MINING POSTAL ADDRESSES

José Carlos Cortizo Pérez
Universidad Europea de Madrid / AINetSolutions

José María Gómez Hidalgo
R+D Department, Optenet

Yaiza Temprado, Diego Martín, Federico Rodríguez
Universidad Europea de Madrid

ABSTRACT

This paper presents FuMaS (Fuzzy Matching System), a system capable of an efficient retrieval of postal addresses from noisy queries. The fuzzy postal addresses retrieval has many possible applications, ranging from datawarehouse de-dumping, to the correction of input forms, or the integration within online street directories, etc.

This paper presents the system architecture along with a series of experiments performed using FuMaS. The experimental results show that FuMaS is a very useful system when retrieving noisy postal addresses, being able to retrieve almost 85% of the total ones. This represents an improvement of the 15% when comparing with other systems tested in this set of experiments.

KEYWORDS

Text mining, Approximate Information Retrieval, Record linkage

1. INTRODUCTION

Information integration is a very active research area inside the database and data mining fields [1], [2]. Integrating different kind of information allows obtaining a more precise and complete view of the world, and to elicit additional knowledge about it. One of the most common problems when integrating big databases is to detect portions of several registers that refer to the same entity, registers that will be after cleaned or merged. Detecting several registers referring to the same entity may be easy if the key that identifies the entity is the same and it is complete in all the registers, but this is rarely the case. Most often, the key used in different databases is not the same; for instance, a person is identified by their name in a database, and by the Social Security number in another one. An additional difficulty is that the keys can be noisy, making hard to guess the entity they refer to.

The expression ``record linkage'' [3], [4] makes reference to the utilization of algorithmic techniques in order to find registers that, if not exactly identifying the same entity, they refer to it. This concept has been phrased quite heterogeneously in the literature, with names including [4]: identity heterogeneity [3], identity identification [6], instance identification [7], merge/purge [8], entity reconciliation [5], and list and data cleaning [9].

There is a wide range of applications of record linkage, especially in management environments: from Customer Relationship Management, fraud detection, data warehousing, finding registers in several medical information sources for a given patient [10], etc.

The overall approach of record linkage algorithms is to calculate a similarity coefficient between each record pair, involving a very expensive process computationally, in the order of $O(n_2)$ (being n the number of records in each of the two databases). The similarity coefficient used has been traditionally a version of an editing distance [11], like the Levenhstein distance [12] or the Smith-Waterman algorithm [13]. These distances are based on computing the minimum number of edition operations required to transform one string into another. The most usual operations are substitution, insertion and elimination of characters, and they are often given the same weight or importance. There is also a wide range of algorithms for the computation of diction distances [14] [15] that may be used in the record linkage process, along with in other applications like noisy signal processing [16] [12] [17], information retrieval with errors in text [18] [19] [20] [21], and computational biology [22] [23] [24].

We approach a problem of postal addresses retrieval, as an application of the record linkage problem. The next section describes our problem in detail. The system's architecture is discussed in the next section. Section 4 presents the performed experiments and discusses the results. Section 5 concludes the paper and sketches our future work.

2. PROBLEM DESCRIPTION

This paper deals with the record linkage problem applied to the problem of fuzzy retrieval of postal addresses. Postal addresses can be considered structured information, as they contain well-known and delimited fields like kind and name of the street, postal code, town, etc. However, a postal address can be written several ways. For instance, let be the following Spanish postal address: ``Calle Santa María Magdalena, 5, 4º B, 28900, Madrid", where ``Calle" means street, ``Santa María Magdalena" is the street name and means ``Saint Mary Magdalene", 5 is the street number, 4 is the floor, B is the door, ``28900" is the postal code and "Madrid" is the city. This particular postal address can be written, for instance, as:

- C/ Santa Maria Magdalena, n. 5, 4B, 28900, Madrid
- C/ Sta Maria Magdalena, n. 5, 4B, 28900, Madrid
- C/ Santa M. Magdalena, n. 5, 4B, 28900, Madrid

These three previous examples show how a postal address can be correctly written several ways by using abbreviations, synonyms, contractions, and other linguistic resources that allow to inputs some ambiguity or variability for the same concepts. Apart from these correct variations, it is also possible to find mistaken occurrences of the same address, due to typos, mistakes or omitted data, like or even combining different kinds of mistakes. All these examples show how hard it can be to find a correct postal address, given one entered by a user. This paper presents a system that, considering the mistakes described and the several correct variants of a postal address, tries to find a canonical and correct way to write a given postal address.

3. DESCRIPTION AND SYSTEM ARCHITECTURE

FuMaS system architecture is a three-layer architecture, each of them representing an abstraction level. FuMaS receives a postal address as input and returns a ranking of possible postal addresses ordered according to a certain quality measure. As can be seen in Figure 1.A, a postal address given as input to the system, will pass through the next levels:

- **Tokenizer**: It can be considered as a preprocessing step, more than an abstraction level (that's the reason why only 3 abstraction levels have been mentioned before). This level will get an input string containing a postal address and will detach the constituent elements: road type, road name, house number, postal code, town, etc.
- **Address level**: It represents the more abstract layer and manages the coherences among all the possible fields composing a postal address. This includes the associations between postal codes and towns, road names and portal number, etc. Furthermore, it is the responsible of generating the output of the system: a decremental ordered list of possible postal addresses.
- **Phrase level**: This layer manages list of words (phrases). Its main goal is to correct mistakes due to elimination, insertion, transposition or substitution of words inside a phrase.
- **Word level**: Is the lower level layer and manages the mistakes produced by insertion, elimination, transposition and substitution of letters inside words, and also manages the normalization of words. FuMaS contemplates two kind of normalization: phonetic normalization and conceptual normalization.

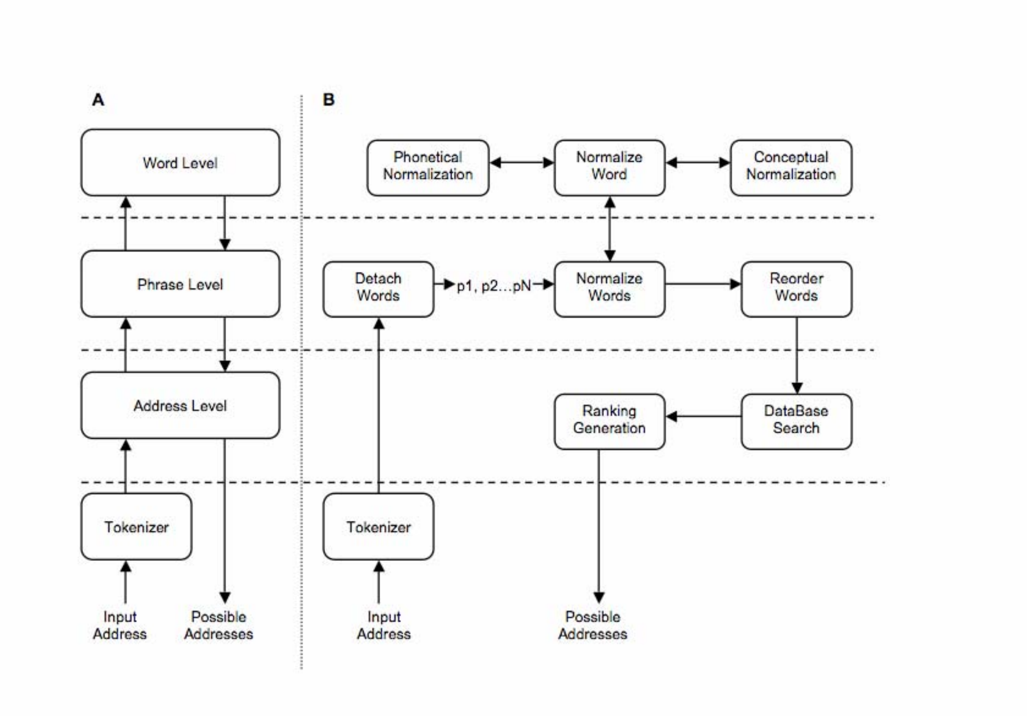

Figure 1. Conceptual view of FuMaS architecture (subfigure A) and an abstraction level module decomposition (subfigure B). As can be seen, FuMaS presents a 3 abstraction levels based architecture, which are expert when working on a certain kind of entities (addresses, phrases and words).

This architecture, based on a three-abstraction layers model, allows a more simple approach to a complicated problem. Each of the layers can be modified, adding more or less complexity taking care about the performance of the system or execution time restrictions. This modular architecture has allowed obtaining a fully functional system in a relatively short period of time, which represents the base and initial framework for a set of research lines.

4. EXPERIMENTS

In order to test the system, a collection of valid postal addresses has been collected (C_{PA}). Then, for each valid postal address, we have generated 3 different non-valid addresses (W_{PA}) by adding different kinds of mistakes, for example:

- **Phonetic mistakes**: This kind of mistakes is very related to the characteristics of Spanish, where, for instance, words containing 'b' and 'v' are pronounced in the same way, and the 'h' is never pronounced.
- **Conceptual mistakes**: Using abbreviations, synonyms or related words (e.g. colonel and captain).
- **Typographic failures**: Random substitution, elimination, insertion or transposition of characters (typically mistakes due to OCR failures or when introducing the information using a keyboard).
- **Incomplete addresses**: Absence of certain data of the input postal address (e.g. postal code, town, address type).
- **Replacement failures**: Appearance of certain incorrect information (e.g. wrong postal code).

For each element $x \in W_{PA}$ there exist an element $y \in C_{PA}$ which is the valid postal address related to the wrong one. Then, each pair *(x, y)* represents a case to test the system.

This collection has been used to test different aspects of the system and also the global performance of the whole system. In order to study certain aspects of the system, three sets of experiments have been designed:

- Study of the distance algorithm to use.
- Study of the normalization step impact on the system.
- Comparative versus other equivalent systems.

4.1 Distance Algorithms

For the study of which distance algorithm to use, five different algorithms have been considered:

- SOUNDEX [24].
- Dice Similarity [25].
- Cosine Distance [26].
- Levenshtein Distance [12].
- Q-Gram [27].

After a small set of experiments, SOUNDEX has been ruled out because it has been designed for English names and when using Spanish words the similarity values has no real sense. Cosine similarity only uses the information of the number of equal words in the given 2 input strings, which is not very useful for this problem. Dice's similarity works similar to cosine similarity and was ruled out soon by the same reason. Then, the only two algorithms that really seems to fit to this problem are Levenshtein distance and Q-Grams[1]. The first set of experiments evaluates the adjustment of each algorithm to the given problem.

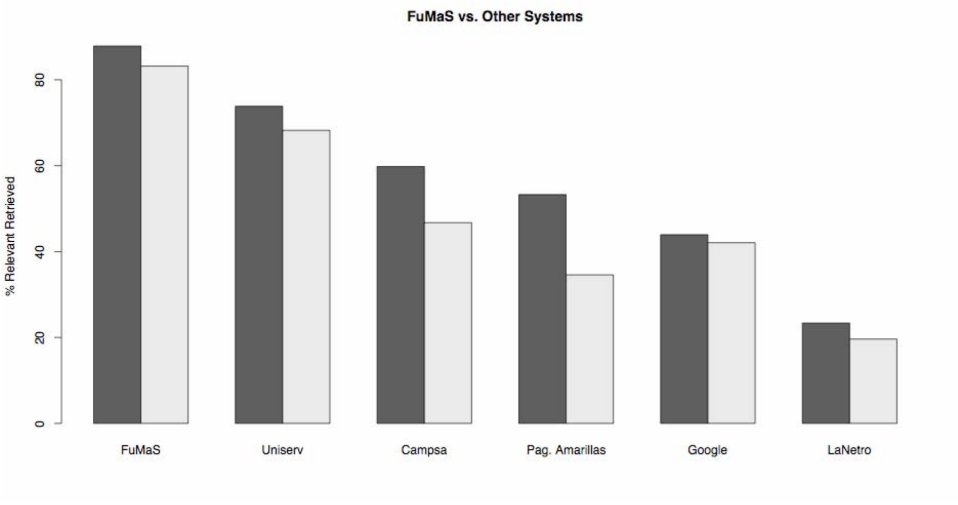

Figure 2. Comparison of relevant addresses retrieved by FuMaS, Uniserv and 4 street directories that integrates some techniques to correct the user inputs. Dark bars show the results for recall 10 and light ones show the results for recall 1

Levenshtein and Q-gram algorithms receives two strings and returns a similarity value, $0 < S < 1$, representing how similar are the given two strings. 0 means the two strings are totally different and 1 represents that the two given strings are identical. The best algorithm will be the one that best correlates high similarity values with relevant addresses. Figure 2 show the number of relevant and non-relevant addresses grouped by similarity ranges of values. As can be seen, Q-Gram gives lower similarity values to non-relevant addresses, and the total number of relevant addresses (65) retrieved with a similarity value higher than 0.8 is higher than the same value for the Levenshtein distance (60). This is the reason why Q-gram has been selected as the string distance algorithm for FuMaS.

Q-grams are used in approximate string matching [28] by sliding a window of length q over the characters of a string to create a number of q length grams for matching a match is then rated as number of q-gram matches within the second string over possible q-grams resulting in a certain metric of similarity. The most q-grams appearing in the two strings, the most similar are the referred two strings.

[1] The implementation of edit distance algorithms used for this paper are extracted from http://sourceforge.net/projects/simmetrics/

4.2 Normalization vs. Non-Normalization

The second set of experiments has been designed to test the effectiveness of the normalization algorithm used to preprocess the postal addresses. Using Q-Gram as the editing distance algorithm, we have repeated the same experiments using normalization and without normalization. The results of this experiments, show that when applying normalization, a higher number of relevant addresses are retrieved while a lower number of non relevant ones are retrieved by the system, which is a characteristic completely desired.

4.3 Comparative vs Other Systems

The fuzzy matching of postal address is a well known problem and that is the reason why there exist some system doing the same and some other systems that integrates techniques for allowing a disambiguation of the input postal addresses. FuMaS has been compared versus some state of the art systems:

- Uniserv (http://www.uniserv.com): a system specifically designed to validate and correct postal addresses.
- A selection of street directories that integrate some kind of techniques to correct the input addresses
    - Campsa Guide (http://www.guiacampsa.com)
    - Google Maps (http://maps.google.com)
    - La Netro (http://callejero.lanetro.com)
    - Paginas Amarillas (http://www.paginas-amarillas.es)

As can be seen in Figure 2, FuMaS is able to retrieve much more addresses (Recall 10: 87,85%, Recall 1: 83,18%) than any other tested system (more than a 14% than the best one of the other systems, Uniserv). Another interesting point is that the total number of correctly corrected addresses is quite high, which means that with a better tuning, FuMaS would be able to correct near the 100% of the addresses.

5. CONCLUSIONS AND FUTURE WORK

This paper presents FuMaS, a system able to effectively retrieve of noisy queries (postal addresses). Experimental results show that FuMaS, in spite of it's novelty, is able to retrieve almost 85% of the erroneous postal addresses given as inputs to the system, almost 15% more than any other system tested.

FuMaS is far from being a finished and closed system; moreover, it has developed as a framework to study some novel techniques that can be applied to the approximate retrieval of semi-structured information. Its modular architecture allows many improvement lines. This paper shows the first big stone in a new research line that tries to solve the approximate information retrieval in semi-structured data, first on the postal addresses domain but extensible to other related areas due to the flexibility and modularity of FuMaS. Regarding this subject, it is worth mentioning that from the 3 abstraction levels in FuMaS, two of them (phrase and word levels) are general enough to be reused as is in other application fields.

FuMaS opens up a sort of future actuation lines, including studies of usual mistakes, d Development of more advanced and sophisticated phonetic-translation algorithms, creation, reusing and integration of lexical resources for a conceptual normalization of some parts of certain addresses, development of approximate indexing methods for approximate string matching in databases, etc.

FuMaS is an effective system when dealing with noisy postal addresses and represents a framework for the study of a set of research and development lines supported by the previous mentioned points.

ACKNOWLEDGMENTS

The research described in this paper has been partially supported by the Spanish Ministry of Education and Science and the European Union from the European Regional Development Fund (ERDF), TIN2005-08988-C02-01 and TIN2005-08988-C02-02, and also by the Madrid autonomous region, IV PRICIT, S-0505/TIC/0267.

REFERENCES

Batini, C. et al, 1986. A comparative analysis of methodologies for database schema integration. *In ACM Computer Surveys*, Vol. 18, No. 4,pp 323-364.

Hernandez, M.A., Stolfo, S.J. 1998. Real-world data is dirty: Data cleansing and the merge/purge problem. *In Data Mining and Knowledge Discovery*, Vol. 2, No. 4, pp. 9-37.

Fellegi, I.P., Sunter, A.B. 1969. A theory for record linkage. *In Journal of the American Statistical Association*, Vol. 64, No. 328, pp. 1138-1210.

Gu, L., et al, Record linkage: Current practice and future directions.

Dey, D., et al, A distance-based approach to entity reconciliation in heterogeneous databases. *In IEEE Transactions on Knowledge and Data Engineering*, Vol. 14, No. 3, pp. 567-582.

Lim, E.P., et al, 1996. Entity identification in database integration. *In Information Sciences*, Vol. 89, No. 1, pp. 1-38.

Wang, Y.R., Madnick, S.E. 1989, The inter-database instance identification problem in integrating autonomous systems. *Proceedings of the Fifth International Conference on Data Engineering*, pp. 46-55.

Hernandez, M.A., Stolfo, S.J. 1995, The merge/purge problem for large databases. *In Proceedings of the SIGMOD Conference*, pp. 127-138.

Christen, P., Churches, T. 2005, febrl-freely extensible biomedical record linkage. *Sourceforge.net*.

Sauleau, E.A., et al. 2005, Medical record linkage in health information systems by approximate string matching and clustering. *In BMC Medical Information Decision Making*, Vol. 32, No. 5, pp. 5-32.

Navarro, G., Raffinot, 2002, M. *Flexible pattern matching in strings – Practical on-line search algorithms for texts and biological sequences*. Cambridge University Press.

Levenshtein, V. 1966, Binary codes capable of correcting deletions, insertions and reversals. *In Soviet Physics* Doklady, Vol. 10.

Smith, T.F., Water, M.S., Identification of common molecular subsequences. *Journal of Molecular Biology*.

Aho, A. 1990, Algorithms for finding patterns in string. In Handbook of Theoretical Computer Science: Algorithms and Complexity. MIT Press, pp. 255-300.

Navarro, G., A guided tour to approximate string matching. *ACM Computing Surveys*, Vol. 33, No. 1, pp. 31-88.

Chollet, G., 1994, Automatic speech and speaker recognition: overview, current issues and perspectives. *Fundamentals of speech synthesis and speech recognition*. pp. 129-147.

Vintsyuk, T.K., 1968, Speech discrimination by dynamic programming. *In Cybernetics and System Analysis*.

Baeza-Yates, R., Navarro, G. 1997, A practical index for text retrieval allowing errors, pp. 273-282.

Navarro, G. et al, 2001. Matchsmile: A flexible approximate matching tool for personal names searching. *In Proceedings of the SBBD'01*, pp. 273-282.

Wagner, R.A., Fischer, M.J. 1974, The string-to-string correction problem. *In Journal of the ACM*.

Wagner, R.A., Lowrance, R., 1975, An extension of the string-to-string correction problem. *In Journal of the ACM*, Vol. 22, No. 2, pp. 177-183.

Gusfield, D., 1997, *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press, New Yotk, USA.

Needleman, S., Wunsch, C., 1970, A general method applicable to the search for similarities in the amino acid sequence of two proteins. *In Journal of Molecular Biology*, Vol. 48, pp. 444-453.

Szucs, L.D., Hargreaves, S. 1996, *The Source: A guidebook of American Genealogy*. Ancestry.com.

Kondrak, G.,et al. Cognates can improve statistical translation models. P*roceedings of HLT-NAACL 2003*, pp. 46-48.

Salton, G., et al, 1975, A vector space model for automatic indexing. *In Communications of the ACM*.

Gravano, L. et al, 2001, Using q-grams in a DBMS for approximate string processing. *In IEEE Data Engineering Bulletin*, Vol. 24, No. 4, pp. 28-34.