Name Matching in Law Enforcement and Counter-Terrorism

L. Karl Branting
BAE Systems, Inc.
Columbia, MD 21046, USA
karl.branting@baesystems.com

ABSTRACT

Name matching is an important task in law enforcement and counter-terrorism. This paper briefly describes the nature of the name-matching task, enumerates typical name-matching applications, explains the technical challenges posed by name-matching, and sets forth several important approaches to these technical challenges.

1. THE NAME-MATCHING TASK

Name matching is the task of recognizing when two different strings denote the same person or other named entity. Name matching is an instance of the general task of approximate string matching, a task that also arises in spell correcting [Pet80], database maintenance [NKAJ59], signal processing, text retrieval, data mining, image compression, and many other areas [Nav01]. Approximate string matching is particularly important in computational biology. For example, base-sequence similarity in genomes is an accurate measure of phylogenetic distance [SM97, Gus99].

The general task of name matching is as follows:

GIVEN:

- One or more pattern strings
- A collection of target strings

DO:

- Find each target that matches a pattern well enough that pattern and target are likely to denote the same person

Patterns are strings representing given names, and *targets* are the strings in which the patterns are sought. For example, the patterns might consist of entries in a terrorist watch list, and the targets might be entries in a passenger manifest. In this case, the name matching task would be to find passenger names that are good matches to entries in the watch list.

ICAIL Workshop on Data Mining, Information Extraction, and Evidentiary Reasoning for Law Enforcement and Counter-Terrorism, June 11, 2005 Bologna, Italy

The threshold for match similarity being "good enough that the pattern and target are likely to denote the same person" is, in general, ill-defined. Typically, the best matches are presented in descending order of similarity, leaving individual users to determine the threshold for themselves.

The name-matching task arises in a wide variety of contexts. Most of the early published work on name matching was done by the statistics community addressing the *record linkage* problem, that is, the task of identifying duplicate records in databases. For example, significant research was performed by the United States Census Bureau to identify duplicate census records [Win94, Win99]. Name matching arises in judicial case management systems when names of litigants before a court must be compared to names of persons with whom a judge has some personal or financial "interest" to prevent case assignments that would create potential conflicts of interest [Bra02, Bra03]. Name matching arises in text understanding as part of the larger problem of co-reference resolution.[1]

In the context of law enforcement and counter-terrorism, important name-matching tasks include searching name lists, such as passenger manifests, credit card transactions, or passport lists, for matches to the names of known suspects [Mil02]. For example, identifying terrorist plans by data mining intelligence reports requires recognizing when two names refer to the same individual [JLM04].

2. WHY NAME MATCHING IS HARD

The difficulty of name matching arises from the wide variety of possible orthographic variations of names. These variations can take a number of different forms.

1. Misspellings, which can arise from transcription or OCR errors.
2. Spelling variations. Phonetically identical names may have distinct legitimate spellings, *, e.g.* "Geoff" vs. "Jeff" and "McDonald" vs. "MacDonald."
3. Cross-lingual transliterations. There is often no standard representation of names from languages that do not use Roman alphabet. Typically, such names are represented phonetically, but languages that contain phonemes not occurring in English can only be approximated with the Roman alphabet. For example, there

[1]See, *, e.g.*, http://acl.ldc.upenn.edu/acl2004/refres/.

are numerous alternative Roman spellings of "Mohammed," including "Mohummed" and "Mohammet," each of which merely approximates the pronunciation of the original Arabic name.

4. Nicknames. Names may have one or more alternative forms, which are often used in informal settings. For example, alternative forms of "Elizabeth" include "Bess", "Bessie", "Bessy", "Beth", "Bette", "Bettie", "Bettie", "Betsy", and "Betsey."

5. Titles, *, e.g.*, "Usama" might be replaced by "Prince" or "Emir," words synonymous with the name's literal meaning.

6. Name changes. Some individuals change names during their lifetime, *, e.g.*, "Karol Wojtyla" vs. "John Paul II."

7. Name permutations and omissions. Cultures differ in name ordering and in the permissibility of abbreviating or omitting some names. For example, in English, surnames appear last (except in alphabetical listings, in which they appear first) whereas surnames always come first in Chinese. In English, it is common for middle names to be omitted or reduced to an initial, whereas the convention does not exist in Chinese. In Spanish, a maternal family name may come after the paternal family name, but it may be omitted or reduced to initial.[2]

8. Definite descriptions and other identifying phrases, *, e.g.*, "The Prime Minister of the United Kingdom" vs. "Tony Blair." Matching identifying phrases to names may require extensive background knowledge and inference.

3. TECHNICAL APPROACHES

The problem of identifying occurrences of pattern names within large target texts is typically addressed in two distinct stages. In the first step, *retrieval*, a computationally inexpensive procedure is used to find an initial set of candidate strings. In the second step, *similarity assessment*, a more accurate, but also more computationally expensive, procedure is applied to each element of the candidate set to determine the actual matches. Retrieval is referred to as *blocking* in the statistical record linkage literature [CRF03].

The performance of a name-matching system can be measured by standard information-retrieval metrics, including recall, precision, and f-measure. For example, recall is the proportion of target strings that should match a pattern name ("true positives") that is in fact identified as matches; precision is the proportion of target strings identified as matches that consisting of true positives.

Evaluation of the best combination of retrieval and similarity-assessment algorithms is complicated by the interaction between these two stages. For example, increasing the recall of the retrieval algorithm will fail to increase the recall of the combined system unless the similarity assessment procedure correctly evaluates the additional true positives.

[2]See http://en.wikipedia.org/wiki/Family_name for a discussion of name-formation rules in different cultures.

3.1 Similarity Assessment

The eight sources of orthographic variation listed above fall into two categories. The last five—nicknames, titles, name changes, and name permutations, omissions and identifying phrases—require culture-specific information. There appears to be little published research on approaches to acquiring, formalizing, and applying such knowledge. Instead, these variation sources appear generally to have been addressed in an *ad hoc* fashion.

The first three orthographic-variation sources—misspellings, spelling variations, and cross-lingual transliterations—have typically been approached with culture-independent string-matching techniques. These string matching technique fall, in turn, into two categories: those based on orthographic similarity, and those based on phonetic similarity.

3.1.1 Orthographic Similarity

The most common approaches to name matching focus on similarities and differences between the letters in the pattern and target. The simplest metric is "Levenshtein" distance, which counts the number of insertions, deletions, or substitutions necessary to transform one string into another.

More complex edit-distance and similarity metrics have been developed to reduce the penalty for the most probable transcription errors or transliteration inconsistencies. Needleman-Wunsch distance permits separate weights for different edit operations [Gus99]. Smith-Waterman distance determines the maximum similarity between substrings of each string [Gus99]. Affine gap cost metrics impose a higher penalty for the first in a series of insertions than for subsequent insertions [ME97]. The intuition behind affine gap cost is that the fact that an insertion occurs at al is more important than the particular length of the insertion.

The Jaro and Jaro-Winkler metrics weights errors near the beginning of strings more heavily than errors occurring later, and reduces the penalty for letters that are not too far out of place [Jar95, Win99]. Jaro-Winkler reduces penalties for errors involving characters that appear similar (*, e.g.*, "I" vs. "l") or that are close together on keyboards (*, e.g.*, "v" and "b").

The comparative evaluations, the relative accuracy of alternative similarity assessment metrics has been observed to be heavily dependent upon the composition of individual test sets [BCF$^+$03].

3.1.2 Phonetic Similarity

An alternative approach to comparing the letters in the pattern and target is to measure the similarity of the probable pronunciations of the pattern and target. The rationale behind this approach is the assumption that orthographic differences often reflect alternative transcriptions of a common pronunciation. To the extent that this assumption is correct, phonetic comparisons are likely to be more accurate than orthographic comparisons.

The most common approach to phonetic similarity assessment consists of using a hash function to map names to phonetic encodings. Two names match if their phonetic encodings are identical.

The oldest hash-based phonetic similarity function is Soundex, which was patented in 1918 and 1922 by Russell and Odell (U.S. Patents 1,261,167 and 1,435,663) and described by Knuth in [Knu75]. However, Soundex has many limitations, including including inability to handle different first letters with identical pronunciations (*, e.g.*, Soundex of "Kris" is K620, but Soundex of "Chris" is C620), truncation of long names, and bias towards English pronunciations.

A number of alternative phonetic encodings have been developed in response to the limitations of Soundex, including the following:

- NYSIIS [Taf70]
- PHONIX [Gad90]
- EDITEX [ZD96]
- Metaphone [Phi90]
- Double metaphone [Phi00]
- Phonetex [HA]

A limitation of all these approaches is use of pronunciation heuristics that are insensitive to many contextual factors that affect pronunciation. Nevertheless, the most primitive of these algorithms—Soundex—is still in use in many law enforcement and national security applications [Diz04].

A more promising approach to phonetic similarity assessment is use of text-to-speech technology, rather than context-insensitive heuristics, to produce pronunciations of the words to be compared. This approach was used for the related task of cognate identification in Aline [Kon00] and is apparently used in at least one commercial product [LG03].

3.2 Retrieval

Several alternative approaches have been applied to retrieval. The simplest approach is exhaustive matching, *i.e.*, applying the similarity measure to every pattern/target pair. This is tractable only for the smallest pattern and target sets.

A second approach indexes patterns using a hash function, such as the phonetic encodings listed above. The hash value of each target string is then used as a key to retrieve the set of all patterns sharing the same hash value. For example, if Soundex were used as the hash function, the Soundex encoding of the target name "Mohammed," M530, would be identical the encoding of the pattern names "Muhamet" and "Mohamed." In general, any hash function that insures matches between some similar strings will fail to match other, equally similar strings. Multiple independent hash function are therefore required to insure high recall [Bra03].

In n-gram indexing, a third approach to retrieval, each pattern string is indexed by every n-element substring, *i.e.*, every sequence of n contiguous letters occurring in the pattern string. The candidates for each target string are retrieved using the n-grams in the target as keys. Recall for n-gram indexing has been reported to be quite high in some data sets [CRF03], but precision can be low. Precision can be increased by ignoring n-grams that discriminate poorly among (*e.g.*, match too high a proportion of) pattern strings.

3.3 Bounded Approximate String Matching

As mentioned above, in practice the minimum match threshold is frequently ill-defined, so matches are often simply displayed in rank order. In situations in which the maximum permissible number of differences k can be specified in advance, however, a number of algorithms are available to efficiently find all target strings that have at most k differences from a given pattern. A full discussion of these algorithms is beyond the scope of this paper, but see [Nav01] for an extensive survey. A major theoretical achievement of this work is a filtering algorithm with average cost $O(n(k + log_\sigma m)/m)$, where m is the pattern size, n the text size, k is the maximum number of errors, and σ the vocabulary size.

4. RESOURCES

SecondString is an open-source Java-based package for approximate string-matching techniques. SecondString was developed by researchers at Carnegie Mellon University from the Center for Automated Learning and Discovery, the Department of Statistics, and the Center for Computer and Communications Security.[3]

A second open-source resource is RIDDLE[4], ("Repository of Information on Duplicate Detection, Record Linkage, and Identity Uncertainty"), which contains data sets, a bibliography, and pointers to other resources.

5. CONCLUSION

Name matching arises in a wide variety of contexts, including many law enforcement and counter-terrorism tasks. Development of powerful approximate string-matching algorithms has improved the accuracy and efficiency of retrieval and character-based similarity assessment. However, there are few published algorithms or resources for culture-specific name matching. Moreover, there have been few systemic evaluations of the effectiveness of similarity assessment algorithms based on edit distance between phonetic representations. Instead, most phonetic approaches have been confined to exact match on phonetic hash functions, such as Soundex. An empirical evaluation of the conditions under which edit distance between phonetic representations leads to higher accuracy than orthographic edit distance would be a significant research contribution. Finally, research in name-matching has been impeded by the absence of archival test sets. A name-matching data repository, analogous to the UCI Machine Learning Repository[5] would be enormously beneficial to the field of name matching.

6. REFERENCES

M. Bilenko, W. W. Cohen, S. Fienberg, R. J. Mooney, and P. Ravikumar. Adaptive name-matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.

K. Branting. Name-matching algorithms for legal case-management systems. *Journal of Information, Law and Technology*, 1, 2002.

[3]Source code for SecondString is available at http://secondstring.sourceforge.net/.
[4]http://www.cs.utexas.edu/users/ml/riddle/.
[5]http://www.ics.uci.edu/ mlearn/MLSummary.html.

K. Branting. A comparative evaluation of name-matching algorithms. In *Ninth International Conference on Artificial Intelligence and Law (ICAIL 2003)*, pages 224–232. ACM Press, 2003.

W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web*, pages 73–78, Acapulco, Mexico, August 2003.

W. Dizard. Obsolete algorithm tangles terrorst/criminal watch lists. *Government Computer News*, 23(12), August 17 2004.

T. Gadd. Phonix: The algorithm. *Program*, 24(4), 1990.

D. Gusfield. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1999.

Victoria J. Hodge and Jim Austin. An evaluation of phonetic spell checkers.

M. A. Jaro. Probabilistic linkage of large public health data files. *Statistics in Medicine*, 14(5–7):491–498, 1995.

P. Jarvis, T. Lunt, and K. Myers. Identifying terrorist activity with aI plan recognition technology. In *Proceedings of the Sixteenth National Conference on Innovative Applications of Artificial Intelligence (IAAI 2004)*, pages 858–863, San Jose, California, July 25–29 2004. AAAI Press.

D. E. Knuth. *Fundamental Algorithms*, volume III of *The Art of Computer Programming*. Addison-Wesley, Reading, Massachusetts, 1975.

G. Kondrak. A new algorithm for the alignment of phonetic sequences. In *Proceedings of the First meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2000)*, pages 288–295. ACL, May 2000.

R. Lutz and S. Greene. Measuring phonlogical similarity: The case of personal names. White paper, Language Analysis Systems, Inc., 2003.

Alvaro E. Monge and Charles P. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proceedings of the SIGMOD 1997 Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 23–29, Tuscon, AZ, May 1997.

S. Milstein. Taming the task of checking for terrorists' names. The New York Times, C4, Monday, December 30 2002.

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

H.B. Newcombe, J.M. Kennedy, S.J. Axford, and A.P. James. Automatic linkage of vital records. *Science*, 130:954–959, 1959.

J. L. Peterson. Computer programs for detecting and correcting spelling errors. *Communications of the ACM*, 23(12):676–687, 1980.

L. Philips. Hanging on the metaphone. *Computer Language Magazine*, 7(12), December 1990.

L. Philips. The double metaphone search algorithm. *C/C++ Users Journal*, 18(1), June 1 2000.

J. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology*. PWS Publishing Co., 1997.

R. Taft. Name search techniques: New york state identification and intelligence system. Technical Report 1, State of New York, 1970.

W. E. Winkler. Advanced methods for record linkage. Technical report, Statistical Research Division, U.S. Census Bureau, Washington, DC, 1994.

W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Census Bureau, Washington, DC, 1999.

J. Zobel and P. Dart. Phonetic string matching: lessons from information retrieval. *SIGIR Forum*, 166–172, 1996.