Journal of Engineering and Technology for Industrial Applications

ITEGAM-JETIA

Manaus, v.6 n.25, p. 4-15, Sep/Oct, 2020
DOI: https://doi.org/10.5935/jetia.v6i25.685

ISSN ONLINE: 2447-0228

RESEARCH ARTICLE OPEN ACCESS

A SYSTEM PROPOSAL FOR AUTOMATED DATA CLEANING ENVIRONMENT

Carlos Roberto Valêncio[1], Toni Jardini[2], Victor Hugo Penhalves Martins[3], Angelo Cesar Colombini[4] and Márcio Zamboti Fortes*[5]

[1, 2, 3] São Paulo State University - UNESP. São Jose do Rio Preto–São Paulo, Brazil.
[4, 5] Federal University of Fluminense - UFF. Niterói–Rio de Janeiro, Brazil.

[1] http://orcid.org/0000-0002-9325-3159 [2] http://orcid.org/0000-0002-0686-6791 [3] http://orcid.org/0000-0003-4507-0128
[4] http://orcid.org/0000-0002-8906-4128 [5] http://orcid.org/0000-0003-4040-8126

Email: *mzamboti@id.uff.br

ARTICLE INFO

Article History
Received: August 12th, 2020
Accepted: October 19th, 2020
Published: October 30th, 2020

Keywords:
Data cleaning,
Semantic,
Knowledge Discovery in Databases (KDD),
Data analysis.

ABSTRACT

One of the great challenges to obtaining knowledge from data sources is to ensure consistency and non-duplication of stored information. Many techniques have been proposed to minimize the work cost and to allow data to be analyzed and properly corrected. However, there are still other essential aspects for the success of data cleaning process that involve many technological areas: performance, semantic and autonomy of the process. Against this backdrop, we developed an automated configurable data cleaning environment based on training and physical-semantic data similarity, aiming to provide a more efficient and extensible tool for performing information correction which covers problems not yet explored such as semantic and autonomy of the cleaning implementation process. The developed work has, among its objectives, the reduction of user interaction in the process of analyzing and correcting data inconsistencies and duplications. With a properly calibrated environment, the efficiency is significant, covering approximately 90% of inconsistencies in the database, with a 0% percentage of false-positive cases. Approaches were also demonstrated to show that besides detecting and treating information inconsistencies and duplication of positive cases, they also addressed cases of detected false-positives and the negative impacts they may have on the data cleaning process, whether manual or automated, which is not yet widely discussed in literature. The most significant contribution of this work refers to the developed tool that, without user interaction, is automatically able to analyze and eliminate 90% of the inconsistencies and duplications of information contained in a database, with no occurrence of false-positives. The results of the tests proved the effectiveness of all the developed features, relevant to each module of the proposed architecture. In several scenarios the experiments demonstrated the effectiveness of the tool.

Copyright ©2016 by authors and Galileo Institute of Technology and Education of the Amazon (ITEGAM). This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

I. INTRODUCTION

Storage of information has become increasingly larger and more frequent, since data may contain valuable information [1]. In order to obtain correct information from all possible or convenient sources, it is necessary to ensure that the analyzed data are integrated consistently and correspond to the reality of the information. This warranty is only possible if there is a data pre-processing; a step where the information stored in different sources is integrated and analyzed to detect inconsistencies, physical or semantic duplications, standardization and corrections, etc. One of the stages of data pre-processing is called data cleaning, which develops computational techniques for handling data issues and inconsistencies from the same source or even distributed sources. Only after that process is the data suitable for analysis and correlation [2].

Although there are several suggestions and techniques in literature to solve the problem of inconsistency and data duplication, enough tools that contemplate efficiency and effectiveness cannot yet be found, and their cost of processing is quite high and results do not always guarantee that information is properly treated and corrected.

This paper presents a data cleaning environment that addresses needs that are extremely important but previously poorly explored, to provide a tool which is effective and independent as possible of human interaction to implement a large data sources cleaning process, providing an extensible tool which considers possible extensions of new algorithms, techniques, rules and settings, besides being portable to different relational database management systems.

II. BACKGROUND

A data cleaning process should detect and remove errors and inconsistencies from one or more information sources to improve data quality. This process is needed in order to analyze and extract useful knowledge from a repository, a file, a data set of databases, as well as is essential for consistent integration of heterogeneous data sources and one of the data pre-processing steps for data mining. Problems and inconsistencies that are found mainly arise due to spelling errors, invalid data or lack of data during information entry in a file or data repository [3].

Systems of heterogeneous data sources such as data warehouses require a comprehensive treatment of information, especially to support data cleaning. In this type of architecture, data is loaded, huge amounts of information are continuously updated from several repositories and the probability of some sources containing "dirty data" is high. As example of this complexity data analysis, we can indicate the research of a herbarium in [4]. Due to the wide variety of possible inconsistencies and the large volume of data, the data cleaning process is considered one of the most complex processes in order to obtain knowledge. Even nowadays, computer systems provide only limited support for data cleaning because they focus on data transformation for translation and integration schemes [5].

A data cleaning process must satisfy several requirements. First of all, it should detect and remove any mistakes and inconsistencies in the information stored in both single and integrated multiple sources. The approach should be supported by tools that minimize the manual effort and are generic and easily extensible to cover new sources of information. Furthermore, data cleaning should not be done in isolation, but together with related data schema transformations based on comprehensive metadata. Mapping functions for cleaning and other data transformations are specified declaratively and can be reusable for other data sources, as well as for query processing. An infrastructure workflow must reliably and efficiently support all stages of processing data from multiple sources and large data sets. According to [3], data cleaning process may involve the following phases: data analysis, conflict treatment, conflict treatment, transformation, verification, transformation and duplicate elimination tasks.

II.1 SIMILARITY AND DUPLICATE DETECTION ALGORITHMS AND TECHNIQUES

Determining matching cases is typically a high cost operation for large databases. Algorithms for character-based similarity detection aim to treat mainly misspellings or data entry. The main algorithms cited in literature include the Smith Waterman Distance, Affine Gap Distance [5], Edit Distance [6-7], Jaro Distance Metric [8], Q-Gram Distance [9-11], Hamming Distance [12-13] and Jaccard Coefficient [14-15].

Algorithms for token-based similarity detection aim to treat mainly misspellings or typing data as character-based, but with the difference of treating compound words or set of words. The main algorithms that address these cases are: Cosine Similarity [16-18], Atomic Strings [19], WHIRL [20] and Q-Grams with tf.idf [21].

Two techniques previously discussed deal similarities representative of characters composing one or a set of words. However, depending on each language, a set of different characters can be phonetically similar, even if they are physically similar, i.e. similarity of characters or tokens. The main algorithms with this approach are: Soundex [22] Metaphone and Double Metaphone [23], NYSIIS - New Your State Identification and Intelligence System [24], ONCA - Oxford Name Compression Algorithm [25] and PSI - Phonetic Similarity Identification [2].

II.2 TOOLS

Over the last years, some data cleaning tools have been presented and research groups have contributed presenting techniques and free software packages which can be used for duplicate records detection. The main tools presented in literature include WHIRL [26], Flamingo Project5 [27], BigMatch [28], Database Cleaner [29], WizSame [5], Febrl Syste3 - Freely Extensible Biomedical Record Linkage [30-31], TAILOR [32] and SmartClean [33].

II.3 REVIEW OF DATA CLEANING RESEARCH

Published researches in recent years in the data cleaning area have covered some important perspectives such as new algorithms, frameworks and tools that include a combination of several techniques, involving detection of duplicate and statistics in order to try to improve information quality with low cost and minimal user interaction in data cleaning process. Some samples work such as [34-41], propose frameworks in order to offer extensibility and efficiency to the data cleaning process.

Some other researches propose new techniques and algorithms for duplicates detection. Ciszak [42] proposes an algorithm based on data correlation methodology (data mining) to identify and correct physical or semantic duplicate information. Qian [43] introduces the need for VGI data cleaning. Wang [44] presents an algorithm for data cleaning using outlier data detection technique. Okita [45] presents an algorithm for automated data translators cleaning. Bertossi et al. [46] presents an example of data cleaning process using the concept of corresponding dependencies as a procedure for detecting duplicates. Chaturvedi [47] proposes a method that selects a set of data records which, when used to create the data cleaning-based rule model, may include the maximum number of records. Prasad [48] present a tool to improve data quality that identifies variations and synonyms of a given entity present in the data.

II.4 CONTRIBUTIONS OF THIS WORK

The most significant contributions of this work refer to the developed tool that, without user interaction, is automatically capable of analyzing and eliminating 90% of the inconsistencies and duplications of information contained in a database, with no occurrence of false-positives. Moreover, the data cleaning environment is extensible and portable, allowing it to be easily upgraded and enhanced with new algorithms, techniques, dictionaries, language, etc.

Features, techniques and approaches for the data cleaning process with various tools and frameworks available in literature and the market are summarized in Table 1, where the character "x" indicates the present of the corresponding feature. It is worth noting that the developed data cleaning environment addresses many aspects still little explored by these works in the state of art, thus confirming its contribution.

Table 1: Comparison of features contemplated by diverse data cleaning tools.

	This work	[26]	[27]	[31]	[2]	[4]	[48]	[47]	[46]	[29]	[39]	[40]	[44]
Duplicate Detection	x	x	x	x	x	x	x	x	x	x	x	x	x
Detection and Transformation	x	x	x			x		x		x	x	x	
3 or more Ad-hoc Algorithms	x			x		x		x	x	x	x	x	
Special Character Treatment	x	x	x	x	x	x	x	x		x	x	x	
Phonetics	x			x	x					x			
Multi-Language Phonetics	x				x								
Stopwords	x												
Multi-Language Stopwords	x												
Semantic	x						x			x			
Multi-Language Semantic	x												
Intelligent Approach and Training	x							x	x				
Manual Data Cleaning	x	x	x			x		x		x	x	x	
Semi-Automated Data Cleaning	x												
Automated Data Cleaning	x												
Extensibility	x					x				x	x	x	
Portability	x					x				x	x		

Source: Authors, (2020).

III. COMPARISON OF FEATURES CONTEMPLATED BY DIVERSE DATA CLEANING TOOLS

The data cleaning process can be basically divided into two main activities: analysis and transformation. For both cases, it needs a large computational effort to scan whole database, find duplicate information and correct them.

In the analysis stage, physical and semantic approaches are needed, since in many cases, words with similar duplication spellings do not match, or otherwise, different words physically may represent the same real-world entity, which would represent duplicity or dirt information in the database.

There are many techniques presented in literature for detecting duplicate information based on its physical resemblance (spelling) and phonetics, but it has still been little explored from a semantic perspective. To cover semantic term variations is difficult, since the semantics within a language vary not only in its formal construction, with synonyms, but vary in the same language meanings in different regions, culture and even in the context in which they are used.

Another important point to be explored corresponds to the frequency that the database needs to be cleaned. Most databases have real-world data constantly inserted, resulting in daily duplications and dirt. This means that a cleaning process has to be performed again on the following day, often a few hours later. It shows that a significant portion of the problems are recurrent, i.e., instances of dirt that had previously been detected and treated.

Available algorithms and tools offer partial solutions, attacking some of the problems, with advantages and disadvantages, but there is still work to be done in order to solve problems in analysis, processing and providing a truly effective and efficient data cleaning solution.

The developed data cleaning environment offers efficiency in the whole data cleaning process. The proposed architecture includes modules for specific treatment of analysis and transformation stages, which will be described in detail in the following sections of this paper.

III.1 DATA CLEANING ENVIRONMENT OVERVIEW

The work was developed using C++ development framework supported by Qt with modules for SQL data manipulation language. One of the advantages of the proposed tool is related to its portability and expandability, since it was modularly constructed and can be portable to different DBMS.

Some of its features are directly dependent on the used DBMS, but possibilities are offered to adapt to new database mechanisms that support foreign keys and referential integrity. It is possible to use a simpler database manager system, but some of environment features can still be used. Repositories as auxiliary files and embedded database [49] for training database, stop words database and synonyms database are also used.

Besides all the offered techniques organized into modules, the environment offers several useful features aimed at the infrastructure of the cleaning process such as connecting database through the GUI – Graphical User Interface framework, database cloning and database backup restore. An application of the GUI Software is presented by [50].

III.2 DATA CLEANING ENVIRONMENT ARCHICTETURE

The proposed environment is organized into modules that support the processes of analysis and data transformation. Global auxiliary modules can be used in each step together with the specific modules, supporting both analysis and processing. Its architecture, illustrated in Figure 1, provides structured and independent modules, allowing the user the possibility to work flexibly and perform tasks ranging from standardization, detection and processing data manually using all the features offered with the fully automated process.

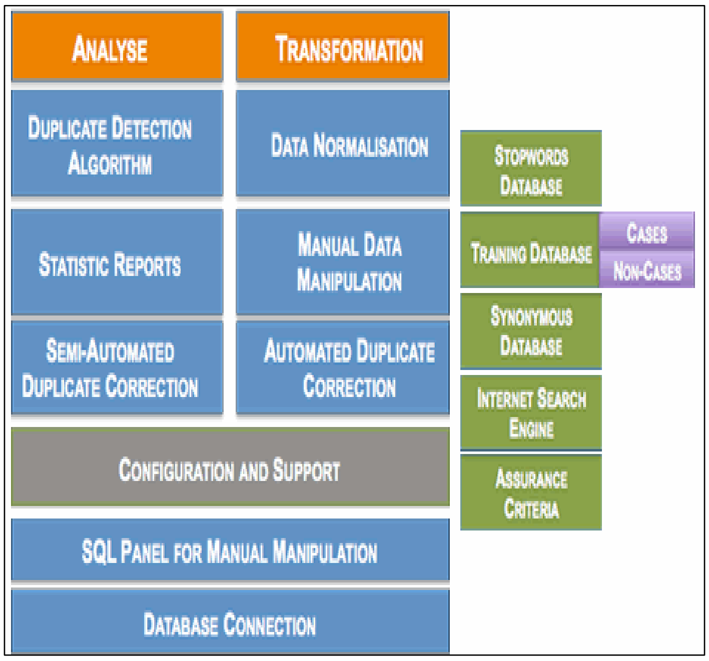


Figure 1: Architecture of developed data cleaning environment. Source: Authors, (2020).

To support the analysis process, there are modules that include duplicate detection algorithms, statistical reports and semi-automated correction of duplicates. For the transformation process, specific modules are contemplated: data normalization, data manipulation and automation of duplicates correction.

The proposed architecture also includes several global modules that are used in support of both the analysis and transformation processes: SQL panel for manual data manipulation; DBMS connection settings and support, which involves the following sub-modules: Stop words database; Training database (divided into duplicate cases and non-duplicate cases); Synonyms database; Internet Search Engines; Assurance Criteria Rule Settings.

III.3 DATA CLEANING ENVIRONMENT ARCHICTETURE: DATA ANALYSIS

There are three modules parts of the data analysis process whose goals include providing analysis and detection of problems, duplicates and data inconsistencies.

The first module provides several algorithms for detecting duplicate and inconsistent data. In the second module there are statistical reports available generated by automated analysis performed by the tool, allowing the user to have knowledge of how dirty or inconsistent the database is in accordance to defined and configured criteria. The third module, named "semi-automated duplicates correction", contains techniques and algorithms that suggest to the user which of the tuples identified as duplicity is correct, based on database historical references criteria, featuring the environment as an intelligent and trainable tool and thus assisting the user in making the decision to perform the correction and data cleaning.

III.4 ALGORITHMS FOR DETECTING DUPLICATE AND INCONSISTENT DATA

There are several variations and algorithms to detect duplicates in databases, each aiming to attack a specific problem. In order to make the developed tool as applicable as possible, main algorithms presented in literature that involve techniques of similarity of characters, and phonetic tokens similarity were implemented.

The environment also provides the user with the option to create new similarity detection algorithms like attributes through the system interface, which support JavaScript language, on a panel that validates syntax of the language, illustrated in Figure 2, and allows total flexibility for the user to apply specific techniques beyond those already offered, according to user's needs. This is one of the ways that the data cleaning environment includes expandability, but it is also possible that other algorithms be included in C++ language, as the natives of the environment. When the analysis and detection of duplicity and inconsistency process is started, one of the algorithms implemented in the environment must be chosen.

III.5 SYNONYMS DATABASE

Synonyms functionality was developed in order to cover the lack of semantics in the process of detecting inconsistencies and duplicate information in the database. With a synonyms database configured by the user, a scan is performed for similarity detection and it is done not only through the techniques of the chosen algorithm, but also includes checks of semantically equivalent terms, such as "helper", "help" and "collaborator".

Depending on the context, words can have the same meaning, but would not be easily detected unless there was a dictionary of equivalent terms. Another example would be a database of accidents that diagnoses animal bites. Generally it reports that the victim was bitten or stung by a specific animal, like a dog, snake, etc., but statistically it is only important to obtain accidents that were caused by animals. In this case, a list of animals can be included in the synonyms database and when any of them is detected, the system automatically converts them to "animal", and the cleaning will be performed in an automated process.

III.6 MULTI-LANGUAGE DATABASE

A multi-language database allows the user to load translation dictionaries for multiple languages in order to transform the environment into a tool that can detect duplicates of information regardless of the language in which it is. Once it is properly configured, similar data attributes or semantically identical attributes will be detected even if they are described in different languages.

For example, a table containing data of animals would be detected as tuples with similar attributes, dog, "cachorro", "perro", "cão", etc. With this functionality, the language barrier between data repositories is broken, making it feasible to prepare an integrated base of different languages for knowledge extraction, after applying the cleaning process in which all information would be cleaned (standarded) to a language key.

III.7 MULTI-LANGUAGE PHONETIC DETECTION

For each of the techniques and algorithms contemplated by the environment, an option that considers the phonetics of words in the data analysis process is also offered. Thus, each technique considers the phonetic differences among the possibilities of writing the same word to scan each tuple of the table in similarity searches and overlaps, caused either by an incorrect entry or misspelling which are highly recurrent on systems with data entry by users, e.g. “S” instead of “SS”, “CH” instead of “X”, etc. The implemented algorithm includes the following Portuguese language phonetic conversions, represented in Table 2.

For data cleaning environment scalability, the phonetic transcriptions are stored into the embedded database tool and can be improved with new rules, allowing phonetic rules insertion of any languages besides Portuguese and English.

Table 2: Portuguese language phonetic transcriptions.

Letter/Syllable	**Phoneme**	**Condition**
X	Z	If not in the beginning or at the end and is surrounded by vowels, have sound Z
Ç	S	
CH	X	
CE	SE	
CI	SI	
CA	KA	
CO	KO	
CU	KU	
GE	JE	
GI	JI	
PH	F	
QU	K	
RR	R	
SS	S	
S	Z	If not in the beginning or at the end and is surrounded by vowels, have sound Z
Z	S	If is the last position, have sound S
W	V	

Source: Authors, (2020).

III.8 SEMI-AUTOMATED DUPLICATES CORRECTION

One of the biggest difficulties for the user in a data cleaning process is to detect two or more similar attributes that actually correspond to the same entity. Often, similar words do not match semantically and the conversion of one to another may generate more database inconsistency. This leads to some limiting factors for cleaning process implementation, for example, the need for a data specialist to make decisions and great efforts in manual searches to check which detected information is the correct one and if indeed these are the same entity. It is important to emphasize that this situation is constant and many data cleanings must be repeatedly performed since the input is inconsistent in most cases and impossible to be completely avoided.

With the constant realization of data cleaning processes, the cost based on historical cleaning can be minimized. An automated analysis was developed based on the quantity of times each tuple is referenced by another if two or more tuples are detected as similar records according to the chosen similarity detection algorithm. The system checks the quantity of times each one is referenced by other tables and, therefore, indicates to the user the tuple most likely to be correct. It is worth thinking that if a tuple is more referenced than another one, the chances of it containing the correct information is greater, since after several cleanings have been done and where other tuples have also been converted to it, their referenced quantity is higher.

The process called semi-automated duplicates correction can be described with the following pseudo-algorithm in Figure 2.

```
Semi-automated duplicates correction

1. Check similarity based on chosen algorithm;
2. For each group of detected similar tuples:
       a. Verify if the tuples had been detected earlier
             i. If yes, check the history of what was
                done,i.e., whether cleaning was done, keep
                the correct tuple and go to step 4;
       b. Check each tuple for how referenced it was by other
          tables in the database;
       c. Keep the total amount;
3. Check between tuples of the group for the one that has the
   biggest amount;
4. Show the user which is the most referenced or historically
correct tuple.
```

Figure 2: Semi-automated duplicates correction algorithm.
Source: Authors, (2020).

This functionality is useful for the manual cleaning process, mainly because it assists the user in making decisions based on historical cleanings performed previously by the amount of tuples and referenced in the current landscape of the database. Even in cases where two attributes are substantially similar, the user has a tool to aid in deciding whether, for example, a company name is written in one way or another, according to the quantity in which they are referenced as it is expected that, in a database constantly cleaned, the correct name is referenced more than a new inconsistent data that has been recently inserted, or a duplicity which has been treated and corrected earlier.

IV. DATA CLEANING ENVIRONMENT ARCHITECTURE: DATA TRANSFORMATION

The data transformation step basically consists in applying the necessary corrections and changes to the database. The data cleaning environment provides a set of functionalities that offer many possibilities of user interaction to make changes and corrections.

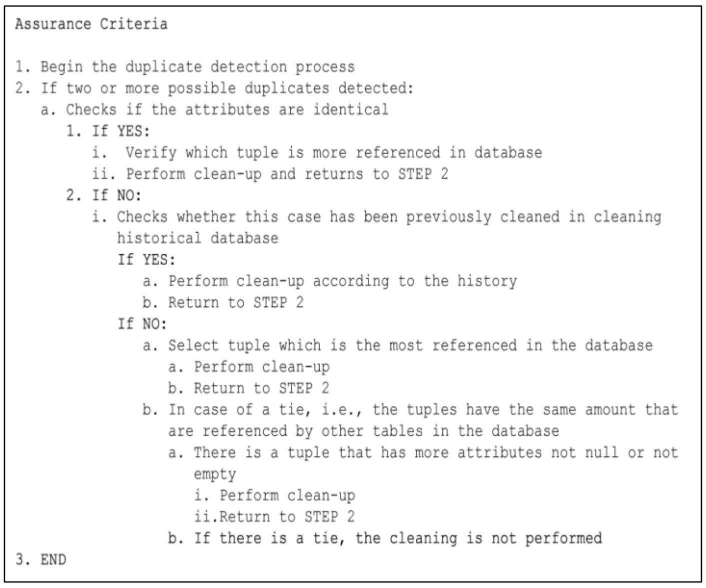
IV.1 AUTOMATION OF DUPLICATES CORRECTION

The main objective for developing this work consisted in the need of avoiding as much user interaction in the data cleaning process as possible, since large relational databases contain millions of records organized in tables and columns, and a human analysis for duplicity and inconsistency detection is impossible without the appropriate computational support.

Using all the analysis techniques previously described, it is possible that some, if not all the database is cleaned and handled automatically. To do that, a module called Automation of Duplicates Correction was developed that consists of an automatic application analysis and database table transformation, without the need for user interaction during the data cleaning process.

The user needs only to indicate which table should be cleaned, choose the type attribute, the similarity detection algorithm that will be used and check the option Automatic Cleaning. Once the mechanism is triggered, the environment will do the work of scanning, analyzing and correcting duplicate and inconsistent data.

In order to avoid the automated correction processes converting tuples that are not actually duplicates, despite the high physical similarity or semantic, a set of rules and conditions that guarantee a right automatic correction was developed, called Assurance Criteria. Their terms and conditions are described by pseudo-following algorithm in Figure 3.



```
Assurance Criteria

1. Begin the duplicate detection process
2. If two or more possible duplicates detected:
   a. Checks if the attributes are identical
      1. If YES:
         i.  Verify which tuple is more referenced in database
         ii. Perform clean-up and returns to STEP 2
      2. If NO:
         i. Checks whether this case has been previously cleaned in cleaning
            historical database
            If YES:
               a. Perform clean-up according to the history
               b. Return to STEP 2
            If NO:
               a. Select tuple which is the most referenced in the database
                  a. Perform clean-up
                  b. Return to STEP 2
               b. In case of a tie, i.e., the tuples have the same amount that
                  are referenced by other tables in the database
                  a. There is a tuple that has more attributes not null or not
                     empty
                     i. Perform clean-up
                     ii.Return to STEP 2
                  b. If there is a tie, the cleaning is not performed
3. END
```

Figure 3: Assurance criteria algorithm for automated duplicates correction.
Source: Authors, (2020).

The conditions contemplated by Assurance Criteria contribute towards corrections not performing automatic errors. Obviously, the process is highly dependent upon the chosen algorithm and that settings for the tuple identified as potential duplicates are indeed equivalent. The environment also provides an additional check before all the fixes are actually performed.

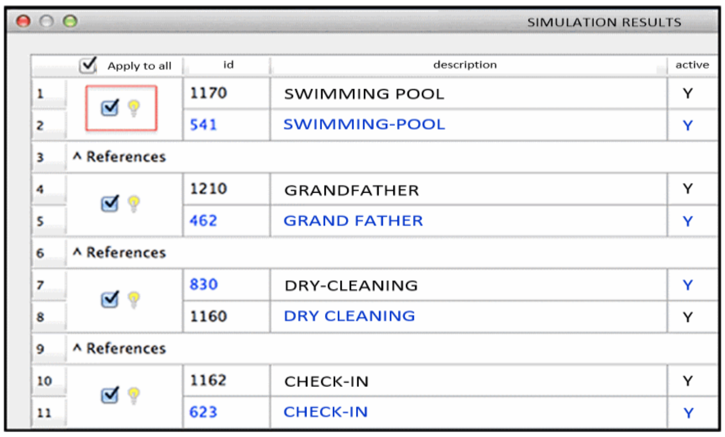
IV.2 DATA CLEANING ENVIRONMENT TRAINING

In every completed data cleaning process, be it manual, semi-automated or automated, the data cleaning environment takes care of storing cases in a training database where clean-ups were done. In the manual and semi-automated options, all converted tuples are stored as historical and a positive-case, even when the event is recurring, the system will automatically suggest the correct tuple. For the automatic cleaning option, all tuples will be automatically converted in accordance with the historical database.

Likewise, the training is performed for negative cases that are stored in the negative historical database in which the algorithm, according to the criteria and parameters set by the user, indicates possible replication but which, although similar, do not represent duplicates. In these cases, it is possible to indicate that this is a false-positive and, once appointed, the recurrent cases are automatically ignored by the tool.

In the case of false-positives, interaction is required from the user to train the tool, indicating which cases should be treated as non-duplicates. With the support of automated clean-up, this analysis can be quick, since the environment generates a report of all possible cases found so that the user can mark them as false-positives. In all labeled cases, history will be recorded and in future clean-up recurrences will be automatically discarded by the environment.

The data cleaning environment training process of false-positives is illustrated in Figure 4. The selected lamp icon indicates that the event identified by the tool is correct, i.e. it is a duplicate. In cases of non-duplicates, the user must click on the lamp, which will be unmarked, and the case will be automatically stored in the database history as a non-duplicate. As the tool gets trained, the cleaning of a specific database becomes increasingly reliable and rapid, besides making the automated process very convenient because the cases are handled automatically to be more reliable.



SIMULATION RESULTS

	✓ Apply to all	id	description	active
1	✓	1170	SWIMMING POOL	Y
2		541	SWIMMING-POOL	Y
3	^ References			
4	✓	1210	GRANDFATHER	Y
5		462	GRAND FATHER	Y
6	^ References			
7	✓	830	DRY-CLEANING	Y
8		1160	DRY CLEANING	Y
9	^ References			
10	✓	1162	CHECK-IN	Y
11		623	CHECK-IN	Y

Figure 4: Training for cases of false-positive.
Source: Authors, (2020).

V. EXPERIMENTS AND RESULTS

The SIVAT – Sistema de Informação e Vigilância de Acidentes de Trabalho (Labour Accidents Vigilance System) system database to manage information about labour accidents in São José do Rio Preto city, state of Sao Paulo, Brazil and the region, totaling more than one hundred cities was used in all performed experiments in this work. It currently has about 90 thousand registered notifications of labour accidents collected by CEREST – Centro de Referência em Saúde do Trabalhador (Labourer Health Reference Centre), responsible for managing and apply actions involving the prevention and investigation of labour accidents.

V.1 COMPARATIVE EXPERIMENTS

This section presents and discusses the results obtained from experiments conducted in the data cleaning process, in order to prove the efficacy of cleaning using the data cleaning environment developed in this work. The table that was used was the “Machine Involved in the Labour Accident” which had a total of 639 records. A manual analysis was performed on all table data and it was found that there were 473 non-duplicate records, or approximately 27% of the table showed inconsistencies or duplicate data.

It is important to highlight that, unlike the results reported by most studies in literature, whose main purpose is to highlight the effectiveness of techniques to detect duplicates of valid cases, this work also presented the percentage of false-positives, i.e., the rate of incorrect duplicates that were detected. Data cleaning process is costly and if the amount of false-positives is high, the difficulty for the user to analyze and make the necessary corrections increases significantly.

It is also noteworthy that the proposal to automate the cleaning process must provide for the treatment of false-positives, since incorrect changes could generate more inconsistencies in the database instead of providing an adequate treatment of corrections and cleaning.

Experiments were carried out by applying manual / semi-automated and automated data cleaning, using the duplicate detection algorithm Q-Gram, with parameter q=3, with the percentages of similarity 90%, 80%, 75%, 70% and 65%. From some preliminary tests it was observed that above 90% similarity, false-positives are not detected and, below 65%, the results presented by the environment are not very relevant.

The application process for cleaning experiments was organized into 6 stages. Functionality or technique to verify the improvement of inconsistencies detection in the database was added at each step.

In the first step, only duplicate detection Q-Gram was applied to the algorithm to verify the percentage of detections that were obtained. Additions to the subsequent steps were: (2) treatment of special characters, normalization and stemming, (3) treatment of word phonetics, (4) stop words treatment and removal, (5) synonyms and multi-languages databases and to the last step (6) the cleaning was done with the previously trained environment. In each step, the percentage of detected false-positives was also measured.

V.2 MANUAL AND SEMI-AUTOMATED DATA CLEANING APPLICATION

As it can be seen from the results shown in Table 3, with a rate of similarity of 90%, 37 inconsistencies were detected with the application of the Q-gram algorithm, corresponding to 22% of all inconsistencies and duplicates present in the database. As refinement techniques are added, the number of presented inconsistencies significantly increases. Interestingly, with the use of synonyms in the cleaning process, the efficiency increased 71% and 75% of these inconsistencies were detected. With the trained environment, the percentages rose to 72% and 78%.

Table 3: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 90% similarity.

Q-Gram, q=3, 90%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	37	0	0%	22%
Special Characters	37	0	0%	22%
Phonetic	42	0	12%	25%
Stop words	58	0	36%	35%
Synonymous	127	2	71%	75%
Training	130	0	72%	78%

Source: Authors, (2020).

With a similarity factor of 80%, as shown in Table 4, 53 inconsistencies were detected with the application of Q-gram algorithm, but 6 of these were false-positives. The difference, namely 47 cases corresponded to 28% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of displayed inconsistencies significantly increased, representing a total of 139 cases detected and the number of false-positives decreased. With the use of synonyms in the cleaning process, the efficiency increased by 62%, and 80% of all the inconsistencies were detected. With the trained environment, the efficiency reached 83%. It is emphasized that, with the trained tool, no false-positives were presented.

Table 4: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 80% similarity.

Q-Gram, q=3, 80%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	53	6	0%	28%
Special Characters	54	5	2%	29%
Phonetic	58	4	9%	32%
Stop words	73	3	27%	42%
Synonymous	139	5	62%	80%
Training	139	0	62%	83%

Source: Authors, (2020).

With similarity factor of 75%, as shown in Table 5, 65 inconsistencies were detected with the application of the algorithm Q-Gram, but 8 of them were false-positives. The difference, namely 57 cases correspond to 34% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of presented inconsistencies further increased, for a total of 143 cases detected and the number of false-positives decreased. With the use of synonyms in the cleaning process, the efficiency increases by 56%, and 83% of these inconsistencies were detected. With the trained environment, the efficiency reached 86%. Same as with the experiment with similarity factor 80% with the trained tool, no false-positives were presented.

Table 5: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 75% similarity.

Q-Gram, q=3, 75%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	65	8	0%	34%
Special Characters	67	9	0%	35%
Phonetic	70	9	4%	37%
Stop words	83	8	19%	45%
Synonymous	147	9	54%	83%
Training	143	0	53%	86%

Source: Authors, (2020).

With a percentage similarity of 70% as shown in Table 6, 74 inconsistencies were detected with the application of Q-gram algorithm, but 11 of them were false-positives. The difference, namely 63 cases corresponded to 38% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of presented inconsistencies rose, representing a total of 143 cases detected and the number of false-positives also decreased. With the use of synonyms in the cleaning process, the efficiency increased by 50% and 83% of all the inconsistencies was detected. With the trained environment, the efficiency reached 86%. Same as with the experiments with similarity factor 90%, 80% and 75%, with the trained tool, no false-positives were presented.

Table 6: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 70% similarity.

Q-Gram, q=3, 70%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	74	11	0%	38%
Special Characters	75	11	0%	38%
Phonetic	78	10	4%	41%
Stop words	85	11	12%	44%
Synonymous	148	10	49%	83%
Training	143	0	48%	86%

Source: Authors, (2020).

Finally, with a percentage similarity of 65% as seen in Table 7, 94 inconsistencies were detected in the application of the algorithm Q-Gram, but 14 of them were false-positives. The difference, namely 80 cases corresponded to 48% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of presented inconsistencies rose, representing a total of 153 cases detected, but the amount of false-positives nevertheless increased until the environment was trained. With the use of synonyms in the cleaning process, the efficiency increased by 42%, and 84% of these inconsistencies were detected. With the trained environment, the efficiency reached 92%.

Table 7: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 65% similarity.

Q-Gram, q=3, 65%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	94	14	0%	48%
Special Characters	97	21	0%	46%
Phonetic	96	17	-1%	47%
Stop words	103	19	6%	50%
Synonymous	161	20	40%	84%
Training	153	0	37%	92%

Source: Authors, (2020).

What stands out in these first experiments is the high degree of effectiveness that developed environment presented in the manual/semi-automated cleaning process and the treatment of false-positives that are not presented to the user for analysis.

V.3 AUTOMATED DATA CLEANING APPLICATION

The next set of experiments approach the automated data cleaning process, that is, where was no user interaction in both the data analysis and processing stages.

As it can be seen from the results shown in Table 8, similarly rate at 90%, 34 inconsistencies were detected with the application of the algorithm Q-gram, corresponding to 20% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of presented inconsistencies significantly increased. It is interesting to note that, same as with the manual process with the use of synonyms in the automated cleaning process, efficiency increased 58% and 54% of these inconsistencies were detected. With the trained environment, the percentages rose to 69% and 74%. With a similarity of 90%, false-positives are virtually undetected, but not all inconsistencies present in the database are found by data cleaning environment.

Table 8: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 90% similarity.

Q-Gram, q=3, 90%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	34	0	0%	20%
Special Characters	39	0	13%	23%
Phonetic	39	0	13%	23%
Stop words	55	0	38%	33%
Synonymous	92	1	58%	54%
Training	124	0	69%	74%

Source: Authors, (2020).

With similarity factor of 80%, as shown in Table 9 were detected 51 inconsistencies with the application of the algorithm Q-Gram, but 5 of them were false-positives. The difference, namely 46 cases corresponded to 28% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of displayed inconsistencies significantly increased, representing a total of 133 cases detected and the number of false-positives decreased. With the use of synonyms in the cleaning process, the efficiency increased 47% and 60% of these inconsistencies were detected. With the trained environment, efficiency reached 80%. It is emphasized that, with the trained tool, no false-positives were presented.

Table 9: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 80% similarity.

Q-Gram, q=3, 80%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	51	5	0%	28%
Special Characters	55	4	7%	31%
Phonetic	55	4	7%	31%
Stop words	70	3	27%	40%
Synonymous	104	3	47%	60%
Training	133	0	59%	80%

Source: Authors, (2020).

With a similarity factor of 75%, as shown in Table 10, 61 inconsistencies were detected with the application of the algorithm Q-Gram, but 9 of them were false-positives. The difference, namely 52 cases corresponds to 31% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of inconsistencies presented further increases, for a total of 136 cases detected and the number of false-positives decreased. With the use of synonyms in the cleaning process, the efficiency increased by 42%, and 62% of these inconsistencies were detected. With the trained environment, the efficiency reached 81%. Same as the experiment with similarity factor 80% with the trained tool, no false-positives were presented.

Table 10: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 75% similarity.

Q-Gram, q=3, 75%	Inconsistencies	False Positive	+Effectiveness	Effectiveness
Ad-hoc Algorithm	61	9	0%	31%
Special Characters	64	9	5%	33%
Phonetic	64	9	5%	33%
Stop words	77	8	21%	41%
Synonymous	111	7	42%	62%
Training	136	0	53%	81%

Source: Authors, (2020).

With a percentage similarity of 70%, as shown in Table 11, 66 inconsistencies were detected with the application of Q-gram algorithm, but these 9 false-positives. The difference, namely 57 cases correspond to 34% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the amount of inconsistencies presented rose, representing a total of 136 detected cases and the number of false-positives also decreased. With the use of synonyms in the cleaning process, the efficiency increased by 39%, and 60% of these inconsistencies were detected. With the trained environment, the efficiency reached 81%. Same as the experiments with similarity factor 90%, 80% and 75% with the trained tool, no false-positives were presented.

Table 11: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 70% similarity.

Q-Gram, q=3, 70%	**Inconsistencies**	**False Positive**	**+Effectiveness**	**Effectiveness**
Ad-hoc Algorithm	66	9	0%	34%
Special Characters	68	10	3%	35%
Phonetic	68	10	3%	35%
Stop words	77	9	14%	41%
Synonymous	111	10	39%	60%
Training	136	0	50%	81%

Source: Authors, (2020).

Finally, with percentage of similarity of 65% seen in Table 12, 79 inconsistencies were detected in the application of the algorithm Q-Gram, but 16 of them were false-positives. The difference, namely 63 cases correspond to 38% of all inconsistencies and duplicates present in the database. On adding refinement techniques, the number of inconsistencies presented rose, representing a total of 147 detected cases, but the amount of false-positives nevertheless increased until the tool was trained. With the use of synonyms in the cleaning process, the efficiency increased 33% and 62% of these inconsistencies were detected. With the trained environment, the efficiency reached 88%.

Table 12: Results obtained with data cleaning process using the algorithm Q-Grams, with q=3 and 65% similarity.

Q-Gram, q=3, 65%	**Inconsistencies**	**False Positive**	**+Effectiveness**	**Effectiveness**
Ad-hoc Algorithm	79	16	0%	38%
Special Characters	80	14	1%	40%
Phonetic	80	14	1%	40%
Stop words	87	14	9%	44%
Synonymous	119	15	33%	62%
Training	147	0	46%	88%

Source: Authors, (2020).

Even with the cleaning process running automatically without user interaction in the data analysis, we highlight the high degree of effectiveness that the environment has developed, covering approximately 90% of total inconsistencies in the database, and an appropriate treatment of false-positives, as no cleaning was done incorrectly when using the trained environment.

V.4 ANALYSIS AND COMPARISON OF OBTAINED RESULTS

The results obtained by applying data cleaning process using manual and automated approaches are displayed in the next comparative graphs. Due to the rules established by Assurance Criteria used in automated approaches, some cases were not cleaned automatically, and therefore the efficiency of coverage of treated inconsistent cases is somewhat lower.

In Figure 5 is shown the percentage of effectiveness since the application of the algorithm, on the left, until the process with all available resources was executed, such as processing of special characters, phonetics, stop words, multi-language synonyms and training. The automated cleaning efficiency was only 5% lower than the automated when the applied algorithm similarity was 90%.

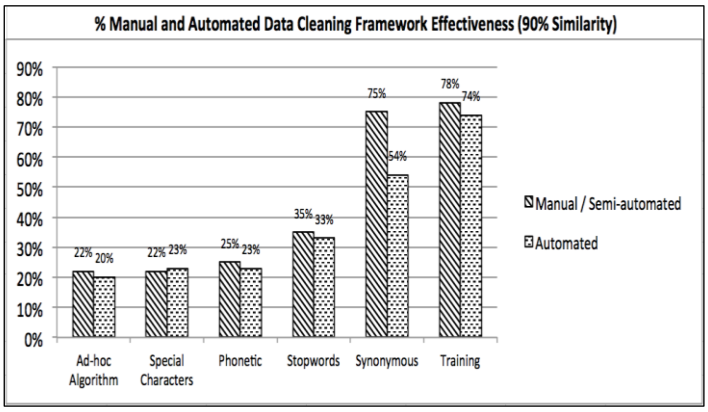


Figure 5: Manual and automated data cleaning environment effectiveness with 90% similarity.
Source: Authors, (2020).

As illustrated in Figure 6, with an 80% similarity algorithm, the automated cleaning with the trained environment had an efficacy of only 3.5% lower than manual.

As can be observed in Figures 7 and 8, with 70% and 65% similarity algorithms, the cleaning automated with the trained environment had an efficiency of approximately 6% less than manual.

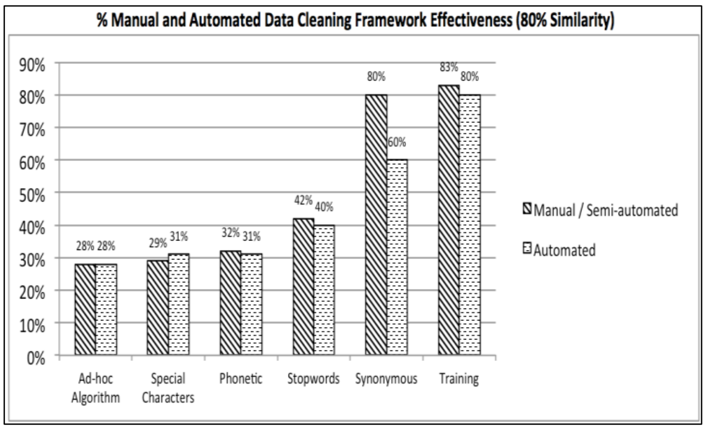


Figure 6: Manual and automated data cleaning environment effectiveness with 80% similarity.
Source: Authors, (2020).

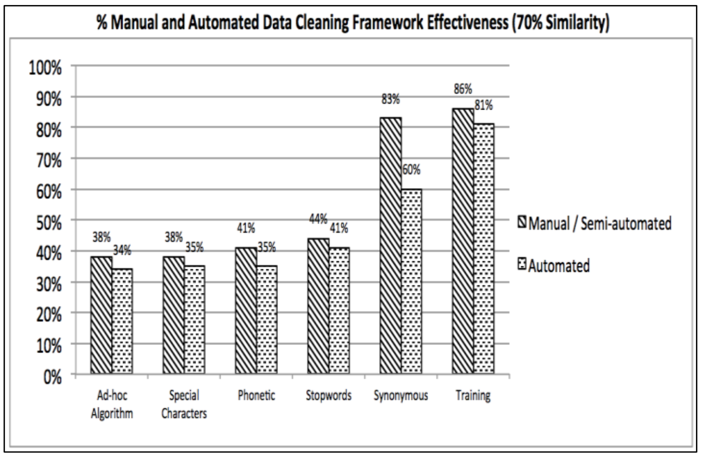


Figure 7: Manual and automated data cleaning environment effectiveness with 70% similarity.
Source: Authors, (2020).

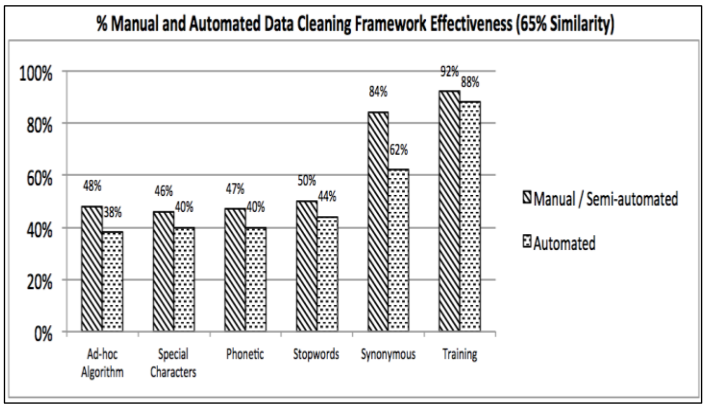


Figure 8: Manual and automated data cleaning environment effectiveness with 65% similarity.
Source: Authors, (2020).

The automated cleaning process appeared to be more efficient when compared to the manual process in analysis, detection and treatment of inconsistency and duplicate problems in databases, but it is important to note that the process can only be automated when there is an environment that supports the treatment of false-positive cases. The percentages of false-positive results shown in Figures 9 and 10. The charts show that only with an intelligent approach is it possible to treat such cases and thus avoid inconsistencies that are presented to the user in case of a manual approach or improperly transformed in the case of automated approach.

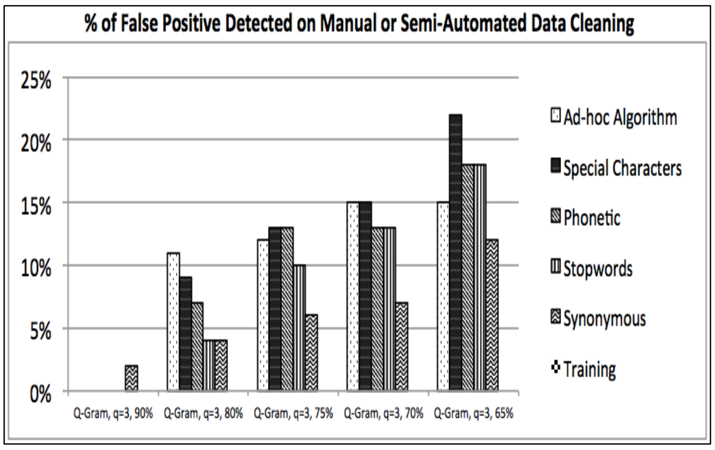


Figure 9: Percentage of false-positives detected in manual data cleaning process.
Source: Authors, (2020).

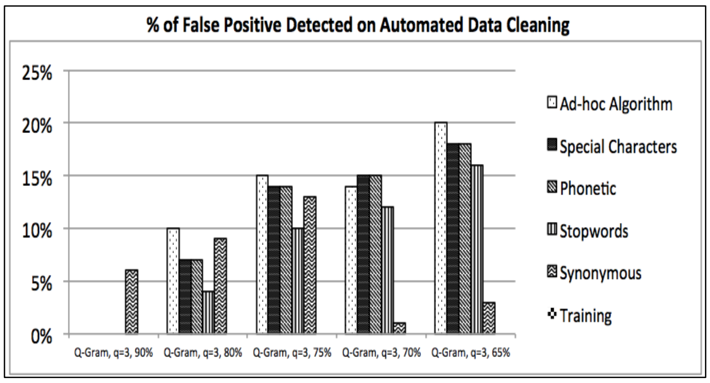


Figure 10: Percentage of false-positives detected in automated data cleaning process.
Source: Authors, (2020).

Depending on the similarity factor used in the application of the algorithms to detect duplicates, the percentage of false-positives can exceed 20%. However, in all experiments in which all these techniques in the developed environment modules were applied simultaneously, including the intelligent approach in which the tool was trained, a false-positive rate of 0% was delivered, ensuring user safety for transformations of inaccurate information present in databases where the environment was applied.

With a proper adequacy, with the proposed algorithm and developed modules being used together, the efficiency of the tool was significant and covered approximately 90% of all inconsistencies in the database, with a 0% percentage of false-positive cases. A survey of the works in literature shows that they only have a 50% efficiency and only part of the functionality and treatments of the developed environment.

Also noteworthy is that the objective of the experiments was to demonstrate technologies and approaches not only to detect and treat positive cases of information inconsistencies and duplication, but also address cases of false-positives detection and analyses the negative impacts that they represent to the data cleaning process, whether manual or automated. It is important to emphasize that this analysis perspective is still largely discussed in literature and can be regarded as being essential in order to obtain increasingly effective results.

VI. CONCLUSIONS

Data cleaning process is shown as an essential and most important step to obtain knowledge. There are few studies that compare the effectiveness of different techniques of data cleaning and the few studies that focus on these analyses are insufficient, since most of the techniques and solutions proposed so far require a great interaction with the user to analyse and decide detections made and, with regard to large databases, it is humanly impossible to direct interaction in the process of cleaning that easily contains millions of records. This interaction is mainly due to the power of semantic decision, because the techniques and work done so far focused specifically on detection of physical similarity between texts and bodies.

This work has, among its objectives, to reduce user interaction in the process of analyzing and correcting inconsistencies and duplications. A data cleaning environment was built for configurable automatic data cleaning based on training and physical-semantic data similarity that provides a more effective and extensible solution, addressing little or unexplored gaps in literature such as training, semantic support, multi-language support and partial or total automation of the cleaning process.

The results of the tests proved the effectiveness of all the developed features, relevant to each module of the proposed architecture. In several scenarios the experiments demonstrated the effectiveness of the tool. With a proper configuration, and with the proposed algorithm and developed modules used together, the efficiency is significant and covers approximately 90% of database inconsistencies, with a 0% percentage of false-positive cases. The work area in literature has presented only part of the functionality and treatments when compared to the developed environment, and their effectiveness does not exceed 50%.

Approaches have also demonstrated that in addition to detecting and treating positive cases of information inconsistencies and duplication, they address cases of detected false-positives and consider the negative impacts that they represent to the data cleaning process, whether manual or automated, not yet discussed in literature.

VII. AUTHOR'S CONTRIBUTION

Conceptualization: Carlos Roberto Valêncio, Toni Jardini and Victor Hugo Penhalves Martins.
Methodology: Carlos Roberto Valêncio and Toni Jardini.
Investigation: Carlos Roberto Valêncio and Toni Jardini.
Discussion of results: Carlos Roberto Valêncio, Toni Jardini, Victor Hugo Penhalves Martins and Angelo Cesar Colombini.
Writing – Original Draft: Toni Jardini.
Writing – Review and Editing: Carlos Roberto Valêncio and Angelo Cesar Colombini.
Resources: Toni Jardini.
Supervision: Carlos Roberto Valêncio and Márcio Zamboti Fortes.
Approval of the final text: Carlos Roberto Valêncio and Márcio Zamboti Fortes.

VIII. REFERENCES

[1] Bharat, K. et al.. Special Issue on Data Cleaning, Bulletin of the Technical Committee on Data Engineering, 2000.

[2] Andrade, T.L.; Souza, R.C.G.; Balbini, M.; Valêncio, C.R.. Optimization of Algorithm to Identification of Duplicate Tuples through Similarity Phonetic Based on Multithreading, in Proc. of 2011. 12th International Conference on Parallel and Distributed Computing, Applications and Technologies - PDCAT, 299–304, 2011. DOI: 10.1109/PDCAT.2011.58.

[3] Rahm, E.; Do, H.H.. Data Cleaning: Problems and Current Approaches. IEEE Data Engineering Bulletin 23, pp. 3–13, 2000.

[4] Silva, L.A.E..A Data Mining Approach for Standardization of Collectors Names in Herbarium Database. IEEE Latin America Transactions, vol.14, no.2, pp.805-810, 2016. DOI: 10.1109/TLA.2016.7437226.

[5] Elmagarmid, A.K.; Ipeirotis, P.G.; Verykios, V.S.. Duplicate Record Detection: A Survey. IEEE Transactions on Knowledge and Data Engineering, vol.19, no.1, p.1–16, 2007. DOI: 10.1109/TKDE.2007.250581.

[6] Ayad, L.A.K.; Barton, C.; Pissis, S.P.; A faster and more accurate heuristic for cyclic edit distance computation. Patter Recognition Letters, vol.88, p.81-87, 2017. DOI: 10.1016/j.patrec.2017.01.018.

[7] Su, Z. et al.. Plagiarism Detection Using the Levenshtein Distance and Smith-Waterman Algorithm, in Proc. of 3rd International Conference on Innovative Computing, Information and Control - ICICIC '08, p.569-573, 2008. DOI: 10.1109/ICICIC.2008.422.

[8] Gueddah, H.; Yousfi, A.; Belkasmi, M.. The filtered combination of the weighted edit distance and the Jaro-Winkler distance to improve spellchecking Arabic texts, in Proc. of 12th International Conference of Computer Systems and Applications – AICCSA, 2015. DOI: 10.1109/AICCSA.2015.7507128.

[9] Hanada, H.; Kudo, M.; Nakamura, A.. Average-case linear-time similar substring searching by the q-gram distance. Theoretical Computer Science. vol. 530, p.23-41, 2014. DOI: 10.1016/j.tcs.2014.02.022.

[10] Wang, G.; Wang, B.; Yang, X.; Yu, G.. Efficiently Indexing Large Space Graphs for Similarity Search. IEEE Transactions on Knowledge and Data Engineering. vol.24, no.3, p.440-451, 2010. DOI: 10.1109/TKDE.2010.28.

[11] Petrovic, S.; Bakke, S.. Improving the Efficiency of Misuse Detection by Means of the q-gram Distance, in Proc. of Fourth International Conference on Information Assurance and Security - ISIAS '08, p.205–208, 2008. DOI: 10.1109/IAS.2008.39.

[12] Khristodulo, O.I.; Makhmutov, A.A.; Sazonova, T.V.. Use Algorithm based at Hamming Neural Network Method for Natural Objects Classification. Procedia Computer Science, vol. 103, p.388-395, 2017. DOI: 10.1016/j.procs.2017.01.126.

[13] Boutalis, Y.S.. A new method for constructing kernel vectors in morphological associative memories of binary patterns. Computer Science and Information Systems, vol.8, no.41, p.141-166, 2011. DOI: 10.2298/CSIS091114026B.

[14] Chen, S.Y. et al.. Concept Extraction and Clustering for Search Result Organization and Virtual Community Construction. Computer Science and Information Systems, vol. 9, no.1, p.323-355, 2012. DOI: 10.2298/CSIS101124020C.

[15] Jiang, Z.; Evans, M.; Oliver, D; Shekkar, S.. Identifying K Primary Conditions from urban bicycle GPS trajectories on a road network. Information Systems, vol.57, p.142-159, 2016. DOI: 10.1016/j.is.2015.10.009.

[16] Hernandez, A.F.R.; Garcia, N.Y.H.; Distributed processing using cosine similarity for mapping Big Data in Hadoop. IEEE Transactions on Latin America, vol.14, no.6, p.2857-2861, 2016. DOI: 10.1109/TLA.2016.7555265.

[17] Jimenez, S.; Gonzalez, F.A.; Gelbukh, A.. Mathematical properties of soft cardinality: Enhancing Jaccard, Dice and cosine similarity measures with element-wise distance. Information Sciences, vol.367-368, p.373-389, 2016. DOI: 10.1016/j.ins.2016.06.012.

[18] Zhu, S.; Wu, J.; Xia, G.. TOP-K Cosine Similarity Interesting Pairs Search, in Proc. of Seventh International Conference on Fuzzy Systems and Knowledge Discovery - FSKD 2010, 1479–1483, 2010. DOI: 10.1109/FSKD.2010.5569212.

[19] Monge, A.E.; Elkan, C.P.. The Field Matching Problem: Algorithms and Applications, in KDD-96 Proceedings, p.267–270, 1996.

[20] Cohen, W.W.. Integration of heterogeneous databases without common domains using queries based on textual similarity, in Proc. 1998 ACM SIGMOD International Conference on Management of data, 201–212, 1998. DOI: 10.1145/276304.276323.

[21] Gravano, L. et al.. Text joins in an RDBMS for web data integration, in Proc. of WWW '03 – 12th International Conference on World Wide Web, 90-101, 2003. DOI: 10.1145/775152.775166.

[22] Holmes, D.; McCabe, M.C.. Improving precision and recall for Soundex retrieval, in Proc. of International Conference on Information Technology: Coding and Computing – ITCC, p.22–26, 2002. DOI: 10.1109/ITCC.2002.1000354.

[23] Mandal, A.K.; Hossain, M.D.; Nadim, M.. Developing an efficient search suggestion generator, ignoring spelling error for high speed data retrieval using Double Metaphone Algorithm, in Proc. of 13th International Conference on Computer and Information Technology – ICCIT, p.317–320, 2010. DOI: 10.1109/ICCITECHN.2010.5723876.

[24] Taft, R.L.. Project SEARCH - Name Search Techniques – Special Report No. 1, New York State Identification and Intelligence System, 1-118, 1970.

[25] Gill, L.E.. OX-LINK: The Oxford Medical Record Linkage System, in International Workshop and Exposition, p.15–45, 1997.

[26] Cohen, W.W.. Data Integration using Similarity joins and a Word-Based Information Representation Language. ACM Transactions on Information Systems, vol.18, no.3, p.288-321, 2000.

[27] Flamingo. Flamingo Project on Data Cleaning, Department of Computer Science. UC Irvine, 2017. http://flamingo.ics.uci.edu.

[28] Yancey, W.E.. Big Match: A program for extracting probably matches from a large file for record linkage. U.S. Census Bureau, 2004. <http://
http://ww2.amstat.org/sections/srms/Proceedings/y2004/files/Jsm2004-000592.pdf>.

[29] Valêncio, C.R. et al.. Otimização de técnicas e algoritmos aplicados à limpeza de Banco de Dados utilizando Multithreading, in Proc. of Conferencia Ibero Americana WWW/Internet 2010 - IADIS, p.414-418, 2010.

[30] ANU Data Mining Group.: Febrl - FREELY Extensible Biomedical Record Linkage, 2012. <http://datamining.anu.edu.au/projects/linkage.html>.

[31] Christen, P.. A Freely Available Record Linkage System with a Graphical User Interface, in Australasian Workshop on Health Data and Knowledge Management - HDKM 2008. p.17–25, 2008.

[32] Elfeky, M.G.; Verykios, V.S.; Elmagarmid, A.K.. TAILOR: a record linkage toolbox, in Proc.of 18th International Conference on Data Engineering, 17–28, 2002. DOI: 10.1109/ICDE.2002.994694.

[33] Oliveira, P.; Rodrigues, F.; Henriques, P.. SmartClean: An Incremental Data Cleaning Tool, in Proc. of 9th International Conference on Quality Software - QSIC 2009, 452–457, 2009. DOI: 10.1109/QSIC.2009.67.

[34] Yan, H.; Diao, X.C.; Li, K.Q.. Research on Information Quality Driven Data Cleaning Framework, in Proc. Of International Seminar on Future Information Technology and Management Engineering - FITME´08, p.537–539, 2008. DOI: 10.1109/FITME.2008.126.

[35] Yu, H.; Yi, Z.X.; Zhen, Y.; Guo-quan, J.. A Universal Data Cleaning Framework Based on User Model, in Proc.of International Colloquium on Computing, Communication, Control and Management - CCCM 2009, p. 200–202, 2009. DOI: 10.1109/CCCM.2009.5267946.

[36] Bao, Y.; Song, J.; Shi, J.; Yu, G.. Case Study on Modeling Approaches and Framework of Scientific Data Cleaning, in Proc. of 9th IEEE International Conference on Computer and Information Technology - CIT'09, p.266–271, 2009. DOI: 10.1109/CIT.2009.88.

[37] Arasu, A.; Kaushik, R.. A grammar-based entity representation framework for data cleaning, in Proc. of ACM SIGMOD International Conference on Management of data - SIGMOD'09, p.233-244, 2009. DOI: 10.1145/1559845.1559871.

[38] Suriadi, S. et al.. Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. Information Systems, vol. 64, p.132-150, 2017. DOI: 10.1016/j.is.2016.07.011.

[39] Shi, Y.; Li, S.. The design and implementation of dynamic data cleaning modeling, in Proc. of 2010 International Conference on Computer Application and System Modeling – ICCASM, 2010. DOI: 10.1109/ICCASM.2010.5619213.

[40] Ali, K.; Warraich, M.A.. A framework to implement data cleaning in enterprise data warehouse for robust data quality, in Proc. of 2010 International Conference on Information and Emerging Technologies - ICIET 2010, p.1–6, 2010. DOI: 10.1109/ICIET.2010.5625701.

[41] Berti-Equille, L.; Dasu, T.; Srivastava, D.. Discovery of complex glitch patterns: A novel approach to Quantitative Data Cleaning, in Proc. of 27th International Conference on Data Engineering – ICDE, p.733 –744, 2011. DOI: 10.1109/ICDE.2011.5767864.

[42] Ciszak, L.. Application of clustering and association methods in data cleaning, in Proc. of International Multiconference on Computer Science and Information Technology - IMCSIT 2008, p. 97–103, 2008. DOI: 10.1109/IMCSIT.2008.4747224.

[43] Qian, X. et al.. Data cleaning approaches in Web2.0 VGI application, in Proc. of 17th International Conference on Geoinformatics, p. 1–4, 2009. DOI: 10.1109/GEOINFORMATICS.2009.5293442.

[44] Wang, H.. The Research of Outlier Data Cleaning through Relevance Comparison, in Proc. of 2nd International Conference on e-Business and Information System Security – EBISS, 1–3, 2010. DOI: 10.1109/EBISS.2010.5473717.

[45] Okita, T.. Data cleaning for word alignment, in Proc. of ACL-IJCNLP Student Research Workshop, p.72–80, 2009.

[46] Bertossi, L.; Kolahi, S.; Lakshmanan, L.V.S.. Data cleaning and query answering with matching dependencies and matching functions, in Proc. of 14th International Conference on Extending Database Technology - ICDT'11, p.268-279, 2010.

[47] Chaturvedi, S. et al.. Optimal Training Data Selection for Rule-Based Data Cleansing Models, in Proc. of 2011 Annual SRII Global Conference, p.126–134, 2011. DOI: 10.1109/SRII.2011.25.

[48] Prasad, K.H. et al.. Data Cleansing Techniques for Large Enterprise Datasets, in Proc. of 2011 Annual SRII Global Conference, p.135–144, 2011. DOI: 10.1109/SRII.2011.26.

[49] SQLite: SQLite. http://www.sqlite.org.

[50] Yu, Z., Bai, C.; Cai, K.Y.. Mutation-oriented test data argumentation for GUI software fault localization. Information and Software Technology, vol.55, no.12, p.2076-2098, 2013. DOI: 10.1016/j.infsof.2013.07.004