Using Soundex Codes for Indexing Names in ASR documents

Hema Raghavan
hema@cs.umass.edu

James Allan
allan@cs.umass.edu

Abstract

In this paper we highlight the problems that arise due to variations of spellings of names that occur in text, as a result of which links between two pieces of text where the same name is spelt differently may be missed. The problem is particularly pronounced in the case of ASR text. We propose the use of approximate string matching techniques to normalize names in order to overcome the problem. We show how we could achieve an improvement if we could tag names with reasonable accuracy in ASR.

1 Introduction

Proper names are often key to our understanding of the information conveyed by a document. This is particularly the case when the domain is news. For example, a document with several mentions of *George W. Bush, Dick Cheney, Baghdad* and *Saddam Hussein*, gives us a good sense of what the contents of the document may be. In comparison, other regular English words like *death, scud* and *missiles*, may be good indicators of more general topics like *war*, but may not give us any indication of the exact event being discussed. Linking stories that discuss the same event, like the Attack on Iraq is very useful for a news filtering systems. When topics are primarily determined by specific events, it is easy to see why names of entities- people places and organizations, play such a critical role in discriminating between events that discuss a topic.

However, when one considers a real life scenario where news is from different media (print and broadcast) and in many different languages, proper names pose many different problems. The problem with proper names is that they often have different spelling variations. For example, the names *Arafat, Araafat,* and *Arafaat* may all refer to the same entity. Human beings can also vary in their spellings of a named entity. Besides that, the output of ASR and Machine Translation systems can also result in different spelling variations of a name. Such slight spelling variations may be acceptable and discernible by humans, but for a machine they are harder to match. A user who issues a query with the term *Arafat* in it may never find a document that discusses *Araafat*, using current TF-IDF matching techniques, even though the document may be pertinent to his or her query. Although this loss may not be critical to some applications, one cannot assume that the problem does not exist. The problem has been addressed by the data-base community in the past by the use of approximate string matching techniques, but in pure-text, we have the added problem of detecting names.

In this paper, we demonstrate with examples how sometimes we may not be able to draw connections between two pieces of text without the use of approximate string matching techniques. We indicate the problems we encounter while detecting names, and propose ways to address those issues. In the discussion of previous work in the next section we describe some tasks that use ASR output, and which may have been benefited by the use of approximate string matching techniques. We describe some preliminary experiments and their results. We then discuss the bottlenecks, in the proposed methodology, and how they may be overcome.

2 Past Work

2.1 Stemming

Stemming (Porter, 1980; Krovetz, 1993) is a method in which the corpus is processed so that semantically and morphologically related words are reduced to a common stem. Thus, *race*, *racing*, and *racer* are all reduced to a single root – *race*. Stemming has been found to be effective for Information Retrieval, TDT and other related tasks. Current stemming algorithms work only for regular English words and not names. In this paper we look at addressing the problem of grouping together and nor-

malizing proper names in the same way that stemming groups together regular English words.

2.2 Approximate String Matching

There has been some past work (French et al., 1997; Zobel and Dart, 1996) that has addressed the problem that proper names can have different spellings. Each of those works, however, only addresses the question of how effectively one can match a name to its spelling variants. They measure their performance in terms of the precision and recall with which they are able to retrieve other names which are variants of a given query name. Essentially, the primary motivation of those works was in finding good approximate string matching techniques. Those techniques are directly applicable only in applications that retrieve tuples from a database record.

However, there is no work that evaluates the effectiveness of approximate string matching techniques for names in an information retrieval or related task. We know of no work that attempts to detect names automatically, and then index names that should go together, in the same way that words of the same stem class are indexed by one common term.

2.3 The TREC SDR and the TDT Link Detection tasks

A single news-source may spell all mentions of a given name identically. However, this consistency is lost when there are multiple sources of news, where sources span languages and modes (broadcast and print). The TDT3 corpus (ldc, 2003) is representative of such real-life data. The corpus consists of English, Arabic and Mandarin print and broadcast news. ASR output is used in the case of the broadcast sources and in the case of non-English stories machine translated output is used for comparing stories. For both ASR systems and Machine Translation systems, proper names are often out-of-vocabulary (OOV). A typical speech recognizer has a lexicon of about 60K, and for a lexicon of this size about 10% of the person names are OOV. The OOV problem is usually solved by the use of transliteration and other such techniques. A breakdown of the OOV rates for names for different lexicon sizes is given in (Miller et al., 2000).

We believe the problem of spelling errors is of importance when one wants to index and retrieve ASR documents. For example, *Monica Lewinsky* is commonly referred to in the TDT3 corpus. The corpus has closed- caption transcripts for TV broadcasts. Closed caption suffers from typing errors. The name *Lewinsky* is also often misspelt as *Lewinskey* in the closed caption text. In the ASR text some of the variants that appear are *Lewenskey*, *Linski*, *Lansky* and *Lewinsky*. This example is typical, with the errors in the closed caption text highlighting how humans themselves can vary in their spelling of a name and the errors in ASR demonstrating how a single ASR system can output different spellings for the same name. The ASR errors are largely because ASR systems rely on phonemes for OOV words, and each of the different variations in the spellings of the same name is probably a result of different pronounciations and other such factors. The result of an ASR system then, is several different spelling variations of each name. It is easy to see why it would help considerably to group names that refer to the same entity together, and index them as one entity. We can exploit the fact that these different spelling variations of a given name exhibit strong similarity using approximate string matching techniques. We propose that in certain domains, where the issue that proper names exist with many different variations is dominant, the use of approximate string matching techniques to determine which names refer to the same entity will help improve the accuracy with which we can detect links between stories. Figure 1 shows a snippet of closed caption text and its ASR counterpart. The names Lewinskey and Tripp are misspelt in the ASR text. The two documents however have high similarity, because of the other words that the ASR system gets right. Allan (Allan, 2002) showed how ASR errors can cause misses in TDT tasks, and can sometimes be beneficial, resulting in a minimal average impact on performance in TDT. In the case of Spoken Document Retrieval (Garofolo et al., 2000) also it was found that a few ASR errors per document did not result in a big difference to performance as long as we get a reasonable percentage of the words right. Of course, factors such as the length of the two pieces of text being compared make a difference. Barnett et al (Barnett et al., 1997), showed how short queries were affected considerably by Word Error rate. ASR errors may not cause a significant drop in performance for any of the Topic Detection and Tracking tasks. But, consider a system where retrieving all documents mentioning *Lewinskey* and *Tripp* is critical, and it is not unrealistic to assume there exist systems with such needs, the ASR document in the above mentioned example would be left out. We therefore, believe that the problem we are addressing in this paper is an important one. The preliminary experiments in this paper, which are on the TDT corpus, only highlight how our approach can help.

3 Story Link Detection

3.1 Task Definition

The Story Link Detection Task is key to all the other tasks in TDT. The system is handed a set of story pairs, and for each pair it is asked to judge whether both the stories discuss the same topic or different topics. In addition to a YES/NO decision the system is also expected to output a confidence score, where a low confidence score implies

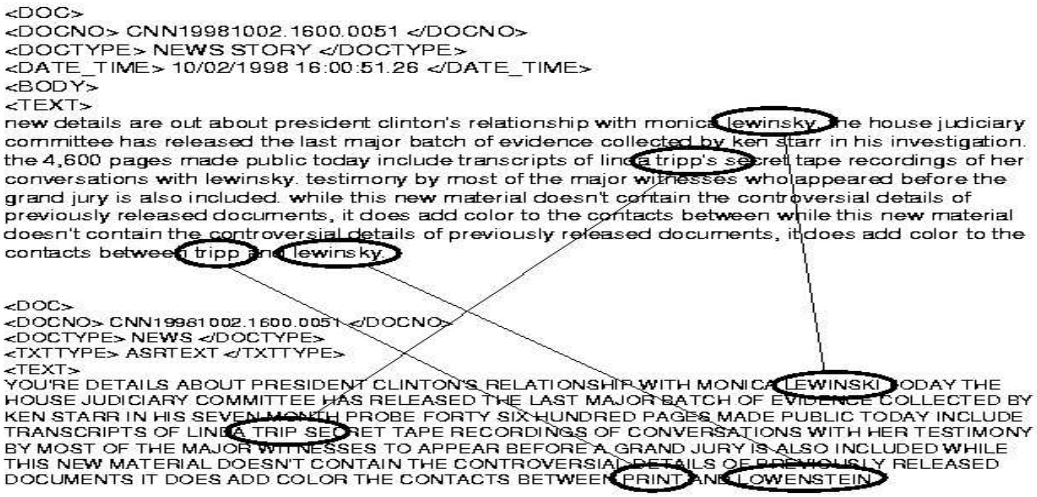

Figure 1: Errors in ASR text The errors in the spellings of names are highlighted in the ASR and the corresponding words in the closed caption snippet are also marked.

that the system is more in favor of the NO decision.

3.2 Our Approach

Simply stated our approach to the SLD task, is to use approximate string matching techniques to compare entities between two pieces of text. The two pieces of text may be a query and a document, or two documents, depending on the task. We first need to identify entities in the two documents. There exist several techniques to automatically identify names. For properly punctuated text, heuristics like capitalization work sufficiently well. However, for ASR text we often do not have sentence boundaries or even punctuation. Hence we rely on a Hidden Markov Model based named entity recognizer (Bikel et al., 1999) for our task.

A simple strategy that incorporates an approximate string matching technique is to first preprocess the corpus, and then normalize all mentions of a named entity to a given canonical form, where the canonical form is independent of mentions of other entities in the two documents being compared. Soundex, Phonix, and other such codes offer us a means of normalizing a word to its phonetic form. The Soundex code is a combination of the first letter of the word and a three digit code which is representative of its phonetic sound. Hence, similar sounding names like ”Lewinskey” and ”Lewinsky” are both reduced to the same soundex code ”l520”. We can pre-process the corpus so that all the named entities are replaced by their Soundex codes. We then compute the similarity between documents in the new corpus as opposed to the old one, using conventional similarity metrics like Cosine or TF-IDF.

4 Experimental Set up

4.1 Data

The corpus (ldc, 2003) has 67111 documents from multiple sources of news in multiple languages (English Chinese and Arabic) and media (broadcast news and newswire). The English sources are Associated Press and New York Times, PRI, Voice of America etc. For the broadcast news sources we have ASR output and for TV we have both ASR output as well as closed caption data. Additionally we have the following Mandarin news-wire, web and broadcast sources - Xinhua news, Zaobao, and Voice of America (Mandarin). For all the Mandarin documents we have the original documents in the native language as well the English output of Systran- a machine translation system. The data has been collected by LDC by sampling from the above mentioned sources in the period from October to December 1998.

The LDC has annotated 60 topics in the TDT3 corpus. A topic is determined by an event. For example topic 30001 is the *Cambodian Government Coalition*. Each

topic has key entities associated with it and a description of the topic. A subset of the documents are annotated as being on-topic or not according to a well formed strategy as defined by the LDC.

4.2 Story Link Detection

To compute the similarity of two documents, that is, the YES/NO decision threshold, we used the the traditional cosine similarity metric. To give some leverage to documents that were very similar even before named entity normalization, we average the similarity scores between documents before and after the named entities have been normalized by their Soundex codes as follows:

$$Sim(D_1, D_2) = \frac{1}{2}(Cos(D_1, D_2) + Cos(D_1', D_2')) \quad (1)$$

Where D_1 and D_2 are the original documents and D_1' and D_2' are the documents after the names have been normalized.

4.3 Evaluation

An ROC curve is plotted by making a parameter sweep of the YES/NO decision thresholds, and plotting the Misses and False Alarms at each point. At each point the cost is computed using the following empirically determined formula (Fiscus et al., 1998).

$$C_{det} = 0.02P(miss) + 0.098P(fa) \quad (2)$$

This cost function is standard across all tasks. The point of minimum cost serves as the comparison between various systems.

5 Results

We tested our idea on the TDT3 corpus for the Story Link Detection Task, using the Cosine similarity metric, and found that performance actually degraded. On investigation we found that the named entity recognizer performs poorly on Machine Translated and ASR source data. Our named entity recognizer relies considerably on sentence structure, to make its predictions. Machine translated output often lacks grammatical structure, and ASR output does not have punctuation, which results in a lot of named entity tagging errors.

We therefore decided to test our idea for newswire text. We created our own test set of 4752 pairs of stories from newswire sources. This test set was created by randomly picking on and off-topic stories for each topic using the same policy as employed by the LDC (Fiscus, 2003). On these pairs, we obtained about 10% improvement (Figure 2), suggesting that there is merit in Soundex normalization of names. However, the problem of poor named entity recognition is a bottle-neck for ASR. We discuss

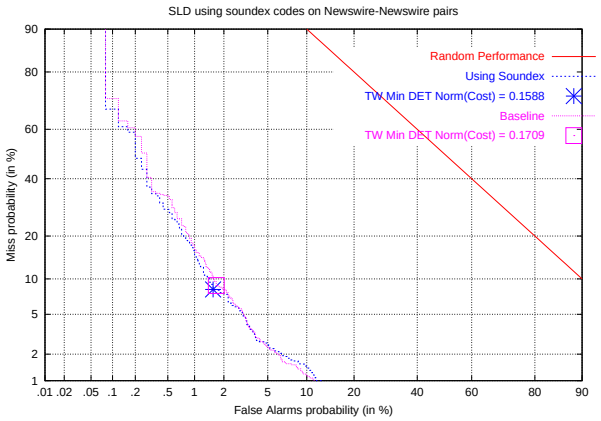

Figure 2: Story Link Detection performance

alternative strategies of how to deal with this, and other ways of using approximate string matching in the next section.

6 Alternative strategies

6.1 To not use an entity recognizer

We were not able to benefit from our approach on the ASR documents because of the poor performance of the named entity recognizer on those types of document. An example of a randomly picked named entity tagged ASR document is given below. The tagging errors are underlined.

< DOC >
< DOCNO > $CNN19981001.0130.0000$ < /DOCNO >
< TEXT >
< ENAMEX TYPE="ORGANIZATION" > BUDGET SURPLUS < /ENAMEX> AND FIGHTING OVER WHETHER IT'S GOING DOOR POCKETS WILL TELL YOU THE < ENAMEX TYPE="ORGANIZATION" > VEHICLES CLIMBED DATES THEREAFTER < /ENAMEX > AND IF YOU'RE REQUIRED TO PAY CHILD SUPPORT INFORMATION THAT YOUR JOB AND COME AND ADDRESS NOW PART HAVE < ENAMEX TYPE="ORGANIZATION" > A NATIONAL REGISTRY THE HEADLINE < /ENAMEX > NEWS I'M < ENAMEX TYPE="PERSON"> KIMBERLY KENNEDY </ENAMEX> THOSE STORIES IN A MOMENT BUT FIRST </TEXT> </DOC >

We need a better performing recognizer, but that may be hard. Instead we might be able to use other information from the speech recognizer to overcome this problem. We did not have confidence scores for the words in the ASR output. If we had had that information, or if we

were able to obtain information about which words were OOV, we could possibly index all words with low confidence scores or all OOV words by their Soundex codes. Or else, one could normalize all words in the ASR output, that are not part of the regular English vocabulary by their Soundex codes.

6.2 Other ways of grouping entities

Another direction of research to pursue is the way in which approximate string matching is used to compare documents. The way we used approximate string matching in this paper was fairly simple. However, it loses out on some names that ought to go together particularly when two names differ in their first alphabet - for example *Katherine* and *Catherine*. The Soundex codes are *k365* and *c365* respectively. This is by virtue of the nature of the Soundex code of word.

There are other ways to compute the similarity between two documents like the Levenshtein distance or edit distance which is a measure of the number of string edit operations required to convert one string to the other. The words *Katherine* and *Catherine* have an edit distance of 1. Given two documents D_1 and D_2, we can compute the distance between them by computing the distance between all pairs of names that occur in the two documents, and using the distances to group entities and finally to find the similarity of the two documents. However this means that each entity in D_1 has to be compared to all entities in D_1 and D_2. Besides, this method brings with it the question of how to use the distances between the names so as to group together similar names. This method is probably a good direction for future research, because the Levenshtein distance could possibly be a better string matching technique. Another plausible strategy would be to use the edit-distance of the Soundex codes of the names, when comparing documents. *Katherine* and *Catherine* would have a distance of 1 in this case too.

Using cross document coreference resolution techniques to find equivalence classes of entities would be yet another alternative approach. In Cross document coreference, two mentions of the same name, may or may not be included in the same group depending on whether or not the context of the two mentions is the same or is different.

7 Conclusions and Future Directions

In this paper we highlighted an important problem that occurs with names in ASR text. We showed how a name may be spelt differently by humans. In ASR the same name had many more different spellings.

We proposed a simple indexing strategy for names, wherein a name was indexed by its Soundex code. We found that our strategy did not work for ASR, but the problem was not with the approach, but because we could not do a good job of identifying names in ASR text.If we could detect names with reasonable accuracy in ASR text we should be able to achieve reasonable improvement. We did not have a named entity recognizer that performed well on ASR text. We therefore verified our idea on news-wire text, which is grammatical, well punctuated text. In the news-wire domain, in spite of there being reasonable consistency in spellings of names, we get about 10% improvement in minimum cost, and a consistent improvement at all points in the ROC curve. Hence, a simple technique like Soundex served as a useful normalization technique for names. We proposed alternative mechanisms that could be applied to ASR text, wherein all OOV words could be normalized by their Soundex codes. We also outlined further directions for research in the way that approximate string matching may be used.

We think the general results of past works that has considered the problems due to ASR errors to be insignificant cannot be assumed to transfer across to other problems. There will arise situations when this problem is material and research needs to be done in this direction.

8 Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval, in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

James Allan. 2002. Detecting and tracking topics in broadcast news,att speechdays 2002.

James Barnett, Steve Anderson, John Broglio, Mona Singh, R. Hudson, and S. W. Kuo. 1997. Experiments in spoken queries for document retrieval. In *Proc. Eurospeech '97*, pages 1323–1326, Rhodes, Greece.

Daniel M. Bikel, Richard L. Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231.

J. Fiscus, G. Doddington, J. Garofolo, and A. Martin. 1998. Nist's 1998 topic detection and tracking evaluation.

John Fiscus. 2003. Personal communication.

J. C. French, A. L. Powell, and E. Schulman. 1997. Applications of approximate word matching in information retrieval. In *Proceedings of the Sixth International Conference on Knowledge and Information Management*, pages 9–15, New York, NY. ACM Press.

J. Garofolo, G. Auzanne, and E. Voorhees. 2000. The trec spoken document retrieval track: A success story.

R. Krovetz. 1993. Viewing Morphology as an Inference Process,. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–203.

2003. http://www.ldc.upenn.edu/tdt/.

David Miller, Richard Schwartz, Ralph Weischedel, and Rebecca Stone. 2000. Named entity extraction from broadcast news.

M.F. Porter. 1980. An algorithm for suffix stripping. *Program*.

J. Zobel and P. W. Dart. 1996. Phonetic string matching: Lessons from information retrieval. In H.-P. Frei, D. Harman, P. Schäble, and R. Wilkinson, editors, *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, pages 166–172, Zurich, Switzerland. ACM Press.