Record Deduplication By Evolutionary Means

Marco Modesto, Moisés G. de Carvalho,
Walter dos Santos
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
31270-901 - Belo Horizonte - MG
Brasil
{mabm,moisesgc,walter}@dcc.ufmg.br

ABSTRACT

Identifying record replicas in digital data repositories is a key step to improve the quality of content and services available, as well as to yield eventual sharing efforts. Several deduplication strategies are available, but most of them rely on manually chosen settings to combine evidence used to identify records as being replicas. In this work, we present the results of experiments we have carried out with a Machine Learning approach for the deduplication problem. Our approach is based on Genetic Programming (GP), that is able to automatically generate similarity functions to identify record replicas in a given repository. The generated similarity functions properly combine and weight the best evidence available among the record fields in order to tell when two distinct records represent the same real-world entity. On a previous work, fixed similarity functions were associated to each evidence. On the present work, the GP will be also used to choose the best evidence and similarity functions associations. The results of the experiments show that our approach outperforms the baseline method by Fellegi and Sunter. It also outperformed the previous GP results, using fixed evidence associations when identifying replicas in a data set containing researcher's personal data.

1. INTRODUCTION

Deduplication is the task of identifying record replicas in a data repository that refer to the same real world entity or object, in spite of misspelling words, typos or different writing styles. Digital Libraries (DLs) are data repositories that can benefit from improvements in replica detection methods since the quality of the data they manage is crucial for providing useful services to their users. Many DLs, such as the ACM Digital Library[1], include OCR processed documents and citations in which misspelling and missing words are common. Also, the lack of conventions for author names, bibliographic reference descriptions, and book, paper and conference titles makes the the detection of replicas even harder to achieve.

Since deduplication is a time consuming task even for small collections, our aim is to foster a method to find the weights, identification bounds and the best evidence that

---

[1]http://www.acm.org

.

should be used for a given DL, using a small representative part of its data for training purposes.

This objective can be achieved by discovering similarity functions that are able to identify the replica records patterns that exist on the digital repository being deduplicated. The discovered similarity function must accomplish distinct, sometimes conflicting objectives, including: to efficiently maximize the identification of replicas and, at the same time, to avoid making mistakes during the process (e.g., to tell that two records are replicas when they are not). Having this in mind, we have chosen a genetic programming-based approach to handle this problem.

Genetic Programming (GP) [12] is a Machine Learning (ML) technique that helps finding good answers to a given problem where the search space is very large and when there is more than one objective to be accomplished. The fact that it has been successfully applied to this problem, has motivated us to use it as a basis for discovering good similarity functions for record replica identification.

In our experiments, we have achieved better results in finding record replicas in data sets containing researchers' personal information compared with the previous work using GP with fixed similarity and evidence associations [5] and than the baseline method by Fellegi and Sunter [10]. Fellegi and Sunter created a formal theory for replica identification using similarity boundaries to classify records as replicas or not.

Using the article citation data set, our GP-based approach outperforms the baseline method best result in more than 3% and the previous GP results on 2%, using the penalty based score presented on [5]. Another advantage of our GP-based approach is that its best similarity functions make use of less record fields as evidence than the baseline's one. Using less evidence to compute the similarity improves the deduplication efficiency, since time is spent computing the similarities between the values of only the most useful fields and more efficient similarity functions.

The remaining of this paper is organized as follow. In Section 2, we discuss related work. In Section 3, we introduce some GP basics. In Section 4, we present our approach and the deduplication framework we have developed. In Section 5, we detail the performed experiments and the results obtained with two data sets. In Section 6, we draw our conclusions and point out some future work directions.

2. RELATED WORK

Duplication detection is a growing research topic in DLs and database fields. To solve the inconsistencies present in

the data store on these repositories, it is necessary to design a similarity function that combines the information available, extracted from the record fields, in order to tell if a pair of records refers to the same real-world entity. For the case of citations, for instance, this problem was extensively discussed by Lawrence et. al. in [14] and in [13], they propose a number of algorithms for matching citations from different sources based on edit-distance, word matching, phrase matching, and subfield extraction.

In a more general context, Fellegi and Sunter [10] have formulated in their seminal work a series of statistical methods for dealing with this problem. In their methods, it is necessary to define the boundary values that are used to classify a pair of records as being replicas or not. Tools that implement these methods, such as Febrl [9], usually work with two boundaries as follows: (1) Positive Identification Boundary – if the similarity value lies above this boundary, the records are confirmed as replicas; (2) Negative Identification Boundary – if the similarity value lies above this boundary, the records are confirmed as not being replicas. For the situation in which similarity values stand between the two boundaries, the records are classified as "possible matches", and in this case, a human judgment is necessary.

Although good results may be obtained with the classical Fellegi and Sunter´s methods, they are not guaranteed to be optimal. Among the main challenges are the problems of choosing what evidence to use, and how to find the best weighting schema to apply to the chosen evidence. This has led the research community to develop a number of alternative methods [2, 3, 4, 6, 7, 8, 11, 15].

More related to our work, are that apply learning methods for deriving record-level similarity functions that combine field-level similarity functions, including the proper assignment of weights [2, 8, 15].

The adaptive approach presented in [8] consists in the use of examples for training a learning algorithm to evaluate the similarity between two given *names*, i.e., strings representing identifiers. This approach is applied to both clustering and pair-wise matching achieving satisfactory experimental results.

While other learning methods may require a training phase to be performed for each matching or deduplication task, our method aims at learning a similarity function that can be deployed for several distinct data sets within a same domain, thus avoiding the need for training in each deduplication task.

3. GENETIC PROGRAMMING

In evolutionary programming, the inspiration idea is a naturally observed process that influences virtually all living beings, the Natural Selection. Genetic programming [1, 12], one of the most known evolutionary programming techniques, can be viewed as an adaptive heuristics whose base ideas came from the gene properties and the natural selection system. It is a direct evolution of programs or algorithms used for the purpose of inductive learning, initially applied to optimization problems. Evolutionary techniques are also known for their capability of working with multi-objective problems, that are normally modeled as environment restrictions during the evolutionary process [1].

The main aspect that differs GP from other evolutionary techniques (e.g., genetic algorithms, evolutionary systems, genetic classifier systems) is that it represents con-

cepts and interpretation as a computer program - actually, even the data is viewed and manipulated in this way. This special characteristic enables the GP representation to model any other machine learning representation [1]. Other strong point in the GP representation that makes it different from many machine learning techniques is that the programs evolved are free, or variable, in length.

3.1 Generational Evolutionary Algorithm

In this work, the GP evolutionary process is guided by a *generational evolutionary* algorithm. This means that there are well-defined and distinct generation cycles. This is the basic idea around all evolutionary algorithms.

The steps describing the algorithm cycle are the following:

1. Initialize the population (with random or user provided individuals).

2. Evaluate all individuals in the present population, assigning a numeric rating or fitness value to each one.

3. Reproduce the best n individuals into the next generation population.

4. Apply the genetic operations (reproduction, selection, crossover and mutation) to all individuals in the present population.

5. Using a selection process, select m individuals that will compose the next generation with the best parents.

6. If the termination criterion is fulfilled, then continue. Otherwise, replace the existing generation with new generated population and repeat steps 2 to 5.

7. Present the best individual in the population as the output of the evolutionary process.

3.2 GP Experimentation Evaluation

In this work, the process of combining evidences to create a record similarity function using GP is separated in two phases:

- The GP training phase, in which the characteristics of similar records are learned.

- The testing phase, in which the best trees selected in the training set are used to identify replicas in a set of records different from the one used in the first phase in this work.

The second phase, besides measuring the real effectiveness of the generated solutions, also serves to verify if the trees evolved during the training phase have not been overspecialized for the features present in the training data set and are not generalized for other data sets. Our idea to guarantee generalization for the generated solutions is to use as a sample in the training phase a small but statistically representative part of the data available in the DL being examined in order to generate effective and generalizable similarity functions that could be used for the entire DL or for another DL with similar characteristics.

In this work, we have integrated the deduplication framework Febrl [9], based on the Fellegi and Sunter's method mentioned before, with our GP framework in order to create an experimental environment. Febrl is an open-source
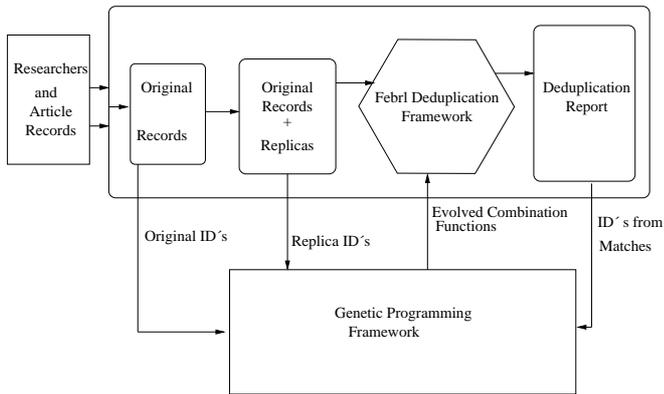
Figure 1: Our Experimental Environment

Symbol	Arity	Operation
*	2	multiplication
/	2	division
+	2	addiction
-	2	subtraction
n	0	random integer from 1 to 9
a to h	0	attribute similarities

Table 1: GP Terminal and Function Set

Parameter	Value
Number of Generations:	20
Records in Table:	500
Tables deduplicated per Individual:	4
Population:	20
Best Parents Selected :	20%
Mutation Rate:	2%

Table 2: Parameters of the First Set of Experiments

record-linkage framework, implemented in Python, that has been developed mainly for deduplication tasks over medical records.

In our experiments, we have used a fitness function based on a weighting-based penalty scheme. It uses the number of right and wrong replica identifications provided by Febrl.

We have used the same lower and upper boundaries of the standard Febrl setting (respectively 0 and 30), for the GP training and the test phases, in order to compare our results with the ones produced by the Fellegi and Sunter's method. The number of right identifications (RI) is the value that counts the number of pairs that were correctly identified as replicas using the generated function. The number of wrong identifications (WI) is the value that counts the number of pairs that were wrongly classified with this function. The total number of pairs classified as replicas is the sum of these two values.

The fitness strategy function used is

$$Fitness() = (RI * 10) - (WI * 2)$$

It gives a higher value for right decisions, but penalizes wrong ones with a 5/1 rate. In a nutshell, our GP-based approach tries to maximize these fitness functions by looking for individuals that can make more correct decisions and less replica identifications errors.

4. EXPERIMENTS

In this section, we present the results of the experiments we have performed to evaluate the GP-based approach to deduplication.

In our experiments, we have chosen to use the Fellegi and Sunter´s method as the comparison baseline, since it is well documented and it is the standard method used in the literature for comparison. Also, there are many deduplication frameworks, like Febrl [9], that implement this method and can be freely used.

The Fellegi and Sunter´s method works by summing up all evidence similarities available for all record fields in a single value:

$$FellegiSunter() = Sim_1() + Sim_2() + ... + Sim_n()$$

Then, the result is compared against the lower and upper boundaries, and classified as explained earlier.

4.1 Experiments with Researchers´ Personal Data

The kind of data used in this set of experiments is commonly extracted from articles, papers and curricula found in Web sites. In this context, the replica problem appears due to the different data schemata and to input errors caused by typos and misspelled words.

Since real data in this domain is not easily available for experimentation, due to privacy and confidentiality constraints, we have used the Data Set Generator Program (DSGP) available in the Febrl framework for creating the data sets used in this set of experiments. DSGP can create data sets containing names (based on frequency tables for given names and surnames), addresses (based on frequency tables for localities, postcodes, streets numbers, etc.), phone numbers and personal number ids (like the social security number). For more details, see [9].

The data fields available from DSGP are similar to those frequently found in researchers' personal data records. Accordingly, we have used DSGP to create a set of tables containing a list of original records. Using the original records, DSGP can create replicas and insert them in the original data set for deduplication experiments. These replicas are created using Febrl supplied look-up tables containing real world spelling variations for names and addresses. These errors are randomly inserted in a record for any of its fields.

Each one of the tables created have 500 records, where 375 are original ones and 125 are random replicas created from the originals. The GP training phase used three different tables for the evaluation of the deduplication performance of each evolving function. The GP testing phase used five other tables, distinct from the ones used in the training phase, for final evaluations.

The records in these tables have the following fields: author given name, author surname, street number, address1, address2, suburb, postcode, date of birth and age. The Febrl framework includes some of the most used similarity functions (e.g., Nysiss, Dmetaphone, Winkler, Bag of Words,

Method	FS	GP Best(1)	GP Avrg(1)	GP Worse(1)	GP Best(2)	GP Avrg(2)	GP Worse(2)
Table 1	70/3	70/0	70/0	64/1	72/0	72/1	68/4
Table 2	73/2	76/0	73/0	71/2	76/0	77/0	77/1
Table 3	78/3	77/0	76/0	66/1	77/0	79/0	72/6
Table 4	71/2	72/0	73/0	67/0	74/2	73/0	72/3
Table 5	73/4	73/0	74/0	65/1	76/0	73/0	70/3

Table 3: Experiments - (Right/Wrong Identifications and Precision Values)- Researchers´ Data Results

Method	FS	GP Best(1)	GP Avrg(1)	GP Worse(1)	GP Best(2)	GP Avrg(2)	GP Worse(2)
Table 1	694	700	700	638	720	718	672
Table 2	726	760	730	708	760	770	768
Table 3	774	770	760	658	770	790	708
Table 4	706	720	730	670	736	730	714
Table 5	722	730	740	648	760	730	694
Average	724.4	736	732	664.4	749.2	747.6	711.2

Table 4: Experiments - (Weight-Based Penalty Values) - Researchers´ Data Results

Date and Age Tolerance, GeoCode Distance, etc.). For the aforementioned record fields, the following suggested similarity measures provided by Febrl were employed:

- a - NYSIIS for given name

- b - WINKLER for given name

- c - DMETAPHONE for surnames

- d - WINKLER for surnames

- e - WINKLER for addresses

- f - Bag of Word Distance for addresses

- g - Character-wise field comparison with a maximum number of errors (different characters) tolerance for locality names

- h - String Edit Distance between locality names

- i - The geographical distance computed through the geographical distance between the given postcode fields

- j - Age difference with error tolerance

Notice that different similarity measures are applied to the same attribute. This setup was done in order to provide the GP process the freedom to choose the best similarity measure to be applied to a given attribute. It can be also possible to use a combination of two different similarity measures for the same attribute, in the same function, in order to improve the replica detection performance for the data set.

The settings applied to our GP framework for the first set of experiments are presented in Table 2. The number of generations is the GP design parameter used as the stop criterion for the algorithm. The tables deduplicated per individual value is the number of tables used for computing the fitness value for each individual during the training phase. The population parameter shows the number of individuals that are handled in a generation processing cycle. The mutation rate value indicates the probability of a given individual in a population to suffer a mutation.

Results obtained are presented in Table 4. In this table, the column FS shows the results obtained with the Fellegi and Sunter´s method, while the columns GP Best, GP Average and GP Worse correspond, respectively, to the results obtained with the Best, the Average and the Worse GP functions produced in the last generation by the two GP strategies, as indicated by the number within parenthesis. The first GP strategy is the same one implemented by Carvalho on [5], using fixed similarity functions applied to each evidence. The second GP strategy is the approach proposed on this work - using the GP process to select the best similarity function to be applied on each attribute. By average, we mean the individual (function) that is located at the middle position in the final ranking of the population.

Table 3 presents a raw accounting of the number of correct and wrong pairs identified by each function. Table 4 presents the values obtained by the weighting-based penalty scheme, used to evaluate the individuals during the training phase of the second strategy. The choice to show those specific individuals (i.e., Best, Average, and Worse) in the tables, was done to illustrate the population diversity, as well as to clarify the way the GP process actually works, by selecting the best generated answers in a given population.

The results show that the both GP approaches outperformed the baseline score level. The GP approach presented on [5] outperformed the baseline on 12 points and our GP approach on 25 points using the penaltyscore fitness measure. The different scores were caused by the different similarity functions that were evolved on each GP solution.

The best similarity function produced our GP strategy was the tree

$$Sim(record_1, record_2) = ((e + (f + c)) + b) + e$$

.

Observing the evidence used on this tree, it can be noticed that:

- The similarity function selected for the given name attribute was the winkler function.

- Both available similarity functions applied on the address attribute were combined

- The similarity function selected for the surname attribute was the dmetaphone encode function.

These evidence were similar to the ones resulted from the solution given by GP approach suggested on [5]. The difference between then rely on the similarity functions selected for the given name and address attributes Our GP process selected the winkler function for given name, and the former work used the nysiis function. Also, our GP process had combined both address attribute available and the former work only used the winkler similarity function alone.

4.2 Remarks

The results obtained from the experiments reinforce the following remarks made on [5]:

- It is not necessary to use all pieces of evidence in order to identify replicas: in specific data sets just a few set of fields are often sufficient to carry out this task.

- GP can be used as a tool to analyze the impact of the evidence during the replica detection process. The evolutionary process selects only the most relevant evidence, discarding those that are not useful.

5. CONCLUSIONS AND FUTURE WORK

Deduplication is a key issue for guaranteeing the quality of the information made available by modern DLs. In this paper, we presented a GP-based approach experiments to identify record replicas in data repositories. Our approach is able to automatically generate similarity functions to identify record replicas based on existing evidences present in the data. The generated similarity functions properly combine and weight the best evidences available among the record fields in order to tell when two distinct records represent the same real-world entity.

In our experiments, using data sets containing researchers' personal data, we have achieved better results than the baseline Fellegi and Sunter's method [10] and than the GP fixed similarity attribute association approach [5].

One of the advantages of using GP, which was highlighted by our experiments, is the fact that it allows us to analyze the final generated functions in order to infer the relevance of each evidence for the deduplication task. Observing the generated functions, it is possible to infer the best similarity function that should be used on a specific data attribute Moreover, at end of the GP processing, there is not only one solution, but a population of individuals (functions) that can execute the task in an equivalent or similar manner. This allows users to choose the best suitable option to fulfill their needs.

As future works, we plan to conduct new experiments using flexible similarity identification boundaries for both the GP and the baseline functions. By doing this, we could discover the best upper and lower boundaries in order to improve the resulting functions. Additionally, we intend to develop a clustering method to group replicas, thus helping to identity the number of replicas found for a given real-world entity in a repository.

The experiments on this problem also leaded to future plans on parallelization efforts, applied on the deduplication process and during the genetic programming evolving work.

6. REFERENCES

[1] Banzhaf, W., Nordin, P., E, R. E. K. R., and Francone, F. D. *Genetic Programming - An Introduction: On the Automatic Evolution of Computer Programs and Its Applications.* Morgan Kaufmann Publishers, 1998.

[2] Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., and Fienberg, S. Adaptive name matching in information integration. *IEEE Intelligent Systems 18*, 5 (September/October 2003), 16–23.

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

[4] Carvalho, J. C. P., and Silva, A. S. Finding similar identities among objects from multiple web sources. In *Proceedings of the fifth ACM International Workshop on Web Information and Data Management* (2003), pp. 90 – 93.

[5] Carvalho, M. G., Laender, A. H. F., Soares, A. S., and Gonçalves, M. A. Learning to deduplicate. In *Joint Conference of Digital Libraries - JCDL* (2006).

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

[7] Cohen, W. W. Data integration using similarity joins and a word-based information representation language. *ACM TOIS 18*, 3 (2000), 288–321.

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

[9] Freely Extensible Biomedical Record Linkage. http://sourceforge.net/projects/febrl.

[10] Fellegi, I. P., and Sunter, A. B. A theory for record linkage. *Journal of American Statistical Association 66*, 1 (1969), 1183–1210.

[11] Guha, S., Koudas, N., Marathe, A., and Srivastava, D. Merging the results of approximate match operations. In *Proc. of VLDB* (2004), pp. 636–647.

[12] Koza, J. R. *Genetic Programming: on the programming of computers by means of natural selection.* MIT Press, 1992.

[13] Lawrence, S., Giles, C. L., and Bollacker, K. D. Autonomous citation matching. In *Proceedings of the Third International Conference on Autonomous Agents* (1999), pp. 392–393.

[14] Lawrence, S., Giles, C. L., and Bollacker, K. D. Digital libraries and autonomous citation indexing. *IEEE Computer 32*, 6 (1999), 67–71.

[15] Tejada, S., Knoblock, C. A., and Minton, S. Learning object identification rules for information integration. *Information Systems 26*, 8 (2001), 607–633.