edit-distance [14], word matching, phrase matching and subfield extraction. Usually, deduplication algorithms combine the values of these metrics (or any other similarity functions) by generating a similarity score between the records. If this score exceeds a similarity threshold, these records are considered sufficiently similar to represent the same real-world object, i.e. same bibliographic reference. Score values depend on the metadata content, the similarity functions and the matching algorithms. So the choice of effective similarity thresholds is not a trivial task.

Dorneles et al. [3] define a strategy to compare similarity scores. These scores are redefined according to the expected precision of record matching. This approach maps similarity scores into precision values using a training set. The choice of the expected precision is an easier task for the expert, but the identification of replicas still requires human intervention.

Other work have proposed strategies for the deduplication problem based on machine learning techniques, mostly supervised ones. These strategies estimate similarities and match duplicate records, without thresholds definition.

The Active Atlas system [6] performs record mappings to integrate different data sources. Attribute mapping rules are specified based on a training process using decision trees [15]. Cohen and Richman [5] propose a scalable and adaptive technique to group objects based on the string similarity of different records. The MARLIN system [4] explores a framework for identifying duplicated records using adaptive string similarity metrics applied to each field, according to the domain of their values. The system defines two similarity metrics: one based on edit distance and another based on Support Vector Machines (SVM) [16].

Carvalho et al. [1] propose an approach based on genetic programming to find suitable similarity functions based on the combination of multiple pieces of evidence. This approach is able to effectively identify whether two entries in a repository are replicas or not. The experiments show that this last one outperforms at least 6.5% the SVM-based method used by the MARLIN system [4].

Most of the papers described in this section contribute with solutions for the general record deduplication problem, which do not take into account the specifics of the digital library domain. In our previous work [7] we propose an approach for metadata record deduplication that is based on a set of similarity functions specially designed for the digital library domain. These functions compare proper nouns and support the following features: variations of spelling, omission of middle names, abbreviations and inversions of names ordering. When compared to the experiments presented in Carvalho et al. [1], our approach performed slightly better in a dataset containing portions of the metadata records from two real digital libraries, and presented a statistic tie in a dataset with article references data. Our strategy greatly reduces the number of comparisons that use string matching algorithms using an efficient two-phase blocking method, but it is sensitive to the definition of similarity thresholds, such as minimum percentage of authors matches and minimum title distance. In this paper, we propose to use classification algorithms to avoid the burden of similarity threshold definition.

III. Deduplication Approach

This section presents our approach to deduplicate bibliographic metadata. We define as duplicates or replicas two or more references that are semantically equivalent, i.e. references that describe the same publication item (digital object) indexed by a digital library. The metadata content is compared using similarity functions, which are chosen according to the domain of each metadata field. The score values returned by similarity functions are used to train a classification model which identify duplicates automatically, without requiring human intervention to set up similarity thresholds.

Our approach to deduplicate bibliographic metadata records are split into distinct phases according to the process of knowledge discovery in databases [17]. The following sections present these phases in detail.

A. Data selection, preprocessing and transformation

Metadata standards like Dublin Core and MARC 21 are represented by a flat structure composed by several metadata fields. Our deduplication approach selects only metadata fields shared by different bibliographic references like books, articles, papers and Web pages. We propose to adopt only title, authors' names and publication year because these are the common attributes found in majority of references. In addition, they are less susceptible to noise when compared to metadata fields such as publication venue, page numbers, among others. For instance, "TKDE" and "IEEE Transactions on Knowledge and Data Engineering" describes the same Journal but do not share any substring and it is quite common to find references that omit page numbers. Besides title, authors' names and publication year, we assume there is a class field that means which real-word article the metadata record refers.

Then, we apply preprocessing operations in order to clean and normalize the selected metadata content. Table I shows a list of cleaning and normalization operations and the metadata fields over which they are applied. The first step is to clean all selected metadata fields by applying usual string transformations. Then the publication year is transformed into a valid integer in the domain. We use the four leftmost digits to extract the year from dates or timestamps in ISO standard as "2011-09-20 08:32:45". Finally, besides the typical transformations in authors' names, we define the delimiter characters to be used by similarity functions that compare this metadata field.

After preprocessing, the references are combined in pairs (ref$_i$, ref$_j$) generating new records combining the metadata fields of any two different references. The similarity functions proposed in [7] are applied on each new record, i.e. to compare pair of distinct references. The similarity scores are added as new fields of each record.

Table II shows the new fields and corresponding functions. These similarity functions return integers or real values varying in the range [0,1]. There are performed differences or similarities between the pairs of publication years, authors' names and titles. If the pair of original classes are equal, a new binary class labeled "duplicated pair" is defined with value *yes*. Otherwise, the value *no* is assigned to duplicated pair. Then original string fields *authors*$_i$, *title*$_i$ and *class*$_i$, where i is the reference identifier, are removed. Only numerical fields with different distributions remain.

Recent solutions in the literature [18], [19], including multiple blocking methods, could be used to optimize the preprocessing phase by applying the similarity functions only over the best candidates for matching. However, some duplicates could not be detected because they could not have been marked as candidates.

Our strategy avoids string fields because only numerical similarity scores are used as input of the classification algorithms, allowing the use of several types of classifiers.

B. Data mining and interpretation of results

The goal of the mining phase is to train a classification model for predicting whether two distinct bibliographic references refer to the same real-world article. Hence the data transformed as described in Section III-A are used as input of a classification algorithm. We have experimented with distinct types of classifier algorithms (based on function, rules, decision trees and Bayes theorem) in order to understand the properties of the data. So far, we have experimented with the following classification algorithms:

- SMO [20] - variation of SVM [16], function-based;
- Multilayer Perceptron [21] - artificial neural network, function-based;
- Naive Bayes [22] - based on the Bayes theorem;
- AdaBoost.M1 [23] - boosting meta-algorithm;
- RIPPER [24] - rule-based;
- C4.5 [25] - based on decision trees.

Evaluation results are interpreted as the quality of the deduplication process. The classification results can be evaluated using three quality measures: precision, recall and balanced f-measure (F1) [10].

IV. Experimental Evaluation

This section describes the experiments developed in order to test the use of classification algorithms for automatically identify duplicated bibliographic metadata. We used a real database consisting of scientific articles references. The classification algorithms are evaluated by the quality of deduplication process. We have performed the experiments in a standard PC using Weka[1] data mining tool [26].

[1] http://www.cs.waikato.ac.nz/ml/weka/

Table I
PREPROCESSING STEPS

Metadata field	**Cleaning and normalization operation**
title, year, authors and class	remove null values remove single and double quotes remove accents switch to lowercase
year	remove any character other than digits if there are only two digits, add 19 before them if there are more than four digits, extract the four leftmost digits
authors	set author delimiter character (e.g. “;”) set inversion delimiter character (e.g. “,”) remove numbers insert space between abbreviations remove double spaces remove spaces at the beginning and end of strings remove identifiers

Table II
DISTANCE AND SIMILARITY FUNCTIONS APPLIED TO EACH PAIR OF REFERENCES

New metadata field	**Distance or similarity function**
authors number$_i$	numbers of authors from ref$_i$
authors number$_j$	numbers of authors from ref$_j$
authors number diff	absolute difference between the numbers of authors ($\|$authors number$_i$ $-$ authors number$_i$$|$)
year diff	absolute difference between the publication years ($\|$year$_i$ $-$ year$_j$$|$)
authors diff	difference between the authors according to the algorithm NameMatch [7]
authors sim	similarity between the authors based on the normalized *authors diff*
title diff	edit-distance [14] between title$_i$ and title$_j$
title sim	similarity between titles based on the normalized difference *title diff*
duplicated pair	binary class (*yes* if class$_i$ $=$ class$_j$ or *no* otherwise)

A. Dataset

The dataset used in our experiments was extracted from the Cora Collection. Cora is a collection of references extracted from a search engine for research papers in Computer Science [27]. References are segmented into multiple fields by an information extraction system, resulting in some crossover noise among the fields. For instance, some publication dates are captured in some fields other than year. There are 2191 records distributed in 305 distinct classes in the raw data. This collection has been used for experimental evaluation of related work [7], [1], [28], [4]. Table III presents the structure of Cora metadata records.

The records were processed according the procedure detailed in Section III-A. The 2191 instances were combined in pairs generating approximately 2.4 million of new records. The distance and similarity functions (Table II) were applied on each new record.

Finally, we have randomly selected 10% of instances to compose our dataset, from which 3035 (1.3%) instances were labeled as replicas (*duplicated pair* $=$ *yes*) and 236,879 (98.7%) as distinct real-world objects (*duplicated pair* $=$ *no*). The number of instances was reduced because the classification algorithms run entirely in memory and they have not linear complexity.

B. Deduplication results

Experiments results using numerical metadata fields are summarized in Table IV. This table presents type, name and the specific parameters of each algorithm that achieved the best results. Besides, it shows three quality measures (in %) considering only the class of interest *duplicated pair* $=$ *yes*: precision (P), recall (R) and balanced f-measure (F1). The models were evaluated using 10 fold cross-validation.

We have also experimented to discretize the numerical fields using a multi-interval discretization filter [29], hence transforming numeric into nominal attributes. Results are summarized in Table V, which presents the same information of Table IV. In addition, it shows the gain in F1 (also in %) when compared to results without discretization of Table IV.

C. Analysis of the results

In this section we examine the experiments results. First, we performed a comparative analysis of classification algorithms applied to deduplication. Then, our approach is compared with the baseline experiments developed in [7].

1) Evaluation of classifiers: Observing Table IV, we notice that all experiments achieved recall values higher than 89%, i.e. all tested algorithms were very effective for identifying replicas. When analyzing only the recall, the best result was yielded by the Naive Bayes algorithm. However, this same classifier achieved precision value lower than

Table III
THE STRUCTURE OF CORA METADATA RECORDS (RAW DATA)

Metadata field	Description
id	unique identification of a record
title	article title
authors	authors' names
year	publication year
venue	publication venue
other	other information contained in the reference such as page numbers, volume and issue
all	full reference (without field segmentation)
class	which real-word article this record refers

Table IV
DEDUPLICATION RESULTS USING NUMERIC ATTRIBUTES (*duplicated pair* = *yes*)

Type	Algorithm	Parameters	P	R	F1
function	SMO	linear kernel	77.3	92.1	84.1
function	Multilayer Perceptron	6 internal nodes, 200 epochs	77.2	97.1	86.0
Bayes	Naive Bayes	default	69.4	**98.6**	81.5
meta	AdaBoost.M1	Naive Bayes default	85.5	89.2	87.3
rules	RIPPER	one optimization phase	86.0	95.9	90.7
tree	C4.5	25% confidence for pruning, at least 2 instances per leaf	**88.2**	95.2	**91.6**

Table V
DEDUPLICATION RESULTS USING NOMINAL ATTRIBUTES (*duplicated pair* = *yes*)

Type	Algorithm	Parameters	P	R	F1	G_{F1}
function	SMO	linear kernel	82.8	95.2	88.6	**5.4**
function	Multilayer Perceptron	6 internal nodes, 200 epochs	**88.9**	91.3	90.1	4.7
Bayes	Naive Bayes	default	75.1	**98.6**	85.3	4.7
meta	AdaBoost.M1	Naive Bayes default	85.5	89.2	87.3	0.0
rules	RIPPER	one optimization phase	86.0	93.0	89.4	-1.5
tree	C4.5	25% confidence for pruning, at least 2 instances per leaf	86.1	95.0	**90.3**	-1.3

70%. Low precision denotes that many false positives were returned, which decreases the deduplication quality. The best values of precision were presented by the algorithm based on trees.

The overall quality of deduplication can be assessed by the f-measure. RIPPER and C4.5 showed the best F1 results, yielding values superior to 90%.

The results of Table V present evidences that attributes discretization improved the precision of algorithms based on function despite a slight fall in the recall. The best precision in this second evaluation was achieved by Multilayer Perceptron. All algorithms achieved recall values higher than 89%, emphasizing the results obtained by Naive Bayes. Multilayer Perceptron, RIPPER and C4.5 performed F1 around 90%.

The use of nominal attributes rather than numerical improved F1 results in 2% on average. For function-based classifiers and Naive Bayes the improvement was around 5%.

Tables VI and VII presents the deduplication results for both classes. Besides F1 values for each class, it shows the macro and micro averages of F1. Macro F1 achieved between 90.6 and 95.8% using numeric attributes and between 92.6 and 95.1% using nominal attributes. We notice that the quality of the identification of distinct citation pairs (represented by $F1_{no}$) is very high for all classification algorithms. The metric F1 for the class *duplicated pair* = *no* ranged between 99.7 and 99.9%. This behavior occurs because the great majority of instances (98.7% of the dataset) did not correspond to pairs of duplicated citations. Consequently, the micro average is very close to F1 results for non-replicas.

2) Comparison with baseline: Table VIII presents the best deduplication results considering our classification approach reported in this paper and our unsupervised heuristic-based approach previously proposed in [7]. It shows precision (P), recall (R) and balanced f-measure (F1) considering the class of interest *duplicated pair* = *yes*.

The results obtained by using classifiers outperformed the heuristic-based approach considering all three quality metrics. They represent a gain of 5% in precision, 17% in recall and 11% in F1. In sum, the use of classification algorithms have improved the quality of deduplication up to 11% yielding F1 values up to 91.6% using numeric attributes.

Table VI
DEDUPLICATION RESULTS USING NUMERIC ATTRIBUTES (*both classes*)

Algorithm	**F1**$_{yes}$	**F1**$_{no}$	**Macro F1**	**Micro F1**
SMO	84.1	99.8	92.0	99.6
Multilayer Perceptron	86.0	99.8	92.9	99.6
Naive Bayes	81.5	99.7	90.6	99.5
AdaBoost.M1	87.3	99.8	93.6	99.7
RIPPER	90.7	99.9	95.3	99.8
C4.5	91.6	99.9	95.8	99.8

Table VII
DEDUPLICATION RESULTS USING NOMINAL ATTRIBUTES (*both classes*)

Algorithm	**F1**$_{yes}$	**F1**$_{no}$	**Macro F1**	**Micro F1**
SMO	88.6	99.8	94.2	99.7
Multilayer Perceptron	90.1	99.9	95.0	99.7
Naive Bayes	85.3	99.8	92.6	99.6
AdaBoost.M1	87.3	99.8	93.6	99.7
RIPPER	89.4	99.9	94.7	99.7
C4.5	90.3	99.9	95.1	99.7

Table VIII
DEDUPLICATION RESULTS FOR BOTH APPROACHES (*duplicated pair* $=$ *yes*)

Approach	**P**	**R**	**F1**
unsupervised heuristic-based	83.9	81.3	82.6
classification-based	**88.2**	**95.2**	**91.6**

V. CONCLUSION

This paper presents an extension of a previous work to identify duplicated bibliographic metadata. Instead of setting similarity thresholds, we use the scores returned by the similarity functions specially designed for the metadata content to train classification algorithms which identify duplicated references.

The main benefits of using classification algorithms are to increase the quality of the deduplication process and to identity duplicates automatically, without requiring human intervention.

The results of our experiments show that the classification algorithms, combined with the similarity functions, identify up to 98.6% of duplicated citations with quality up to 91.6% measured according to F1. Our classification-based approach increases up to 11% the quality of results when compared to the unsupervised heuristic-based approach.

For small databases, better results can be obtained with a minimally acceptable cost in data transformation phase. However, for very large databases with millions of references, as The Collection of Computer Science Bibliographies[2], the cost can be very high or even prohibitive. It is necessary to calculate the values of new attributes by applying the similarity functions between authors' names and articles titles for all possible pairs of references. For enhancing the performance in the deduplication process, besides an efficiently blocking strategy, the evaluation of similarities can be parallelized using a programming model as MapReduce [30], [31].

Future work will include new experiments with multiple blocking strategies, parallelized evaluation of similarities and other datasets. The use of synthetic data allows varying parameters such as number of replicas and the distance between the original and the replicated values present in the repository of bibliographic references.

[2] http://liinwww.ira.uka.de/bibliography/

REFERENCES

[1] M. G. Carvalho, A. H. F. Laender, M. A. Gonçalves, and A. S. da Silva, "Replica identification using genetic programming," in *Proceedings of the ACM Symposium on Applied Computing*, 2008, pp. 1801–1806.

[2] A. Doan, N. F. Noy, and A. Y. Halevy, "Introduction to the special issue on semantic integration," *SIGMOD Record*, vol. 33, no. 4, pp. 11–13, 2004.

[3] C. F. Dorneles, M. F. Nunes, C. A. Heuser, V. P. Moreira, A. S. da Silva, and E. S. de Moura, "A strategy for allowing meaningful and comparable scores in approximate matching," *Information Systems*, vol. 34, no. 8, pp. 673–689, 2009.

[4] M. Bilenko and R. J. Mooney, "Adaptive duplicate detection using learnable string similarity measures," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 39–48.

[5] W. W. Cohen and J. Richman, "Learning to match and cluster large high-dimensional data sets for data integration," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 475–480.

[6] S. Tejada, C. A. Knoblock, and S. Minton, “Learning object identification rules for information integration,” *Information Systems*, vol. 26, no. 8, pp. 607–633, 2001.

[7] E. N. Borges, M. G. de Carvalho, R. Galante, M. A. Gonçalves, and A. H. F. Laender, “An unsupervised heuristic-based approach for bibliographic metadata deduplication,” *Information Processing and Management*, vol. 47, no. 5, pp. 706–718, 2011.

[8] S. Lawrence, C. L. Giles, and K. Bollacker, “Digital libraries and autonomous citation indexing,” *IEEE Computer*, vol. 32, no. 6, pp. 67–71, 1999.

[9] R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[10] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[11] M. Ley, *String Processing and Information Retrieval*, ser. Lecture Notes in Computer Science. Springer, 2002, vol. 2476, ch. The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives, pp. 481–486.

[12] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani, “Robust and efficient fuzzy match for online data cleaning,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data Conference*, 2003, pp. 313–324.

[13] J. C. P. Carvalho and A. S. da Silva, “Finding similar identities among objects from multiple web sources,” in *Proceedings of ACM International Workshop on Web Information and Data Management*, 2003, pp. 90–93.

[14] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions and reversals,” *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[15] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[16] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[17] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “The kdd process for extracting useful knowledge from volumes of data,” *Communications of the ACM*, vol. 39, no. 11, pp. 27–34, 1996.

[18] P. Christen, “A survey of indexing techniques for scalable record linkage and deduplication,” *IEEE Transactions on Knowledge and Data Engineering*, In Press, DOI: 10.1109/TKDE.2011.127, 2011.

[19] R. Baxter, P. Christen, and T. Churches, “A comparison of fast blocking methods for record linkage,” in *Proceedings of the ACM SIGKDD Workshop Data Cleaning, Record Linkage, and Object Consolidation*, 2003, pp. 25–27.

[20] J. C. Platt, “Using analytic qp and sparseness to speed training of support vector machines,” in *Proceedings of the Conference on Advances in Neural Information Processing Systems*, 1999, pp. 557–563.

[21] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, USA: Prentice-Hall, Inc., 2007.

[22] G. John and P. Langley, “Estimating continuous distributions in bayesian classifiers,” in *Proceedings of the Conference in Uncertainty in Artificial Intelligence*, 1995, pp. 338–345.

[23] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in *Proceedings of the International Conference on Machine Learning*, 1996, pp. 148–156.

[24] W. W. Cohen, “Fast effective rule induction,” in *Proceedings of the International Conference on Machine Learning*, 1995, pp. 115–123.

[25] J. R. Quinlan, *C4.5: programs for machine learning*. San Francisco, USA: Morgan Kaufmann Publishers Inc., 1993.

[26] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.

[27] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, “Automating the construction of internet portals with machine learning,” *Information Retrieval*, vol. 3, no. 2, pp. 127–163, 2000.

[28] M. G. Carvalho, M. A. Gonçalves, A. H. F. Laender, and A. S. da Silva, “Learning to deduplicate,” in *Proceedings of the ACM IEEE Joint Conference on Digital Libraries*, 2006, pp. 41–50.

[29] U. M. Fayyad and K. B. Irani, “Multi-interval discretization of continuous-valued attributes for classification learning,” in *Proceedings of the International Joint Conference on Artificial Intelligence*, 1993, pp. 1022–1029.

[30] G. Dal Bianco, R. Galante, and C. A. Heuser, “A fast approach for parallel deduplication on multicore processors,” in *Proceedings of the ACM Symposium on Applied Computing*, 2011, pp. 1027–1032.

[31] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.