A Comparison of Personal Name Matching: Techniques and Practical Issues

Peter Christen
Department of Computer Science, The Australian National University
Canberra ACT 0200, Australia
Peter.Christen@anu.edu.au

Abstract

Finding and matching personal names is at the core of an increasing number of applications: from text and Web mining, search engines, to information extraction, deduplication and data linkage systems. Variations and errors in names make exact string matching problematic, and approximate matching techniques have to be applied. When compared to general text, however, personal names have different characteristics that need to be considered. In this paper we discuss the characteristics of personal names and present potential sources of variations and errors. We then overview a comprehensive number of commonly used, as well as some recently developed name matching techniques. Experimental comparisons using four large name data sets indicate that there is no clear best matching technique.

1. Introduction

Increasingly large amounts of data are being created, communicated and stored by many individuals, organisations and businesses on a daily basis. At lot of this data contains information about people, for example e-mails, customer and patient records, news articles, or business and political memorandums. Even most scientific and technical documents contain details about their authors. Personal names are often used to search for documents in large collections (like Web searches, retrieval of patient records, or bibliographic searches of books and articles). Names are also important pieces of information when databases are deduplicated and when data sets are linked or integrated and no unique entity identifiers are available [5].

Personal names have characteristics that makes them different from general text. While there is only one correct spelling for many words, there are often several valid variations for personal names, for example 'Gail', 'Gale' and 'Gayle'. People also frequently use (or are given) nicknames in daily life, for example 'Bill' rather than the more formal 'William'. Personal names are also heavily influenced by a person's cultural background, and in certain situations people do change their names. All these issues make matching of personal names more challenging compared to matching of general text [2, 20].

As names are often recorded with different spelling variations, applying exact string matching leads to poor results. Many different techniques for approximate string matching have been developed in the last four decades [12, 17, 21, 25], and new techniques are still being invented [11, 15]. Most techniques are based on pattern matching, phonetic encoding, or a combination of these two approaches.

The contributions of this paper are a detailed discussion of the characteristics of personal names and possible sources of variations and errors in them, an overview of a range of name matching techniques, and a comparison of their performance using several large real world data sets containing personal names. A longer, more detailed version of this paper is available as [4].

2. Personal name characteristics

Even when only considering the English-speaking world, a name can have several different spelling forms for a variety of reasons. In the Anglo-Saxon region and most other Western countries, each person has usually a given name, an optional middle name, and a surname or family name [20]. Both 'Gail Vest' and 'Gayle West' might refer to the same person, while 'Tina Smith' might be recorded in the same database as 'Christine J. Smith' and as 'C.J. Smith-Miller'. People change their name over time, most commonly when they get married. Compound names are often used by married women, while in certain countries husbands can take on the surname of their wives.

In daily life, people often use (or are given) nicknames. These can be short forms of their given names (like 'Bob' for 'Robert', or 'Liz' for 'Elizabeth'), they can be variations of their surname (like 'Vesty' for 'Vest') or they might relate to some life event, character sketch or physical characteristics of a person [2]. While having one given and one middle name is common for Anglo-Saxon names, several

European countries favour compound given names instead, for example 'Hans-Peter' or 'Jean-Pierre'. In general, there are no legal regulations of what constitutes a name [2].

In today's multi-cultural societies and worldwide data collections (e.g. global online businesses, or international crime and terrorism databases), the challenge is to be able to match names coming from different cultural backgrounds. For Asian names, for example, there exist several transliteration systems into the Roman alphabet [20], the surname traditionally appears before the given name, and frequently a Western given name is added. Hispanic names can contain two surnames, while Arabic names are often made of several components and contain various affixes.

An early study [8] on spelling errors in general words found that over 80% of errors were single character errors (inserts, deletes, or substitutions). Other studies [12, 16, 23] reported similar results. However, in a study [9] that looked at patient names within hospital databases, different types and distributions of errors were found (with 36%, insertion of an additional name word, initial or title, were the most common errors, and single character errors accounted for only 39% of all errors in this study). Thus, there seem to be significant differences between general text and personal names, which have to be considered when name matching algorithm are being developed and applied.

In [16] character level (or non-word) misspellings are classified into (1) typographical errors, where it is assumed that the person doing the data entry does know the correct spelling of a word but makes a typing error (e.g. 'Sydeny' instead of 'Sydney'); (2) cognitive errors, assumed to come from a lack of knowledge or misconceptions; and (3) phonetic errors, coming from substituting a correct spelling with a similar sounding one. The combination of phonetical and spelling variations, as well as potentially totally changed name words, make name matching challenging.

2.1 Sources of name variations

Besides the variations in personal names discussed above, the nature of data entry [16] will determine the most likely types of errors and their distributions.

- When handwritten forms are scanned and optical character recognition (OCR) is applied [12, 23], the most likely types of errors will be substitutions between similar looking characters (like 'q' and 'g'), or substitutions of one character with a similar looking character sequence (like 'm' and 'r n', or 'b' and 'l i').
- Manual keyboard based data entry will mainly result in wrongly typed neighbouring keys (for example 'n' and 'm', or 'e' and 'r').
- Data entry over the telephone (for example as part of a survey study) is a confounding factor to manual keyboard entry. The person doing the data entry might not request the correct spelling, but rather assume a default spelling (which is based on the person's knowledge and cultural background).
- Limitations in the maximum length of input fields can force people to use abbreviations, initials only, or even disregard some parts of a name.
- Finally, people themselves sometimes report their names differently depending upon the organisation they are in contact with, or deliberately provide modified or wrong names.

If names from different sources are to be matched, like in text mining or data linkage systems [5], then the variability and error distributions will likely be larger than if all the names come from one source only. This will also limit the use of adaptive name matching algorithms [7] that are trained to deal with certain types of variations and errors.

As many personal names have several valid spelling variations, it is often not possible to disregard a name as wrong if it is not found in a dictionary of known names. When matching names, one has to deal with legitimate name variations (that should be preserved and matched), and errors introduced during data entry and recording (that should be corrected) [2]. The challenge lies in distinguishing between these two sources of variations.

3. Matching techniques

Name matching can be defined as *the process of determining whether two name strings are instances of the same name* [20]. As name variations and errors are quite common [9], exact name string comparison will not result in good matching quality. Rather, an approximate measure of how similar to names are is desired. Generally, a normalised similarity measure between 1.0 (two names are identical) and 0.0 (two names are totally different) is used.

The two main approaches for matching names are phonetic encoding and pattern matching. Many techniques have been developed for both approaches, and several techniques combine the two with the aim to improve the matching quality. In the following we give a brief overview of the most commonly used as well as several recently proposed new techniques. More detailed descriptions are given in [4].

3.1 Phonetic encoding

Common to all phonetic encoding techniques is that they convert a name string into a code according to how a name is pronounced (i.e. the way a name is spoken). Naturally, this process is language dependent. Most techniques have been developed mainly with English in mind.

- *Soundex* [13, 17] is the best known phonetic encoding algorithm. It keeps the first letter and converts the rest into numbers according to an encoding table.
- *Phonex* [17] is a variation of Soundex that aims to improve the encoding quality by pre-processing names according to their English pronunciation.
- *Phonix* goes a step further than Phonex and applies more than one hundred transformation rules on groups of letters [10]. Some of these rules are limited to the beginning of a name, some to the end, others to the middle, and some will be applied anywhere.
- *NYSIIS (New York State Identification Intelligence System)* is based on transformation rules similar to Phonex and Phonix, but it returns a code only made of letters.
- *Double-Metaphone* [22] aims to better account for non-English words, like European and Asian names. The algorithm contains many rules that take the position within a name, as well as previous and following letters, into account. Similar as NYSIIS, it returns a code only made of letters.
- *Fuzzy Soundex* is based on q-gram substitutions [13] and combines elements from other phonetic encoding algorithms. Similar to Phonix, it has transformation rules that are limited to the beginning or end of a name, or that are applicable anywhere.

When matching names, phonetic encoding can be used as a filtering step (called *blocking* in data linkage [5]), i.e. only names having the same phonetic code will be compared using a computationally more expensive pattern matching algorithm. Alternatively, exact string comparison of the phonetic encodings can be used.

3.2 Pattern matching

Pattern matching techniques are commonly used in approximate string matching [12, 14, 19], which has widespread applications, from data linkage and duplicate detection [2, 5, 7, 24], information retrieval [11, 15, 25], correction of spelling errors [8, 16, 23], to bio- and health informatics [9].

- *Levenshtein* or *Edit distance* [19] is defined as the smallest number of edit operations (inserts, deletes and substitutions) required to change one string into another. It is calculated using a dynamic programming algorithm [14].
- *Damerau-Levenshtein distance* is a variation of edit distance where a transposition of two characters is also considered to be an elementary edit operation [8, 19].
- *Bag distance* [1] has recently been proposed as a cheap approximation to edit distance (a *bag* is defined as a multi-set of the characters in a string). The bag distance is always smaller or equal to the edit distance.
- *Smith-Waterman distance* [18] was developed to find optimal alignments between biological sequences. It is based on a dynamic programming approach and allows gaps as well as character specific match scores.
- *Longest common sub-string (LCS)* [9] repeatedly finds and removes the longest common sub-string in the two strings compared, up to a minimum lengths.
- *Q-grams* [16] are sub-strings of length q. A similarity measure is calculated as the number of q-grams in common divided by the minimum, average or maximum number of q-grams in two strings.
- *Positional q-grams* are an extension to q-grams with positional information added (the locations of q-grams within a string), and only q-grams that are within a maximum distance from each other are matched.
- *Skip-grams* [15] are based on the idea of not only forming bigrams (2-grams) of two adjacent characters, but also bigrams that skip one or more characters. Certain skip-grams have properties that relate to character edits like inserts, deletes and substitutions.
- *Compression* based similarity calculations have recently been investigated [6] for use in clustering of biological sequences, optical character recognition, and music. The basic idea is to use the length of the compressed strings (using a standard compressor like *Zip* or *BZ2*) to calculate a similarity measure.
- *Jaro* [24] is an algorithm commonly used in data linkage systems [5]. A similarity measure is calculated using the number of common characters (same characters that are within half the length of the longer string) and the number of transpositions.
- *Winkler* [24] (or Jaro-Winkler) improves upon the Jaro algorithm by applying ideas based on studies [23] that found fewer errors typically occur at the beginning of names. The algorithm increases the Jaro similarity measure for up to four agreeing initial characters.

3.3 Combined techniques

Two techniques combine phonetic encoding and pattern matching with the aim to improve the matching quality.

- *Editex* [25] combines edit distance based methods with the letter-grouping techniques of Soundex and Phonix.

	Pairs	Singles
Midwives given names	15,233	49,380
Midwives surnames	14,180	79,007
Midwives full names	36,614	339,915
COMPLETE surnames	8,942	13,941

Table 1. Number of name pairs and single names in test data sets used for experiments.

Its edit costs are 0 if two letters are the same, 1 if they are in the same letter group, and 2 otherwise. Comparison experiments [25] showed that Editex performed better than edit distance, q-grams, Phonix and Soundex on a database containing around 30,000 surnames.

- *Syllable alignment distance* [11] is based on the idea of matching two names syllable by syllable, rather than character by character. It uses Phonix transformations as a preprocessing step and then applies rules to find the beginning of syllables. The distance between two strings of syllables is then calculated using an edit distance based approach.

4 Experiments and discussion

Here we present and discuss the results from a series of comparison experiments using four name data sets. The aim was to investigate which techniques achieve the best matching quality for different personal name types. All name matching techniques were implemented in Python as part of the *Febrl* open source data linkage system [5].

4.1 Name data sets

Three of the test data sets were based on given- and surnames extracted from a midwives database [3]. The deduplication status in this data (indicating which records correspond to the same women) allowed us to extract true name pairs (known matches). From these we removed all pairs that were exact matches, leaving only pairs of names that were to some degree different. We then created a *full name* data set by concatenating given- with surnames values (separated by a whitespace). We also extracted single names from records that did not have duplicates, and randomly created name pairs (the same number as known matched pairs in order to get balanced test data sets). The fourth data set was created in a similar way using the COMPLETE name database [11, 21] by forming surname pairs from 90 randomly chosen and manually matched queries. Table 1 shows the size of our four test data sets.

4.2 Matching results

We ran a total of 123 tests on all four data sets, by applying all phonetic encoding (combined with exact string matching of the phonetic codes) and pattern matching techniques presented in Section 3 with their various ways of calculating similarity measures and other options (like padded and non-padded q-grams, longest common sub-string with minimum set to 2 and 3, etc.). For the similarity measures a threshold can be varied between 0.0 and 1.0 that influences the classification performance (name pairs with a similarity value above the threshold are classified matches, and pairs with similarity value below as non-matches). We evaluated the matching quality using the *f-measure* [4]. Here we report average f-measures over all possible threshold values as they indicate the overall quality of a matching technique. A more detailed analysis is provided in [4].

Table 2 shows the best results achieved for each of the presented techniques on all four data sets. As can be seen, no technique performs better than all others. Pattern matching clearly outperform phonetic encoding techniques. The simple Phonex technique performs better than the more complex Phonix and Double-Metaphone algorithms (despite their larger number of transformation rules). Both surname data sets seem to be harder to match than given names, which might be due to complete surname changes when women get married or divorced (the Midwives database obviously only contains women). The Jaro and Winkler techniques both perform well on all four data sets, showing their suitability for personal name data. The two approaches that combine phonetic encoding with pattern matching (Editex and syllable alignment distance) do not perform as well as expected, and neither do the recently developed skip-grams.

5 Conclusion and future work

We have discussed the characteristics of personal names and the potential sources of variations and errors in them, and we presented an overview of both pattern matching and phonetical encoding based name matching techniques. Experimental results on different real name data sets have shown that there is no single best technique available. The characteristics of the name data to be matched, as well as computational requirements, have to be considered when selecting a name matching technique.

Personal name matching is very challenging, and more research into the characteristics of both name data and matching techniques has to be conducted in order to better understand why certain techniques perform better than others, and which techniques are most suitable for what type of data. More detailed analysis into the types and distributions of errors is needed to better understand how certain types of errors influence the performance of matching techniques.

	Midwives			COMPLETE
	given names	sur-names	full names	surnames
Soundex	.342	.341	.376	.485
Phonex	**.423**	**.369**	**.499**	.579
Phonix	.339	.330	.368	**.617**
NYSIIS	.275	.296	.299	.351
DMetaphone	.304	.306	.330	.410
FuzSoundex	.327	.311	.359	.396
Leven dist	.658	.513	.737	.624
Dam-L dist	.659	.517	.739	.625
Bag dist	.597	.522	.670	.616
SWater dist	.889	.579	.802	.617
LCS-2	**.915**	.564	**.877**	.514
LCS-3	.909	.529	.866	.500
1-grams	.839	.588	.787	.627
2-grams	.885	.498	.867	.519
3-grams	.783	.442	.833	.416
Pos 1-grams	.890	.574	.724	.653
Pos 2-grams	.880	.473	.697	.508
Pos 3-grams	.768	.416	.659	.416
Skip grams	.844	.496	.825	.521
Compr BZ2	.458	.547	.568	.633
Compr Zip	.532	.456	.684	.481
Jaro	.853	**.601**	.829	**.712**
Winkler	.891	.588	.868	.707
Editex	.631	.561	.706	.646
SyllAlign dist	.656	.426	.710	.532

Table 2. Average f-measure values (best results shown boldface and worst results underlined).

Acknowledgements

This work is supported by an Australian Research Council (ARC) Linkage Grant LP0453463 and partially funded by the NSW Department of Health.

References

[1] I. Bartolini, P. Ciaccia, and M. Patella. String matching with metric trees using an approximate distance. In *SPIRE, LNCS 2476*, pages 271–283, Lisbon, Portugal, 2002.

[2] C. L. Borgman and S. L. Siegfried. Getty's synoname^{TM} and its cousins: A survey of applications of personal name-matching algorithms. *Journal of the American Society for Information Science*, 43(7):459–476, 1992.

[3] Centre for Epidemiology and Research, NSW Department of Health. New South Wales mothers and babies 2001. *NSW Public Health Bull*, 13:S-4, 2001.

[4] P. Christen. A comparison of personal name matching: Techniques and practical issues. Technical Report TR-CS-06-02, Department of Computer Science, The Australian National University, Canberra, 2006.

[5] P. Christen, T. Churches, and M. Hegland. Febrl – a parallel open source data linkage system. In *PAKDD, Springer LNAI 3056*, pages 638–647, Sydney, 2004.

[6] R. Cilibrasi and P. M. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[7] W. W. Cohen, P. Ravikumar, and E. Fienberg, Stephen. A comparison of string distance metrics for name-matching tasks. In *Proceedings of IJCAI-03 workshop on information integration on the Web*, pages 73–78, Acapulco, 2003.

[8] F. J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[9] C. Friedman and R. Sideli. Tolerating spelling errors during patient validation. *Computers and Biomedical Research*, 25:486–509, 1992.

[10] T. Gadd. PHONIX: The algorithm. *Program: automated library and information systems*, 24(4):363–366, 1990.

[11] R. Gong and T. K. Chan. Syllable alignment: A novel model for phonetic string search. *IEICE Transactions on Information and Systems*, E89-D(1):332–339, 2006.

[12] P. A. Hall and G. R. Dowling. Approximate string matching. *ACM Computing Surveys*, 12(4):381–402, 1980.

[13] D. Holmes and C. M. McCabe. Improving precision and recall for soundex retrieval. In *Proceedings of the IEEE International Conference on Information Technology – Coding and Computing (ITCC)*, Las Vegas, 2002.

[14] P. Jokinen, J. Tarhio, and E. Ukkonen. A comparison of approximate string matching algorithms. *Software – Practice and Experience*, 26(12):1439–1458, 1996.

[15] H. Keskustalo, A. Pirkola, K. Visala, E. Leppanen, and K. Jarvelin. Non-adjacent digrams improve matching of cross-lingual spelling variants. In *SPIRE, LNCS 2857*, pages 252–265, Manaus, Brazil, 2003.

[16] K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.

[17] A. Lait and B. Randell. An assessment of name matching algorithms. Technical report, Deptartment of Computer Science, University of Newcastle upon Tyne, 1993.

[18] A. E. Monge and C. P. Elkan. The field-matching problem: Algorithm and applications. In *Proceedings of ACM SIGKDD*, pages 267–270, Portland, 1996.

[19] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[20] F. Patman and P. Thompson. Names: A new frontier in text mining. In *ISI-2003, Springer LNCS 2665*, pages 27–38.

[21] U. Pfeifer, T. Poersch, and N. Fuhr. Retrieval effectiveness of proper name search methods. *Information Processing and Management*, 32(6):667–679, 1996.

[22] L. Philips. The double-metaphone search algorithm. *C/C++ User's Journal*, 18(6), 2000.

[23] J. J. Pollock and A. Zamora. Automatic spelling correction in scientific and scholarly text. *Communications of the ACM*, 27(4):358–368, 1984.

[24] W. E. Yancey. Evaluating string comparator performance for record linkage. Technical Report RR2005/05, US Bureau of the Census, 2005.

[25] J. Zobel and P. Dart. Phonetic string matching: Lessons from information retrieval. In *Proceedings of ACM SIGIR*, pages 166–172, Zürich, Switzerland, 1996.