Semantic Similarity Match for Data Quality

Fernando Martins
André Falcão
Francisco Couto

DI–FCUL TR–07–25

November 2007

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Campo Grande, 1749–016 Lisboa
Portugal

Technical reports are available at `http://www.di.fc.ul.pt/tech-reports`. The files are stored in PDF, with the report number as filename. Alternatively, reports are available by post from the above address.

Semantic Similarity Match for Data Quality

Fernando Martins fmp.martins@gmail.com
Francisco Couto fcouto@di.fc.ul.pt
André Falcão afalcao@di.fc.ul.pt

Faculty of Sciences of the University of Lisbon
Portugal

November 2007

Abstract

Data quality is a critical aspect of applications that support business operations. Often entities are represented more than once in data repositories. Since duplicate records do not share a common key, they are hard to detect. Duplicate detection over text is usually performed using lexical approaches, which do not capture text sense. The difficulties increase when the duplicate detection must be performed using the text sense. This work presents a semantic similarity approach, based on a text sense matching mechanism, that performs the detection of text units which are similar in sense. The goal of the proposed semantic similarity approach is therefore to perform the duplicate detection task in a data quality process.

1 Introduction

The exploration of data is critical in todays modern economy. This means that the quality of data has a great impact in every task that relies on it. Unfortunately, the existence of anomalies and impurities in real world data is a well known reality, leading to a higher cost and less accurate data processing. For instance, data mining results, a scientific research or a data warehouse can be affected by the quality of the data in ways that the results may be erroneous and lead to incorrect conclusions [12, 14, 24, 27]. Data cleaning arises in this scenario as a required tool for data quality improvement.

The term *data cleaning*, also known as *data cleansing* or *scrubbing*, refers, in a loose sense, to the task of detecting and eliminating errors from data. The original data cleaning goal was centered in the elimination of duplicates in a data collection, a problem already present in single database applications. This problem got worse when multiple database applications spread across organization departments and it gained a new dimension when heterogeneous data sources integration became an organization demand. A new challenge arose when large amounts of structured information started to derive from unstructured information, like text documents or web pages. This information is typically imprecise and noisy, making duplicate record detection techniques crucial for improving the quality of the data. Therefore, data quality changed from a single problem to became a critical issue for organizations.

1.1 Text Similarity Detection

Elmagarmid et al. [7] show the distinct approaches proposed to detect duplicate text, which are all based on lexical or phonetic matching techniques. The typical approach to find the similarity between two text segments is to use a simple similar matching method and then produce a similarity score based on the number of lexical units that occur in both input segments [9, 15]. Improvements may be obtained by using Natural Language Processing (NLP) techniques such as stemming, stop-word removal, part-of-speech tagging, longest subsequence matching, as well as various weighting and normalization factors. While successful to a certain degree, these lexical matching similarity methods fail to identify the semantic similarity of texts. For instance, there is an obvious similarity between the text segments "*I own a car*" and "*I have an automobile*". Existing works [5, 11, 16, 25, 26] already address the capture of semantic content in texts, usually using machine learning techniques, but most of the current text similarity metrics fail in identifying any kind of connection between these texts since they rely essentially on lexical matching or explore specific or constrained linguistic aspects [5, 7, 11].

The only exception to this trend is perhaps the Latent Semantic Analysis method [16, 26] which represents an improvement over earlier attempts to use measures of semantic similarity for information retrieval.

This work address a distinct matching technique which is based on semantic text similarity. The proposed matching technique is aimed to be used as an *approximate join* operator in order to guarantee Uniqueness, i.e. perform approximate record matching in the duplicate elimination cleaning process. The duplicate detection problem has been widely studied [3, 4, 8, 17] and is centered in the detection and elimination of duplicate entities, i.e. entities that don't have an unique representation on a data repository.

This work is organized as follows. Section 2 describes data quality. Section 3 introduces a semantic similarity definition and its usage within duplicate detection and elimination. Section 4 describes WordNet and its role in the presented semantic similarity approach. Section 5 overviews open questions and problems of the proposed approach. Finally, Section 6 concludes this work.

2 Data Quality

The term *data quality* refers to the problems and solutions that aim to maintain data *clean*, i.e. the detection and elimination of data errors. Comprehensive data cleaning is defined as the set of operations performed on existing data to detect and remove anomalies so that there is an unique and correct representation of each entity. This section presents the data anomalies, quality metrics and cleaning process in order to achieve and maintain high quality data.

2.1 Anomalies

A data anomaly is a violation of a syntactic, semantic or coverage rule, as identified by Muller and Freytag [23], and it can be seen as a property of data values that holds a wrong value of the value representation. Data anomalies arise from several distinct sources like inappropriate data model, multiple data sources

integration and user inputs. The brief overview of common data anomalies presented here follows the definitions and taxonomies discussed in previous works [1, 13, 23, 24].

Syntactical anomalies describe characteristics concerning the format and values used in the representation of entities. **Lexical errors** are discrepancies between the structure of the data items and the specified format. A simple example is a *gender* attribute that should only contain a M or F symbol but indeed it holds other symbols, like digits. **Domain format errors** specify errors where the given value for an attribute does not conform with the anticipated domain format. An example is a *phone number* specified in the format of +{*country*} {*operator*}{*area*}-{*number*} but holding values like *+351 214-187-298* and *+351 214187298*. They represent valid phone numbers but do not comply with the specified domain format. **Irregularities** are concerned with non-uniform use of values, units and abbreviations. This can happen, for example, if different currencies are used to specify an employees expense. In this example, it is critical that the currency is explicitly listed for each value, otherwise the value interpretation will be incorrect.

Integrity constraints violation describe tuples, or sets of tuples, that do not satisfy at least one integrity constraints defined by the schema. A simple example is a foreign key value to a lookup table that holds a non-existent lookup table key value.

The semantic anomalies cause the representations to be incorrect in the data collection. This happens because the semantic anomalies hinder the representations to be non-redundant and comprehensive. **Contradictions** are values that violate some kind of dependency between them, like a functional dependency or duplicates with inexact values. An example is person attributes *age* and *birth date* holding contradictory values. **Duplicates** are several representations of the same entity. The value of each representation does not need to be completely identical. Inexact duplicates are specific cases of contradiction between two or more entity representations, they represent the same entity but with different values for all or some of its properties. A simple example is a company that may is represented by *ACME*, *ACME Corp.* and *ACME Corporation*. The entity is in fact the same, but its representation is duplicated. **Invalid tuples** do not display anomalies such as the ones described above but still do not represent valid entities. This makes invalid tuples the most complex class of anomaly since they result from the inability to describe reality within a formal model through integrity constraints.

The coverage anomalies derive from missing data and impacts the amount of representations in the data collection. **Missing values** result from omissions, usually while collecting data. This is to some degree a constraint violation if there are *null* values for attributes where there exists a *not null* constraint defined. In other cases such constraint may not be present thus allowing *null* values. In these cases it is necessary to decide whether the value exists and, if so, what it is. **Missing representations** arise from omissions of complete existent entities that are not represented in the data.

2.2 Quality Criteria

A quality criteria has three main purposes, (i) quantify the necessity of data cleaning, (ii) prioritize the execution of data cleaning transformations and, (iii)

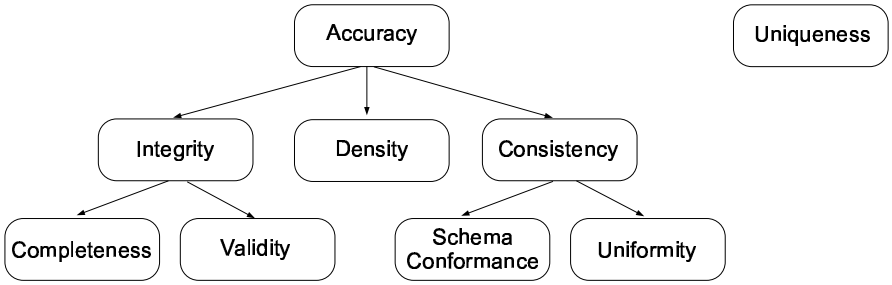

Figure 1: Hierarchy of data quality criteria by Muller and Freytag [23].

quantify the success of a performed data cleaning process. Figure 1 shows the hierarchy of data quality criteria presented by Muller and Freytag [23], from which a brief overview of the main criteria definition follows.

Accuracy, also known as correctness, verifies how the data reflects the real world entities, i.e., it is concerned with the entities value representation. Simply stated, accuracy means that a data collection does not have any anomalies, except duplicates.

Uniqueness, verifies if an entity has a single representation in the data collection. Simply stated, a collection that is unique does not contain duplicates.

A data collection that is accurate and unique does not contain any of the anomalies described in Section 2.1. This describes a data collection that does not need to be cleaned, hence to be said of high quality.

2.3 Data Cleaning Process

Data cleaning is defined as a semi-automatic process of operations performed on data since there is intervention from a domain expert. Data cleaning may include structural transformation, i.e. transformation of data into a format that is better manageable or better fitting. The data cleaning process, displayed in Figure 2, comprises the following major steps: (i) auditing data to identify the types of anomalies, (ii) choosing appropriate methods to automatically detect and remove anomalies, i.e. the specification and execution of the cleaning workflow, (iii) applying the methods to the tuples in the data collection and, (iv) post-processing, or control, where the results are examined.

The data cleaning operations that transform dirty data, as defined in Section 2.1, in clean data, as defined in Section 2.2, should be applied preferable in the following order:

i format adaptation for tuples and values, aimed at Accuracy through the elimination of lexical and domain format anomalies;

ii integrity constraint enforcement, aimed at Accuracy through the elimination of irregularities and integrity constraint anomalies;

iii derivation of missing values from existing ones, aimed at Accuracy through the elimination of missing values and representations, when possible;

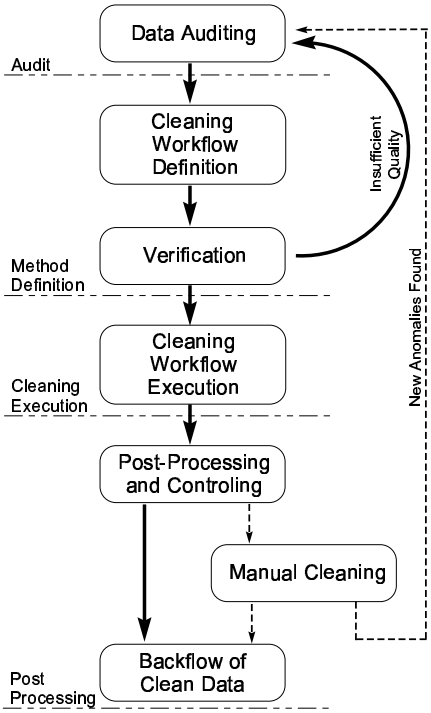

Figure 2: Data cleaning process.

iv removing contradictions within or between tuples, aimed at Accuracy through the elimination of contradictions;

v detecting, merging and eliminating duplicates, aimed at Uniqueness through the detection and elimination of duplicate entities;

vi and detection of outliers, aimed at Accuracy through the elimination of invalid tuples and values.

The cleaning operations (i), (ii), (iii), (iv) and (vi) are aimed at Accuracy, as specified in the previous Section 2.2, since they are centred in the value representation of the entities. The current semantic similarity approach presented here is aimed at the cleaning operation (v), the duplicate detection task, in order to guarantee the Uniqueness specified in Section 2.2. A set of techniques to prevent, verify or repair each identified problem is further detailed in other works [1, 13].

3 Semantic Similarity

Similarity is a complex concept which has been widely discussed in the linguistic, philosophical, and information theory communities. For the current task, semantic similarity between two text units is defined as sense share, i.e. both text units share concepts above a predefined threshold. This work assumes an uniform relation distance between senses, meaning that all relations are equal in distance. The assumption of uniform distances is not entirely correct and Resnik [25] proposes a metric of semantic similarity in an IS-A taxonomy based on probability to avoid the unreliability of edge distance. Figure 3 shows fragments of a taxonomy where it is possible to see the uniform relation distance. While *credit card* is equivalent to *nickel* and *dime*, they only relate far up in the hierarchy, despite both are a *medium of exchange*, since *nickel* and *dime* have to go through a more dense taxonomy. For the current task, a Relation Distance of one means that *nickel* is related with *dime* since both relate with *coin* at a distance of one link, but it is not related with *credit card* despite both are a *medium of exchange*. The word similarity is judged by taking the maximal information content over all concepts between words. This quantitative characterization of information provides a way to measure the semantic similarity. The more information two words share in common, the more similar they are.

3.1 Related Work

Although previous works on semantic similarity exist [5, 6, 11, 16, 18, 25, 26], they differ in definition, usage of linguistic features, such as synonyms match in Hatzivassiloglou et al. [11] and word relation in Resnik [25], and approach as a whole from the presented approach. The three main differences among the approaches are: (i) the proposed approach aims at providing a data cleaning operator for the duplicate detection task while previous works did not address this, as to the extent of the knowledge acquired for this work; (ii) the conceptual distance that formalizes the notion of similarity, while also creating a profile of characteristics of a text fragment, is not obtained by calculations based on frequency or vector distance, but instead it is obtained by measuring the shared

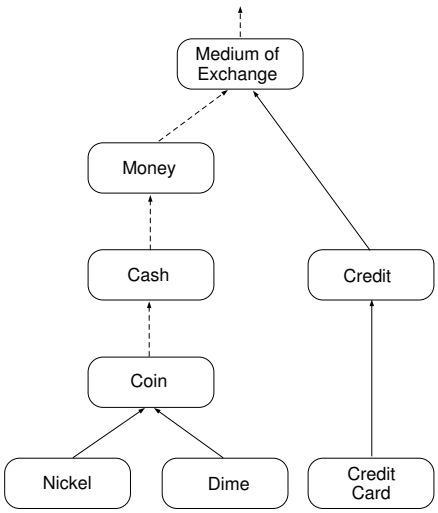

Figure 3: WordNet Is-A Relation example from Resnik [25]. Solid lines represent direct relations and dashed lines represent relations with other nodes in between.

senses between text units; (iii) machine learning techniques over annotated corpora are commonly used, but the proposed approach does not use this technique since it is, usually, considered unfeasible on a real data cleaning scenario.

Despite the sharing of some common features, like stop word removal, linguistic features combination and the usage of thesaural information, the conceptual semantic similarity technique presented in this work is an original approach that uses previous works only as a startup basis.

3.2 Proposed Approach

The semantic similarity matching mechanism has three distinct phases, a configuration phase, a setup phase and a matching phase, as described below.

The configuration phase allows to tune the semantic similarity matching mechanism. If no specific configuration is provided, default values will be provided. Since no real tests have been performed with the proposed approach, it is needless to say that the default values have been set on an empirical basis.

1. The parameter *relation distance* set the maximum number of semantic links between two senses until a common sense is reached. The value must be a natural number. The parameter default value is 1.

2. The parameter *sst*, short for *semantic similarity threshold*, set the limit from which two text units are considered semantically similar. The value is a real number from $[0, 1]$. Since lower values produce loose similarities and uninteresting results, empirically, *sst* should actually not be lower than 0.75. If no value is provided, the default value of 0.85 will be used.

When the parametrization if concluded, the setup phase will build a set of sense matrices for each text unit through the following way:

1. The sentence is parsed in order to assess the *syntactical category* of each word.

2. The *stop words* are removed to reduce the nonsense words and to decrease the computation time.

3. The words are tagged with a *key*, which is the radical derived from the morphological inflection of the original word.

4. All equal *keys* under the same *syntactical category* are merged under a single tuple $(key, syntactical\ category)$.

5. Each *key* from each tuple $(key, syntactical\ category)$ is used to get the correspondent *synonym set*, i.e. a vector with all *key* synonyms, using WordNet, presented in Section 4.

6. A *relation distance* is defined as the maximum number of semantic links between two senses until a common sense is reached.

7. Each *key* from each tuple $(key, syntactical\ category)$ is used to get the *related word list*, depending on the predefined *relation distance* value, also using WordNet, presented in Section 4.

8. Finally, each *syntactical category* origins a *sense matrix* which is populated with the *keys*, from the correspondent tuples $(key, syntactical\ category)$, *synonym sets* and *related word lists*, as exemplified in Table 1.

In a formal way, let T be the set of text units, i.e. $T = \{t_1, ..., t_n\}$. Let C be the set of syntactical categories, i.e. $C = \{c_1, ..., c_n\}$. Let K be the set of keys of a text unit t_i for a given syntactic category c_j, i.e. $K(t_i, c_j) = \{k_1, ..., k_n\}$, where k_i represents a *key*. Let S be the set of synonyms for a text unit such that $S = \{s_1, ..., s_n\}$, then $S(k_i)$ is the set of synonyms of k_i, meaning $S(k_i) = s_i$, where s_i represents a *synonym set*. Let R be the set of related words of a text unit such that $R = \{r_1, ..., r_n\}$, then $R(k_i)$ is the set of related words of k_i, meaning $R(k_i) = r_i$, where r_i represents a *related word list*, i.e. $r_i = \{w_1, ..., w_n\}$. Let $Sense$ be the function that asserts about the similarity of two sense matrices. Let $\sim$ be the *semantic similarity* operator. Let sst represent the *semantic similarity threshold* from which two text units are considered semantically similar.

Once the texts units have been setup, each text unit has its own set of sense matrices, as exemplified in Table 1, and the matching mechanism can now evaluate the texts semantic similarity. Note that in order to comply with the predefined goal of working as an approximate join operator, $Sense$, is a binary operator.

$$T_i \sim T_j = (Sense(S_i, S_j) \geq sst) \tag{1}$$

Equation 1 verifies if two text units, T_i and T_j, are semantically similar by verifying if their correspondent sense matrices, S_i and S_j, are equal or above the predefines *semantic similarity threshold*.

Key $K(t_i, noun) = \{k_1, ..., k_n\}$	**Synonym Set** $S(k_i) = s_i$	**Related Word List** $R(k_i) = r_i$
k_1 =car	$S(car)$ = { car, auto, automobile, machine, motorcar, gondola, railcar, railway car, railroad car, elevator car }	$R(car)$ = { wheeled vehicle, compartment, automotive vehicle, motor vehicle }
k_2 =automobile	$S(automobile)$ = { car, automobile, auto, machine, motorcar }	$R(automobile)$ = { motor vehicle, automotive vehicle }

Table 1: Example of a Noun Sense Matrix for a text unit t_i with a Relation Distance set to 1.

$$Sense(S_i, S_j) = \frac{\sum_{k_i \in S_i, k_j \in S_j} \begin{cases} 1 & \text{if} \quad k_i \in S(k_j) \\ 1 & \text{if} \quad k_i \in R(k_j) \\ 1 & \text{if} \quad k_j \in R(k_i) \\ \frac{\sum_{w_a \in r_i, w_b \in r_j} {}^{w_a = w_b}}{|r_i| \times |r_j|} & \text{otherwise} \end{cases}}{min(|S_i|, |S_j|)} \tag{2}$$

Equation 2 sums all the matches between the senses of each key of each text unit and normalizes it through the minimum number of keys, since that is the maximum number of matches possible. The senses between the keys are computed the following way: (i) if a *key* belongs to the *synonym set* of the other key, note that $k_i \in S(k_j) <=> k_j \in S(k_i)$, then a hit is achieved and the result of the match is 1; (ii) if a *key* belongs to the *related word list* of the other key, then a hit is also achieved and the result of the match is also 1; (iii) else, the matches of the *words* of the *relation word lists* are summed and then normalized. Note that the words w_a and w_b being evaluated are really senses, rather than words, of the original word in the original text T. If the resulting sum is equal or greater than sst, that means that $T_i \sim T_j$ and the texts are semantically similar.

3.3 Duplicate Elimination

Duplicate elimination, also known as *merge/purge*, *entity reconciliation*, *record linkage* and *duplicate record detection*, is centred in the detection and elimination of duplicate entities, i.e. entities that do not have an unique representation on a data repository, as defined in Section 2.2. Duplicate elimination has been widely studied [3, 4, 8, 10, 17], but, as stated in Elmagarmid et al. [7], the matching techniques used are lexical, phonetic and numeric, leaving semantic matching unrefered. Also as stated in Elmagarmid et al. [7], there are currently two main approaches for duplicate record detection: (i) database research, that aims to simple and fast duplicate detection techniques in order to be applied to millions of records, and (ii) machine learning and statistics research, that aims

to develop more sophisticated matching techniques that rely on probabilistic models.

The detection of duplicate entities usually does not rely on a single entity attribute but rather on a set of entity attributes. The standard algorithm for duplicate elimination presented in Gu et al. [10] relies on a likelihood ratio that can be achieved using the presented semantic similarity matching mechanism to find a score when comparing two text attibutes.

When a match is found, i.e. when two representations of the same entity are found, it is necessary to update the system in order to keep an unique entity representation. There are two main techniques used to achieve this goal. The simplest of them all is to select one of the representations, eliminating the other one. The second technique is to merge the information of the representations, i.e. to create a new representation by selecting information from the two original representations, which are then eliminated.

4 WordNet

Meaningful sentences are composed of meaningful words and any system that hopes to process natural languages as people do, must have information about words and their meanings. Since the approach proposed in this work aims to detect text similarity through its sense, it must *understand* the meaning of the words mentioned in the sentences. To that aim, WordNet can be used.

WordNet is an on-line lexical database whose design is inspired by psycholinguistic theories of human lexical memory, as explained by Miller et al. [20, 21], and is currently available for use under program control. Therefore, WordNet provides an effective combination of traditional lexicographic information and modern computation. WordNet started as an English language project but has now spread into other languages. Nouns, verbs, adjectives and adverbs are organized into synonym sets, commonly known as *synsets*, each representing one underlying lexical concept. WordNet provides sense information, placing words in the synonym sets, with semantic relation links, as presented in Table 2.

Semantic Relation	**Syntactic Category**	**Examples**
Synonymy (similar)	N, V, Aj, Av	pipe, tube; rise, ascend; sad, unhappy; rapidly, speedily;
Antonymy (opposite)	Aj, Av, (N, V)	wet, dry; powerful, powerless; friendly, unfriendly; rapidly, slowly;
Hyponymy (subordinate)	N	sugar maple, maple; maple, tree; tree, plant;
Meronymy (part)	N	brim, hat; gin, martini; ship, fleet;
Troponomy (manner)	V	march, walk; whisper, speak;
Entailment	V	drive, ride; divorce, marry;

Table 2: Semantic Relations in WordNet from Miller [20] where N stands for nouns, V for verbs, Aj for Adjectives and Av for Adverbs.

In WordNet, a *form* is represented by a string of ASCII characters, in this work referred only as *word* without loss of generality, a *sense* is represented by

the set of one or more synonyms that have that sense, and a *gloss* explains a concept and provides one or more examples with typical usage of that concept.

WordNet respects the syntactic categories noun, verb, adjective, and adverb. Inflectional morphology for each syntactic category is accommodated by the interface to the WordNet database. For example, if information is requested for *went* the system will return what it knows about the verb *go*.

WordNet is widely used in semantic based similarity approaches [2, 5, 11], and in several distinct areas as information retrieval [22].

The previously described matching mecanism in Section 3.2 can also make use of WordNet. The *synonym sets* described in Step 5 can be directly retrived from WordNet synonym sets and the *related word list* described in Step 7 can be obtained from the available semantic relations, such as the IS-A relation exemplified in Figure 3. As explained by Couto et al. [6], it is possible that higher correlations could be achieved through the usage of disjunctive common ancestors as in GraSM.

5 Questions and Problems

As stated by Corley and Mihalcea [5], the *bag-of-words* approach, as seen in the lexical approaches, ignores many important relationships in sentence structure, such as dependencies between words, or roles played by the various arguments in the sentence. The proposed approach does account for relationships, like sentence syntax structure and word sense, making it more like a *bag-of-senses*. However, there are some issues to which it was not possible to find an answer.

5.1 Prototype

A prototype of the proposed approach, as specified in Section 3.2, has been implemented, but unfortunately it was only possible to perform some small and *ad-hoc* tests. The prototype used a part-of-speech tagger based on the Penn Treebank part-of-speech tag set, as explained by Marcus et al. [19], forcing a translation between the part-of-speech tag into the four syntactic categories available in WordNet. The prototype used the default values and its results, computed over small journalistic texts, looked promising. Unfortunately, there were no real tests performed, thus not allowing to assert about its real behavior.

5.2 Open Questions and Problems

Since this approach has not been really tested and since it relies on linguist features, like text tagging and semantic relationships which are still under discussion in several communities, some problems and open questions arise.

Tagger. When tagging text with a syntactic tag set distinct from the one available in WordNet, the translation between the tag sets can be a problem.

WordNet. WordNet is constantly under development and the stages are distinct for all languages. Limitations, like a missing synonym or an incomplete related word list, becomes a semantic similarity match limitation.

Synonymy and Related Words. When an original word from a text unit is substituted by a synonym its truth value is expected to be maintained. The same is expected when a word from the related word list is used, but unfortunately that is not guaranteed. Even if the resulting sentence yields a true value, the original sentence has been compromised. The proposed approach is agnostic to such cases. A possible evolution to avoid this issue would be to select a more restrict and *correct* related word list.

Uniform Sense Relation Distance. As described in Section 3, Resnik [25] points that the assumption of uniform distance is not entirely correct. A possible solution could be to select a more restrict and *correct* related word list.

Antonymy. Antonymy seems surprisingly difficult to account in the proposed approach. The antonym of a word x is sometimes $not - x$, but not always, as in the case of *rich* and *poor*. Antonymy is a real problem when two text units differ only in some key words that are actually antonyms, which may result in an incorrect semantic similar match.

Negation. Negation is very difficult to account since it can change completely the sense of an entire text through the usage of a single word, like *not* as in *I do not won a car*. The usage of double negation is also difficult to account, since a double negation can resolve to a negative, such as in *I can't get no sleep*, or to a positive, such as in *I don't disagree*.

Text Size. Text size may be a problem since a text may not carry enough information to assert about similarity or it can carry to much information leading to an incorrect match. There are no real tests performed in order to assert about this issue nor *small text* or *large text* are defined.

False Negatives. When compared with the common lexical approaches, false negatives are expected to decrease since there is an approximation between the text units through the translation of the original words into a set of senses. Since there are more possible matches, this increases the probability of text matching when the text units hold the same sense. Nevertheless, false negatives may occur, and most certainly will, due to some of the problems and issues stated above, like text size.

False Positives. The same reason false negatives are expected do decrease may lead to the thinking that the number of false positives will also increase. When compared with the common lexical approaches, an increase of false positives is not expected, since there is a separation of the text units when their senses are distinct. Nevertheless, false positives may occur due to some of the problems and issues stated above, like antonymy.

Some of the issues described above are just hypotheses since there were no real tests performed on the presented semantic similarity match. Nevertheless, they must be taken into account when implementing the proposed approach.

6 Conclusion

This work presented a semantic text similarity measure aimed to guarantee Uniqueness when used as an *approximate join* operator in the duplicate detection task of a data cleaning process.

Text matching in data cleaning is commonly performed through lexical and phonetic matching mechanisms, which ignores completely the sense of the text, performing a, somewhat, *blind* matching. Previous works on semantic similarity have addressed several linguistic features, but they always had a specific goal, as within a taxonomy or centred in short text passages. The usage of machine learning techniques over annotated corpora has also been used, a feature this approach does not explore since it is, usually, unfeasible on a real data cleaning scenario. All the previous works had restrict focus and were never aimed to data cleaning as a duplicate detection text matching operator, which is precisely the approach proposed here. Therefore, the semantic similarity presented here brings an original approach into data cleaning. By building sense matrices based on linguistic features a semantic representation of the text is achieved. This representation is used as the input of the computation of the semantic similarity, which is expected to approximate texts with the same sense and to separate texts with distinct senses.

6.1 Future Work

Some questions and problems have been identified and, as a future work, some should be further studied. For instance, the selection of a more restrict and correct related word list, possibly using GraSM, or the inclusion of other semantic relations would probably yield in better results.

Real tests should be performed using the prototype in order to find the answers for the identified open questions. Prototype evolutions, like using a better selection of related word list as stated above, should also be tested.

The evaluation of the proposed approach against other works should be performed using a classified corpus in terms of text similarity, allowing to compare the scores of correct hits and erroneous false positives and false negatives.

References

[1] José Barateiro and Helena Galhardas. A survey of data quality tools. 2005.

[2] Regina Barziay, Kathleen R. McKeown, and Michael Elhadad. Information fusion in the context of multi-document summarization. pages 550–557, 2002.

[3] Andrew Borthwick. The choicemaker 2 record matching system. 2004.

[4] Martin Buechi, Andrew Borthwick, Adam Winkel, and Arthur Goldberg. Cluemaker: A language for approximate record matching. 2003.

[5] Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, 2005.

[6] Francisco Couto, Mário Silva, and Pedro Coutinho. Measuring semantic similarity between gene ontology terms. *Data & Knowledge Engineering*, 61:137–152, 2007.

[7] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection: A survey. *IEEE Transactions On Knowledge And Data Engineering*, 19(1):1–16, 2007.

[8] Helena Galhardas, Daniela Florescu, Dennis Shasha, Eric Simmon, and Cristian-Augustin Saita. Declarative data cleaning: Language, model, and algorithms. *Proceedings of the 27th VLDB Conference*, 2001.

[9] Luis Gravano, Panagiotis G. Ipeirotis, H. V. Jagadish, Nick Koudas, S. Muthukrishnan, and Divesh Srivastava. Approximate string joins in a database (almost) for free. *Proceedings of the 27th VLDB Conference*, 2001.

[10] Lifang Gu, Rohan Baxter, Deanne Vickers, and Chris Rainsford. Record linkage: Current practice and future directions. 2002.

[11] Vasileios Hatzivassiloglou, Judith L. Klavans, and Eleazar Eskin. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. 1999.

[12] Katherine Herbert, Narain Gehani, William Piel, Jason Wang, and Cathy Wuq. Bio-ajax: An extensible framework for biological data cleaning. *SIGMOD Record*, 33(2):51–57, 2004.

[13] Won Kim, Byoung-Ju Choi, Eui-Kyeong Hong, Soo-Kyung Kim, and Doheon Lee. A taxonomy of dirty data. 2003.

[14] Ralph Kimball and Margy Gross. *The Data Warehouse Toolkit*. Whiley Computer Publishing, 2002.

[15] Nick Koudas, Amit Marathe, and Divesh Srivastava. Flexible string matching against large databases in practice. 2004.

[16] Thomas Landauer, Peter Foltz, and Darrell Laham. An introduction to latent semantic analysis. 1998.

[17] Chen Li Liang Jin and Sharad Mehrotra. Efficient record linkage in large data sets. 2003.

[18] P. Lord, R. Stevens, A. Brass, and C. Goble. Semantic similarity measures as tools for exploring the gene ontology. 2003.

[19] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1994.

[20] George Miller. Wordnet: A lexical database for english. *Communications Of The Acm*, 38(11):39–41, 1995.

[21] George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Introduction to wordnet: An on-line lexical database. 1993.

[22] Dan I. Moldovan and Rada Mihalcea. Improving the search on the internet by using wordnet and lexical operators. 1999.

[23] Heiko Muller and Johann-Christoph Freytag. Problems, methods, and challenges in comprehensive data cleansing. 2003.

[24] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. 2000.

[25] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[26] Peter Wiemer-Hastings. Latent semantic analysis. 2004.

[27] Ian Witten and Eibe Frank. *Data Mining, Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Else, 2005.