Intelligent Application for Duplication Detection

Plamen Paskalev, Anatoliy Antonov

Abstract: *The paper considers a realization of a software application, which performs identification of duplicated records in a database, containing customer information. Some of the most important works in this direction are overviewed. The selected algorithm is discussed. The problems and appropriate solutions due to handling of multi-language information are listed. The developed application, built using CLIPS engine and rule scripts is debated.*
Keywords: *Duplicate detection, data mining, record linkage, edit distance algorithm, Intelligent Interfaces, Artificial Intelligence, CLIPS, Computer Systems and Technologies*

INTRODUCTION

It is common case in the real life, that the databases contain records which points to the same object (book, person, etc.) but are not syntactically identical. Using of abbreviations or different front-office systems, integration of multiple data sources or just typographical errors and misspellings are the main reasons for this kind of problems with database content. This is a serious problem because it has harmful effects on the statistics, reporting features, preventing data-mining algorithms from discovering regularities in the data. For example, the marketing specialists in a large company need to know who their customers are, and they must have an extraordinarily clean customer list in order to communicate with them. Using wrong or misspelled addresses or sending multiple letters to the same person is not acceptable. Some other areas, where data, clear of this type of problems is crucial are customer matching, marketing, merging different sources of data, tracking retail sales, medical records, library records, etc.
This article describes the approach, used for realization of duplicate detection in a medium-sized international bank. The application is realized using CLIPS engine. The analysis of the data records and algorithms for comparison and determining of potential duplicates are realized as CLIPS rules. The main problems in the realization, like supporting of different languages, handling of language-specific special symbols, fast response of the system, etc., as long as the selected technology and methodology are discussed.

DUPLICATION FINDING

The problem with finding and fixing of duplicated data is typically handled using manual data cleaning process. This is a rather tiresome process, especially in larger databases. So, there exists many works, concentrated on the way of using specially designed software for performing of this task. Some works have addressed the problem of identifying duplicate records, referred to as record linkage [1,2], the merge/purge problem [3], duplicate detection [4,5], hardening soft databases [6], reference matching [7], and entity-name clustering and matching [8]. For determining of the level of likeness standard string similarity metrics such as edit distance [9] or vector-space cosine similarity [10] are used. Investigations in the using of pairing functions [8, 5, 11] that combine multiple standard metrics were made. Accurate similarity computations require adapting string similarity metrics for each field of the database with respect to the particular data domain. Trainable similarity measures [12] were suggested instead of hand-tuning a distance metric for each field.
The problem of identifying duplicate records in databases was originally identified by Newcombe [1] as record linkage in the context of identifying medical records of the same individual from different time periods. Fellegi and Sunter [19] developed a formal theory for record linkage and offered statistical methods for estimating matching parameters and error rates. In his work in statistics, Winkler proposed using EM-based methods for

obtaining optimal matching rules [2]. That work was highly specialized for the domain of census records and used hand-tuned similarity measures. Hernandez and Stolfo [3] developed the sorted neighborhood method for limiting the number of potential duplicate pairs that require distance computation, while McCallum et. al. proposed the canopies clustering algorithm [16] for the task of matching scientific citations. Monge and Elkan developed the iterative merging algorithm based on the union-find data structure [18] and showed the advantages of using a string distance metric that allows gaps [17]. Cohen et. al. [6] posed the duplicate detection task as an optimization problem, proved NP-hardness of solving the problem optimally, and proposed a nearly linear algorithm for finding a local optimum using the union-find data structure. Cohen and Richman have proposed an adaptive framework for duplicate detection that combines multiple similarity metrics [8]. Sarawagi and Bhamidipaty [5] and Tejada et. al. [11] developed systems that employ active learning methods for selecting record pairs that are informative for training the record-level classifier that combines similarity estimates from multiple fields across different metrics.

SOFTWARE SOLUTION

1. APPLICATION'S TASKS

The discussed application realizes duplication detection in a heterogeneous set of data bases, belonging to sub-banks of a medium-sized European bank. The data, collected in the independent sub-banks (institutes) contains information for customers of the corresponding institute only. The data has been entered using different front-office systems, in different software realizations. The information in the different institutes is written down using different European languages, using the corresponding special symbols, specific for the certain language (Ö, Ä, Ü, ß, š, r, í, á, c, e etc.).

The main goal of the application is to collect and clean the data for the bank customers, to create a proper hierarchy of the database and to eliminate potential duplicates in the customer's information.

The standardization of the customer's data and the structural improvement are essential for building of proper view on the company customers and portfolio as long as for the computation of credit limits for the customers. The task of the customer unification consists in the identification and allocation of duplications based on thorough analysis of the information of a limited number of data fields, comparing degree of similarity of the data. After production of the duplication lists, they are sent to the corresponding institutes and responsible persons. The clearing of the duplicates is realized from the responsible persons according to their authorization rights.

2. REALIZATION

The duplication finding module was realized as part of a larger WEB application, responsible for maintaining the data, collected from different institutes. The existing works in the area of duplication finding were investigated; the best approaches for this case were selected and implemented. The data is received in 3 different ways: via file transfer, which updates database during overnight batch mode, via WEB services and via GUI interface. One important requirement is the time for response. This is the reason why the speed of implemented algorithms has very high priority.

First stage of the analysis is check of duplications on column level. The analysis depends on the contents of the data fields. Comparing rules, calculating the level of similarity are applied. For each data field for a pair of customers a degree of similarity is determined using the rules. In case the content is the same, the result is 100%, similar fields are scored with result < 100%. The final result is determined after weighting with consideration of different criteria defining the importance of the fields.

In the following sections the main steps in the realization are discussed in more details. Finally, the technical aspects of the realization are overviewed.

a) Collecting the data

The application performs analysis of a previously defined subset of data columns from customer's database. The name, address, postal code, sex, birth date is being compared in an attempt to find duplicate records. A weight is set to each of the columns, which defines the importance of the column in the general duplication search process. The data records in the data window are picked out over fast data base accesses using binary indices from the data base. One important requirement is that the original data must not be changed. For this reason, the reformatting of the fields, included in the process of identification of customer dupes is done using additional data tables.

b) Preparations of data

Replacing local special characters

Replacing language specific special characters is essential, because the input data can contain either the special symbols or their well-known substitutes. All local special characters from used languages are represented as it is shown in table 1. A phonetic substitution takes place. In the final result all character strings can be represented into all data fields over the 7-bit ASCII Code. In this way the rules will be executed faster and they will be simplified because there is no need to handle these special characters.

Country	Character	Substitution	Country	Character	Substitution
Croatia	C	ch	Austria / Germany	Ü	ue
Croatia	Š	sh	Slovenia	c	Ch
Czech republic	C	ch	Slovenia	ž	Dz
Czech republic	Ž	dz	Malta	g	J
Austria / Germany	Ä	ae	Malta	gh	H
Austria / Germany	Ö	oe	Slovakia	ä	Ae
Austria / Germany	ß	ss	Slovakia	ž	Dz

Table 1: Special symbols in European Countries and their

conversion Legal forms substitution

The legal form is usually a component of the name of the companies. There exist different abbreviations of legal forms as well. Below are some examples of German / Austrian legal forms:

"GmbH" "GesmbH" "mbH" "m.b.H." "CO GmbH" "GmbH & Co KG" "m.b.H. & Co. KG" "GmbH & Co." "GmbH&Co.KG" "Co. KEG" "Ltd" "m.b.H.Nfg KG" "...gesellschaft mbH"

The legal form is also language-dependent, i.e. in different languages different legal forms are used (see table 2). The legal forms are standardized during the mapping process and converted as synonyms to a sample form.

Country	Company Name	Abbrev.	Country	Company Name	Abbrev.
Germany	Aktiengesellschaft	AG	Czech rep.	akciová spolecnost	A.S.
Germany	Eingetragene Erwerbsgesellschaft Professional Partnership	EEG	Czech rep.	Spolecnost s rucením omezeným	Spol S.R.O.
Germany	Kommanditgesellschaft.	KG	Czech rep.	Verejna obchodni spolecnost	V.O.S.
Germany	Offene Handelsgesellschaft. Partnership	OHG	Czech rep.	Komanditni spolecnost	K.S.
Germany	See GmbH	GesmbH	Hungary	Betéti társaság	Bt.
Germany	Gesellschaft mit beschränkter	GmbH	Hungary	Korlátolt felelosségu	Kft.

	Haftung			társaság	
Croatia	dionicko društvo	D.D.	Hungary	Közkereseti társaság	Kkt.
England	Company Limited by Guarantee	Ltd.	Hungary	Közös vállalat	Kv.
England	Private limited company	Ltd.	Hungary	Részvénytársaság	Rt.
England	Public limited company	Plc.			

Table 2: Legal Forms in European Countries

Removing of non-relevant words

Some words or parts of compound words carry not too important information content and can be removed after assignment of small penalty (e.g. only 2%) from the text. Thus, the relevant text contents have greater importance, which contribute to the duplication identification better.

Example: Comparison of "steel processing company m.b.h" with "steel processing GesmbH" gives level of similarity 1.92 equal words at maximum number of 2 words, the similarity coefficient is 96.00% = 1.92/2. The 4% penalty is result of the deletion of "society" from the first character string and the replacement of "m.b.h" and "GesmbH" from both strings to "GmbH".

c) Edit distance algorithm

String edit distance is the usual way of defining the degree of similarity between two strings. However, in order to achieve reasonable accuracy, most real problems require the use of extended sets of edit rules with associated costs that are tuned specifically to each data set. Edit distances are calculated by using table of string edit costs. This table implicitly contains the edit cost of every permutation of edit rules required to transform one string into another. The table is a two-dimensional array whose values are calculated from the upper left diagonally to the bottom right. The upper left corner is always initialized to zero since there is no cost to transform the null string into the null string. The edit distance at each point is calculated from the surrounding values to the left and above the current position. These surrounding values represent the minimum edit distance required to reach that particular position within the two strings. The edit cost to obtain the current position from one of the surrounding positions is first calculated and then added to the value at that surrounding position. This is done because an edit rule is required to move from the previous position to the current one. Performing these operations produces three edit distances. The minimum edit distance is assigned to the current position. This algorithm guarantees that the minimum edit distance will always be the value at the lower right corner.

The best-known character-based string similarity metric is Levenshtein distance, defined as the minimum number of insertions, deletions or substitutions necessary to transform one string into another [12]. Needleman and Wunsch [13] extended the model to allow contiguous sequences of mismatched characters, or gaps, in the alignment of two strings, and described a general dynamic programming method for computing edit distance. While character-based metrics work well for estimating distance between strings that differ due to typographical errors or abbreviations, they become computationally expensive and less accurate for larger strings. The vector-space model [15] of text avoids this problem by viewing strings as “bags of tokens” and disregarding the order in which the tokens occur in the strings.

In our realization the comparison algorithm implemented by Reinhard Schneider and Chris Sander [20] for comparison of protein sequences, but implemented to compare two ASCII strings is used. It was extended with including of several features and assignment of weights and bonuses as long as introducing of similarity tables (phonetic similarity, characters, located near to each other on the keyboard, etc.) as it is shown in table 3.

Delete	Penalty	Remark	Replace	Score	Remark
Phonetic similarity	-0,45	a, e, u, ...	Same Indication	1,00	
Consonants	-0,40	r, t, p, s, ...	Same type	0,20	Phon, Cons, Numb.
Numbers	-0,70	0,1,2,3,4,5,6,7,8,9	Different Type	0,15	Phon, Cons, Numb.
Blank “ ”	-0,40	“ ”	Number	0,25	2 ☒ 3
Dot “.”	-0,65	“.”	Small – Capital	0,80	“a”☒“A”, “A”☒“a”,
Comma “,”	-0,45	“,”	Adjacent keys on the keyboard	0,40	“a”, “d”, “w” or “x” instead “s”
Other	-0,20		Phonetic Similarity	0,45	For example “t”-“d”, “v”-“f”, “2”-“3”
Repeated deletion	-0,30	Deleting of symbols in a row	Similar characters string	0,20	For example “abcde”-“cdexz”

Table 3: Penalties and scores in similarity search algorithm realization

d) Sequence of the words

The measurement of the level of similarity of isolated words in two-character strings is not enough in many cases. The words in the sentence can be in different order. The algorithm must be able to find whether the replacing of the order of the words will give higher level of similarity. Our realization divides the sentence in words and investigates potential disorder also. For example, the comparison between **"Delphi automobile system"** and **"automobile systems Delphi"** gives a level of similarity 2.8 equal words from maximum number of 3 words, the similarity coefficient is 93.33% = 2.8/3. The 6.67% are lost because of penalty for the unequal order of the otherwise same words. The comparison of the isolated words is realized using the algorithm discussed above. For example, mistyping in the word of "automobile" and input as "Automibile" results in 2.78 equal words and a similarity coefficient of 92,53%.

e) Address analysis

The address field analysis is complicated because of the not clear format used. The task of the allocation of the street name and number in the data field road is realized in the following steps:

- Determining of the separate parts (words) of the address data field.
- The word which contains numbers presumably contains street number
- This word is treated separately as street number; the remaining character string should contain the street name.
- The standardisation of the street name includes standardization of similar terms, which replaced synonyms to road, place, etc. in different languages. An example of synonyms replacing in German is shown below:

1. "strasse" <---"str" "weg" "gasse" "gs" "allee"
2. "platz" <---"pl" "markt" "park" "ring" "anlage" "garten"
3. "ort" <---"dorf","bach","berg","burg","graben","siedlung","laende","staette"

f) Clustering of the data

The important idea which was applicable in the discussed approach is, that the analysis is done over several data fields (with different priority). Thus, the size of the data screen can be kept small due to implementing ‘multi-pass approach’ [3], executing several independent runs using different keys. The amount of data records to be compared is requested out from the data base. The following conditions must be fulfilled during this process:

- All relevant data fields for the dupe search are kept in the data base in their original form and in modified form, where all the preparation tasks discussed above were executed. On the modify form a set of binary indices is applied. The modified form is prepared from the original form during overnight batch preparation mode and it is used during the next day.
- An adjustable logical function is built using SQL instruction on the columns of the modified form. The logical function determines the size of the scanning window based on the importance of the data fields and using the multi-pass approach.
- It is expected that the scanning window will contain between 200 to 300 data records. If the number of data records becomes too large, then the logical function calculating the selection limit can be changed. An example of such logical function is given below:

Select row if

- at least one of the fields with priority 1 is equal to the searched value or
- at least two fields with priority 2 are equal or
- at least three fields with priority 3, 4, 5 or 6 are equal to the corresponding value.

The total similarity is calculated based on the similarity levels in every column using formula like the one below:

Total Similarity = 1 – ? (1 - SimilarityCoefficient(Field i))

3. EXAMPLE OF POTENTIAL DUPLICATION

An example of search of potential customer dupes is given in the following example (table 4). The column priority contains importance of the fields for the total result. The result from the comparison of the individual fields is registered in the column similarity. The total result contains the result of the weight formula discussed above.

Data Field	Sample record	Comparison set	Priority	Similarity Coefficient
First Name	Anna	Anna	1	100,00%
Name	Mustermann	Müller	1	6,00%
Customer ID	200100195485	300200946734	2	3,00%
Birth Date	22.08.1981	22.08.1981	2	100,00%
Zip Code	1234	1234	4	100,00%
City	Musterdorf	Musterdorf	4	100,00%
Address	Parkstrasse 24	Parkstrasse 24	3	100,00%
Register Number	456123	456123	2	100,00%
			Total Similarity	71,65%

Table 4: Duplication Example

Flexible assignment of weights and using of rules for non-linear addition of the coefficients are used. AI aggregates values for data fields which meaning is overlapped (post code and city for example).

4. TECHNICAL REALIZATION AND CONFIGURATION

The technical realization of the duplicate detection for the discussed customer database is based on the following prerequisites:

The data fields of a potential duplicate data are inputted into the system via Web interfaces or via data import. After converting of the data in the modified form discussed above, the logic for the duplicate detection in the application server is activated.
A set of dynamically-created SQL requests is sent to the database server, picking out the data records from the scanning window. This operation uses specially created binary indices which met the substantial performance requirements of this task. The data records in their modified form are returned.
The comparison of the individual records with the potential duplicate one is realized in a module built using the CLIPS rule based engine. All methods and rules for the comparison of the individual fields and for the total comparison are developed as CLIPS rules.
Algorithmic parts are implemented directly in Java and bound to CLIPS. Using of CLIPS rules provides maximal flexibility and the ability for further adjustments and development of the functionality.
The result from the comparison of the data records is a similarity coefficient, in form of percentage (0 <= SC <= 100). Categorization as duplication can take place manually or automatically based on the list of the similarities and the threshold value. A manual confirmation of the duplications list is necessary for the avoidance of errors in any case. The configuring of the process of the identification of duplicates is very easy because the definition of the rules is stored in plain language in CLIPS-module files. The engine loads the script file and activates the rules.
The penalty and bonus values of "Edit Distance" algorithm, discussed above are also subject of adjustment using test data. They are also set in CLIPS file, so the change of these values without any changes in the module is possible.
Some other configurable settings include the keyboard layout for adjacent keys and the list of phonetically similar characters, threshold values and penalties for recognition and replacement of synonyms and deletion of non-relevant words, lists for corresponding legal forms and their abbreviations, lists of standardized address formats, address parts and their abbreviations in different languages, etc.

CONCLUSIONS AND FUTURE WORK

The discussed module was developed and tested using sample data. The following conclusions were made:

- The preparation tasks, involving data modification must be performed on the database server. This approach gives better performance, compared to the one with performing transformations using the CLIPS engine.
- To reach the needed performance in the real-time, the additional databases and the binary indices must be prepared during the previous night batch mode.
- Using of CLIPS rules provides maximum flexibility for adjustment of the algorithm.
- The edit distance algorithm must be implemented as a separated module in C to achieve maximum performance.
- Future development concerns practical implementations in following directions:
- Analysis and adjustments of the score and penalty table for the edit-distance algorithm. Collecting of nomenclature data for proper handling of input from other languages (French, Italian) and alphabets (Cyrillic, etc.)

REFERENCES

[1] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. Science, 130:954–959, 1959.
[2] W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census, Wachington, DC, 1999.
[3] M. A. Hernandez and S. J. Stolfo. The merge/purge problem for large databases. In Proceedings of the 1995 ACM SIGMOD, pages 127–138, San Jose, CA, May 1995.

[4] A. E. Monge and C. P. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In Proceedings of the SIGMOD 1997 Workshop on Research Issues on Data Mining and Knowledge Discovery, pages 23–29, Tuscon, AZ, May 1997.
[5] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton, Alberta, 2002.
[6] W. W. Cohen, H. Kautz, and D. McAllester. Hardening soft information sources. In Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000), Boston, MA, Aug. 2000.
[7] A. K. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000), pages 169–178, Boston, MA, Aug. 2000.
[8] W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton, Alberta, 2002.
[9] D. Gusfield. Algorithms on Strings, Trees and Sequences. Cambridge Univ. Press, NY, 1997.
[10] R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. ACM Press, NY, 1999.
[11] S. Tejada, C. A. Knoblock, and 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-2002), Edmonton, Alberta, 2002.
[12] Bilenko M., Mooney R. Adaptive Duplicate Detection Using Learnable String Similarity Measures In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2003), Washington DC, pp.39-48, August, 2003
[13] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequences of two proteins. Journal of Molecular Biology, 48:443–453, 1970.
[14] Ristad, E. S., and Yianilos, P.N. Learning string edit distance. Research Report CS-TR-532-96, Department of Computer Science, Princeton University, Princeton, NJ, October 1996, Revised (October, 1997).
[15] J.J. Zhu and L.H. Ungar. String Edit Analysis for Merging Databases. In proceedings of International Conference on Knowledge Discovery and Data Mining(KDD) 2000.
[16] A. K. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000), pages 169–178, Boston, MA, Aug. 2000.
[17] A. Monge and C. Elkan. The field-matching problem: algorithm and applications. In Proceedings of the 2nd Conference on Knowledge Discovery and Data Mining, August 1996.
[18] A. Monge and C. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In The proceedings of the SIGMOD 1997 workshop on data mining and knowledge discovery, May 1997.
[19] I. P. Fellegi and A. B. Sunter. A theory for record linkage. Journal of the American Statistical Association, 64:1183–1210, 1969.
[20] C. Sander and R. Schneider, "Database of homology-derived protein structures and the structural meaning of sequence alignment," Proteins, vol. 9, no. 1, pp. 56--58, 1991.

ABOUT THE AUTHORS

Eurorisk Systems Ltd.
31, General Kiselov Str.
9002 Varna, Bulgaria

Plamen Paskalev
E-Mail: ppaskalev at eurorisksystems dot com
Dr. Anatoliy Antonov
E-mail: antonov at eurorisksystems dot com