A Comparison of String Distance Metrics for Name-Matching Tasks

William W. Cohen[*†]
[*]Center for Automated Learning and Discovery
Carnegie Mellon University
Pittsburgh PA 15213
wcohen@wcohen.com

Pradeep Ravikumar[*]
[†]Center for Computer and Communications Security
Carnegie Mellon University
Pittsburgh PA 15213
pradeepr@cs.cmu.edu

Stephen E. Fienberg[*†‡]
‡Department of Statistics
Carnegie Mellon University
Pittsburgh PA 15213
fienberg@stat.cmu.edu

Abstract

Using an open-source, Java toolkit of name-matching methods, we experimentally compare string distance metrics on the task of matching entity names. We investigate a number of different metrics proposed by different communities, including edit-distance metrics, fast heuristic string comparators , token-based distance metrics, and hybrid methods. Overall, the best-performing method is a hybrid scheme combining a TFIDF weighting scheme, which is widely used in information retrieval, with the Jaro-Winkler string-distance scheme, which was developed in the probabilistic record linkage community.

Introduction

The task of matching entity names has been explored by a number of communities, including statistics, databases, and artificial intelligence. Each community has formulated the problem differently, and different techniques have been proposed.

In statistics, a long line of research has been conducted in *probabilistic record linkage*, largely based on the seminal paper by Fellegi and Sunter (1969). This paper formulates entity matching as a classification problem, where the basic goal is to classify entity pairs as matching or non-matching. Fellegi and Sunter propose using largely unsupervised methods for this task, based on a feature-based representation of pairs which is manually designed and to some extent problem-specific. These proposals have been, by and large, adopted by subsequent researchers, although often with elaborations of the underlying statistical model (Jaro 1989; 1995; Winkler 1999; Larsen 1999; Belin & Rubin 1997). These methods have been used to match individuals and/or families between samples and censuses, e.g., in evaluation of the coverage of the U.S. decennial census; or between administrative records and survey data bases, e.g., in the creation of an anonymized research data base combining tax information from the Internal Revenue Service and data from the Current Population Survey.

In the database community, some work on record matching has been based on knowledge-intensive approaches (Hernandez & Stolfo 1995; Galhardas *et al*. 2000; Raman & Hellerstein 2001). However, the use of string-edit distances as a general-purpose record matching scheme was proposed by Monge and Elkan (Monge & Elkan 1997; 1996), and in previous work, we proposed use of the TFIDF distance metric for the same purpose (Cohen 2000). In the AI community, supervised learning has been used for learning the parameters of string-edit distance metrics (Ristad & Yianilos 1998; Bilenko & Mooney 2002) and combining the results of different distance functions (Tejada, Knoblock, & Minton 2001; Cohen & Richman 2002; Bilenko & Mooney 2002). More recently, probabilistic object identification methods have been adapted to matching tasks (Pasula *et al*. 2002). In these communities there has been more emphasis on developing "autonomous" matching techniques which require little or or no configuration for a new task, and less emphasis on developing "toolkits" of methods that can be applied to new tasks by experts.

Recently, we have begun implementing an open-source, Java toolkit of name-matching methods which includes a variety of different techniques. In this paper we use this toolkit to conduct a comparison of several string distances on the tasks of matching and clustering lists of entity names.

This experimental use of string distance metrics, while similar to previous experiments in the database and AI communities, is a substantial departure from their usual use in statistics. In statistics, databases tend to have more structure and specification, by design. Thus the statistical literature on probabilistic record linkage represents pairs of entities not by pairs of strings, but by vectors of "match features" such as names and categories for variables in survey databases. By developing appropriate match features, and appropriate statistical models of matching and non-matching pairs, this approach can achieve better matching performance (at least potentially).

The use of string distances considered here is most useful for matching problems with little prior knowledge, or ill-structured data. Better string distance metrics might also be useful in the generation of "match features" in more structured database situations.

Methods

Edit-distance like functions

Distance functions map a pair of strings s and t to a real number r, where a smaller value of r indicates greater similarity between s and t. *Similarity functions* are analogous, except that larger values indicate greater similarity; at some risk of confusion to the reader, we will use this terms interchangably, depending on which interpretation is most natural.

One important class of distance functions are *edit distances*, in which distance is the cost of best sequence of *edit operations* that convert s to t. Typical edit operations are character insertion, deletion, and substitution, and each operation much be assigned a cost.

We will consider two edit-distance functions. The simple *Levenstein distance* assigns a unit cost to all edit operations. As an example of a more complex well-tuned distance function, we also consider the *Monger-Elkan* distance function (Monge & Elkan 1996), which is an affine[1] variant of the Smith-Waterman distance function (Durban *et al*. 1998) with particular cost parameters, and scaled to the interval [0,1].

A broadly similar metric, which is *not* based on an edit-distance model, is the *Jaro* metric (Jaro 1995; 1989; Winkler 1999). In the record-linkage literature, good results have been obtained using variants of this method, which is based on the number and order of the common characters between two strings. Given strings $s = a_1 \ldots a_K$ and $t = b_1 \ldots b_L$, define a character a_i in s to be *common with* t there is a $b_j = a_i$ in t such that $i - H \leq j \leq i + H$, where $H = \frac{\min(|s|,|t|)}{2}$. Let $s' = a'_1 \ldots a'_{K'}$ be the characters in s which are common with t (in the same order they appear in s) and let $t' = b'_1 \ldots b'_{L'}$ be analogous; now define a *transposition for* s', t' to be a position i such that $a'_i \neq b'_i$. Let $T_{s',t'}$ be half the number of transpositions for s' and t'. The Jaro similarity metric for s and t is

$$\mathit{Jaro}(s,t) = \frac{1}{3} \cdot \left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s',t'}}{|s'|} \right)$$

A variant of this due to Winkler (1999) also uses the length P of the longest common prefix of s and t. Letting $P' = \max(P, 4)$ we define

$$\mathit{Jaro\text{-}Winkler}(s,t) = \mathit{Jaro}(s,t) + \frac{P'}{10} \cdot (1 - \mathit{Jaro}(s,t))$$

The Jaro and Jaro-Winkler metrics seem to be intended primarily for short strings (e.g., personal first or last names.)

Token-based distance functions

Two strings s and t can also be considered as multisets (or *bags*) of words (or tokens). We also considered several token-based distance metrics. The *Jaccard similarity* between the word sets S and T is simply $\frac{|S \cap T|}{|S \cup T|}$. *TFIDF* or *cosine similarity*, which is widely used in the information retrieval community can be defined as

$$\mathit{TFIDF}(S,T) = \sum_{w \in S \cap T} V(w,S) \cdot V(w,T)$$

where $\mathrm{TF}_{w,S}$ is the frequency of word w in S, N is the size of the "corpus", IDF_w is the inverse of the fraction of names in the corpus that contain w,

$$V'(w,S) = \log(\mathrm{TF}_{w,S} + 1) \cdot \log(\mathrm{IDF}_w)$$

and $V(w,S) = V'(w,S)/\sqrt{\sum_{w'} V'(w,S)^2}$. Our implementation measures all document frequency statistics from the complete corpus of names to be matched.

Following Dagan *et al* (1999), a token set S can be viewed as samples from an unknown distribution P_S of tokens, and a distance between S and T can be computed based on these distributions. Inspired by Dagan *et al* we considered the *Jensen-Shannon* distance between P_S and P_T. Letting $KL(P||Q)$ be the Kullback-Lieber divergence and letting $Q(w) = \frac{1}{2}(P_S(w) + P_T(w))$, this is simply

$$\mathit{Jensen\text{-}Shannon}(S,T) = \frac{1}{2}\left(KL(P_S||Q) + KL(P_T||Q)\right)$$

Distributions P_S were estimated using maximum likelihood, a Dirichlet prior, and a Jelenik-Mercer mixture model (Lafferty & Zhai 2001). The latter two cases require parameters; for Dirichlet, we used $\alpha = 1$, and for Jelenik-Mercer, we used the mixture coefficient $\lambda = 0.5$.

From the record-linkage literature, a method proposed by Fellegi and Sunter (1969) can be easily extended to a token distance. As notation, let A and B be two sets of records to match, let $C = A \cap B$, let $D = A \cup B$, and for $X = A, B, C, D$ let $P_X(w)$ be the empirical probability of a name containing the word w in the set X. Also let e_X be the empirical probability of an error in a name in set X; $e_{X,0}$ be the probability of a missing name in set X; e_T be the probability of two correct but differing names in A and B; and let $e = e_A + e_B + e_T + e_{A,0} + e_{B,0}$.

Fellegi and Sunter propose ranking pairs s, t by the odds ratio $\log(\frac{\Pr(M|s,t)}{\Pr(U|s,t)})$ where M is the class of matched pairs and U is the class of unmatched pairs. Letting $\mathrm{AGREE}(s,t,w)$ denote the event "s and t both agree in containing word w", Fellegi and Sunter note that under certain assumptions

$$\begin{aligned} \Pr(M|\mathrm{AGREE}(s,t,w)) &\approx P_C(w)(1-e) \\ \Pr(U|\mathrm{AGREE}(s,t,w)) &\approx P_A(w) \cdot P_B(w)(1-e) \end{aligned}$$

If we make two addition assumptions suggested by Fellegi and Sunter, and assume that (a) matches on a word w are independent, and (b) that $P_A(w) \approx P_B(w) \approx P_C(w) \approx P_D(w)$, we find that the incremental score for the odds ratio associated with $\mathrm{AGREE}(s,t,w)$ is simply $-\log P_D(w)$. In information retrieval terms, this is a simple un-normalized IDF weight.

Unfortunately, updating the log-odds score of a pair on discovering a disagreeing token w is not as simple. Estimates are provided by Fellegi and Sunter for

[1] *Affine* edit-distance functions assign a relatively lower cost to a sequence of insertions or deletions.

$P(M|\neg\text{AGREE}(s,t,w))$ and $P(U|\neg\text{AGREE}(s,t,w))$, but the error parameters $e_A, e_B, \ldots$ do not cancel out– instead one is left with a constant penalty term, independent of w. Departing slightly from this (and following the intuition that disagreement on frequent terms is less important) we introduce a *variable* penalty term of $k \log P_D(w)$, where k is a parameter to be set by the user.

In the experiments we used $k = 0.5$, and call this method *SFS distance* (for Simplified Fellegi-Sunter).

Hybrid distance functions

Monge and Elkan propose the following *recursive matching scheme* for comparing two long strings s and t. First, s and t are broken into substrings $s = a_1 \ldots a_K$ and $t = b \ldots b_L$. Then, similarity is defined as

$$sim(s,t) = \frac{1}{K} \sum_{i=1}^{K} \max_{j=1}^{L} sim'(A_i, B_j)$$

where sim' is some secondary distance function. We considered an implementation of this scheme in which the substrings are tokens; following Monge and Elkan, we call this a *level two distance function*. We experimented with level two similarity functions which used Monge-Elkan, Jaro, and Jaro-Winkler as their base function sim'.

We also considered a "soft" version of TFIDF, in which similar tokens are considered as well as tokens in $S \cap T$. Again let sim' be a secondary similarity function. Let $CLOSE(\theta, S, T)$ be the set of words $w \in S$ such that there is some $v \in T$ such that $dist'(w,v) > \theta$, and for $w \in CLOSE(\theta, S, T)$, let $D(w,T) = \max_{v \in T} dist(w,v)$. We define

$$SoftTFIDF(S,T) = \sum_{w \in CLOSE(\theta,S,T)} V(w,S) \cdot V(w,T) \cdot D(w,T)$$

In the experiments, we used Jaro-Winkler as a secondary distance and $\theta = 0.9$.

"Blocking" or pruning methods

In matching or clustering large lists of names, it is not computationally practical to measure distances between all pairs of strings. In statistical record linkage, it is common to group records by some variable which is known *a priori* to be usually the same for matching pairs. In census work, this grouping variable often names a small geographic region, and perhaps for this reason the technique is usually called "blocking".

Since this paper focuses on the behavior of string matching tools when little prior knowledge is available, we will use here knowledge-free approaches to reducing the set of string pairs to compare. In a *matching task*, there are two sets A and B. We consider as candidates all pairs of strings $(s,t) \in A \times B$ such that s and t share some substring v which appears in at most a fraction f of all names. In a *clustering task*, there is one set C, and we consider all candidates $(s,t) \in C \times C$ such that $s \neq t$, and again s and t share some not-too-frequent substring v. For purely token-based methods, the substring v must be a token, and otherwise, it must be a character 4-gram. Using inverted indices this set of pairs can be enumerated quickly.

For the moderate-size test sets considered here, we used $f = 1$. On the matching datasets above, the token blocker finds between 93.3% and 100.0% of the correct pairs, with an average of 98.9%. The 4-gram blocker also finds between 93.3% and 100.0% of the correct pairs, with an average of 99.0%.

Name	Src	M/C	#Strings	#Tokens
animal	1	M	5709	30,006
bird1	1	M	377	1,977
bird2	1	M	982	4,905
bird3	1	M	38	188
bird4	1	M	719	4,618
business	1	M	2139	10,526
game	1	M	911	5,060
park	1	M	654	3,425
fodorZagrat	2	M	863	10,846
ucdFolks	3	M	90	454
census	4	M	841	5,765
UVA	3	C	116	6,845
coraATDV	5	C	1916	47,512

Table 1: Datasets used in experiments. Column 2 indicates the source of the data, and column 3 indicates if it is a matching (M) or clustering (C) problem. Original sources are 1. (Cohen 2000) 2. (Tejada, Knoblock, & Minton 2001) 3. (Monge & Elkan 1996) 4. William Winkler (personal communication) 5. (McCallum, Nigam, & Ungar 2000) communication)

Experiments

Data and methodology

The data used to evaluate these methods is shown in Table 1. Most been described elsewhere in the literature. The "coraATDV" dataset includes the fields author, title, date, and venue in a single string. The "census" dataset is a synthetic, census-like dataset, from which only textual fields were used (last name, first name, middle initial, house number, and street).

To evaluate a method on a dataset, we ranked by distance all candidate pairs from the appropriate grouping algorithm. We computed the non-interpolated average precision of this ranking, the maximum F1 score of the ranking, and also interpolated precision at the eleven recall levels 0.0, 0.1, ..., 0.9, 1.0. The *non-interpolated average precision* of a ranking containing N pairs for a task with m correct matches is $\frac{1}{m} \sum_{r=1}^{N} \frac{c(i)\delta(i)}{i}$, where $c(i)$ is the number of correct pairs ranked before position i, and $\delta(i) = 1$ if the pair at rank i is correct and 0 otherwise. *Interpolated precision* at recall r is the $\max_i \frac{c(i)}{i}$, where the max is taken over all ranks i such that $\frac{c(i)}{m} \geq r$. *Maximum F1* is $\max_{i>0} F1(i)$, where $F1(i)$ is the harmonic mean of recall at rank i (i.e., $c(i)/m$) and precision at rank i (i.e., $c(i)/i$).

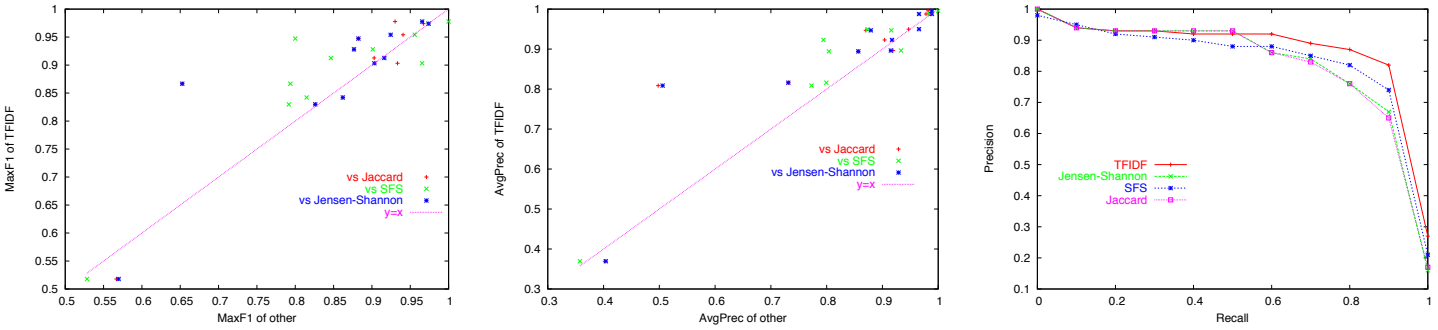

Figure 1: Relative performance of token-based measures. Left, max F1 of methods on matching problems, with points above the line $y = x$ indicating better performance of TFIDF. Middle, same for non-interpolated average precision. Right, precision-recall curves averaged over all matching problems. Smoothed versions of Jensen-Shannon (not shown) are comparable in performance to the unsmoother version.

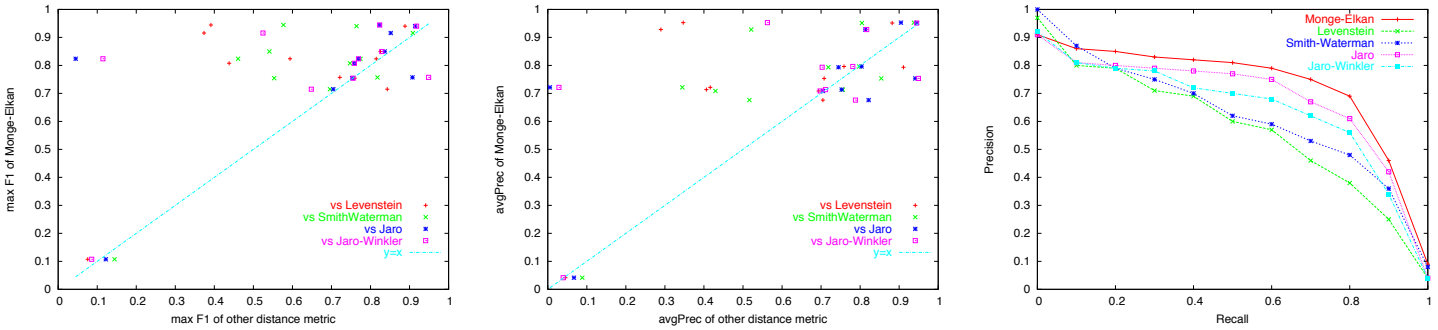

Figure 2: Relative performance of edit-distance measures. Left and middle, points above (below) the line $y = x$ indicating better (worse) performance for Monge-Elkan, the system performing best on average.

Results for matching

We will first consider the matching datasets. As shown in Figure 1, TFIDF seems generally the best among the token-based distance metrics. It does slightly better on average than the others, and is seldom much worse than any other method with respect to interpolated average precision or maximum F1.

As shown in Figure 2, the situation is less clear for the edit-distance based methods. The Monge-Elkan method does best on average, but the Jaro and Jaro-Winkler methods are close in average performance, and do noticably better than Monge-Elkan on several problems. The Jaro variants are also substantially more efficient (at least in our implementation), taking about 1/10 the time of the Monge-Elkan method. (The token-based methods are faster still, averaging less than 1/10 the time of the Jaro variants.)

As shown in Figure 3, SoftTFIDF is generally the best among the hybrid methods we considered. In general, the time for the hybrid methods is comparable to using the underlying string edit distance. (For instance, the average matching time for SoftTFIDF is about 13 seconds on these problems, versus about 20 seconds for the Jaro-Winkler method, and 1.2 seconds for pure token-based TFIDF.)

Finally, Figure 4 compares the three best performing edit-distance like methods, the two best token-based methods, and the two best hybrid methods, using a similar methodology. Generally speaking, SoftTFIDF is the best overall distance measure for these datasets.

Results for clustering

It should be noted that the test suite of matching problems is dominated by problems from one source—eight of the eleven test cases are associated with the WHIRL project—and a different distribution of problems might well lead to quite different results. For instance, while the token-based methods perform well on average, they perform poorly on the census-like dataset, which contains many misspellings.

As an additional experiment, we evaluated the four best-performing distance metrics (SoftTFIDF, TFIDF, SFS, and Level 2 Jaro-Winkler) on the two clustering problems, which are taken from sources other than the WHIRL project. Table 2 shows MaxF1 and non-interpolated average precision for each method on each problem. SoftTFIDF again slightly outperforms the other methods on both of these tasks.

	UVA		CoraATDV	
Method	MaxF1	AvgPrec	MaxF1	AvgPrec
SoftTFIDF	0.89	0.91	0.85	0.914
TFIDF	0.79	0.84	0.84	0.907
SFS	0.71	0.75	0.82	0.864
Level2 J-W	0.73	0.69	0.76	0.804

Table 2: Results for selected distance metrics on two clustering problems.

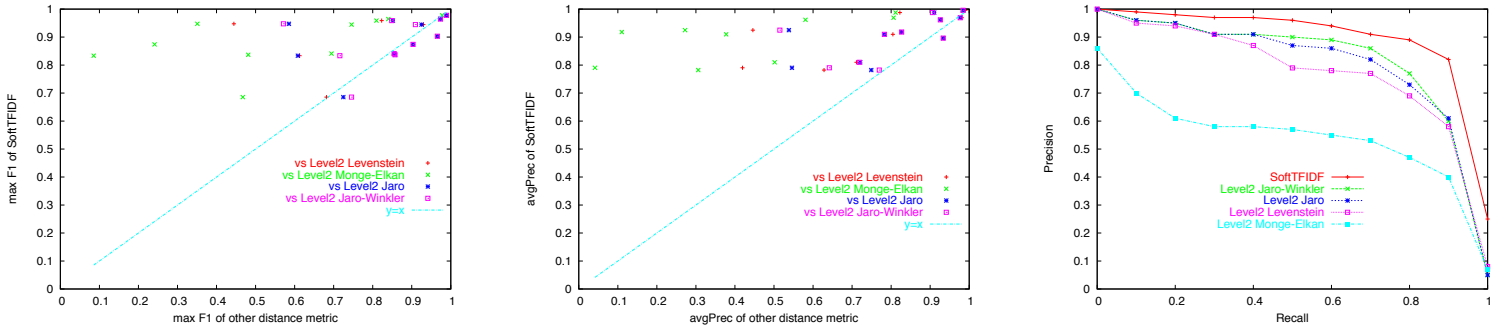

Figure 3: Relative performance of hybrid distance measures on matching problems, relative to the SoftTFIDF metric, which performs best on average.

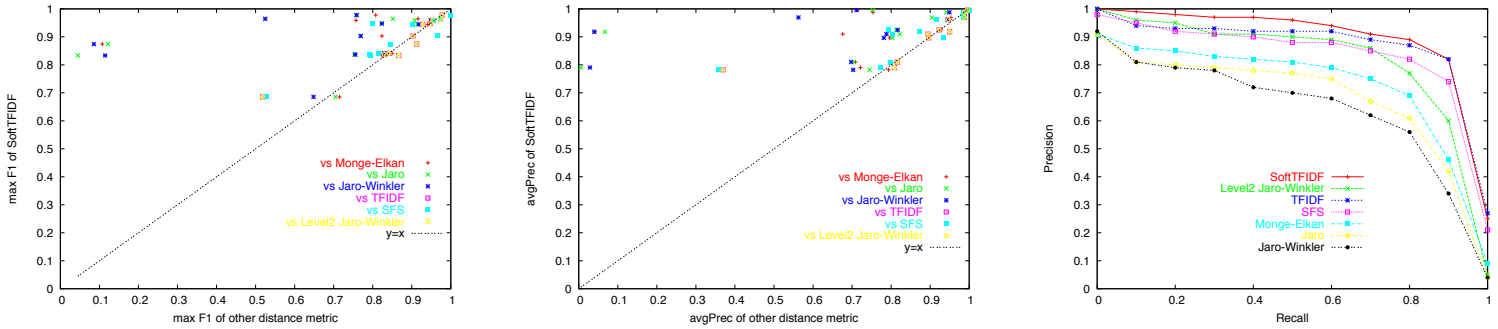

Figure 4: Relative performance of best distance measures of each type on matching problems, relative to the SoftTFIDF metric.

Learning to combine distance metrics

Another type of hybrid distance function can be obtained by combining other distance metrics. Following previous researchers (Tejada, Knoblock, & Minton 2001; Cohen & Richman 2002; Bilenko & Mooney 2002) we used a learning scheme to combine several of the distance functions above. Specifically, we represented pairs as feature vectors, using as features the numeric scores of Monge-Elkan, Jaro-Winkler, TFIDF, SFS, and SoftTFIDF. We then trained a binary SVM classifier (using SVM Light (Joachims 2002)) using these features, and used its confidence in the "match" class as a score.

Figure 5 shows the results of a three-fold cross-validation on nine of the matching problems. The learned combination generally slightly outperforms the individual metrics, including SoftTFIDF, particularly at extreme recall levels. Note, however, that the learned metric uses much more information: in particular, in most cases it has been trained on several thousand labeled candidate pairs, while the other metrics considered here require no training data.

Concluding remarks

Recently, we have begun implementing an open-source, Java toolkit of name-matching methods. This toolkit includes a variety of different techniques, as well as the infrastructure to combine techniques readily, and evaluate them systematically on test data. Using this toolkit, we conducted a comparison of several string distances on the tasks of matching and clustering lists of entity names. Many of these were techniques previously proposed in the literature, and some are novel hybrids of previous methods.

We compared these accuracy of these methods for use in an automatic matching scheme, in which pairs of names are proposed by a simple grouping method, and then ranked according to distance. Used in this way, we saw that the TFIDF ranking performed best among several token-based distance metrics, and that a tuned affine-gap edit-distance metric proposed by Monge and Elkan performed best among several string edit-distance metrics. A surprisingly good distance metric is a fast heuristic scheme, proposed by Jaro (Jaro 1995; 1989) and later extended by Winkler (Winkler 1999). This works almost as well as the Monge-Elkan scheme, but is an order of magnitude faster.

One simple way of combining the TFIDF method and the Jaro-Winkler is to replace the exact token matches used in TFIDF with approximate token matches based on the Jaro-Winkler scheme. This combination performs slightly better than either Jaro-Winkler or TFIDF on average, and occasionally performs much better. It is also close in performance to a learned combination of several of the best metrics considered in this paper.

Acknowledgements

The preparation of this paper was supported in part by National Science Foundation Grant No. EIA-0131884 to the National Institute of Statistical Sciences and by a contract from the Army Research Office to the Center for Computer

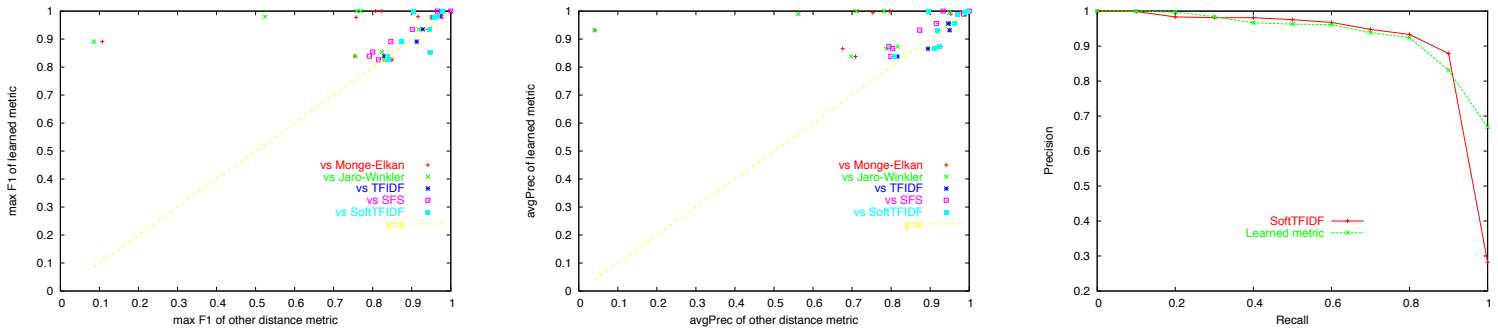

Figure 5: Relative performance of best distance measures of each type on matching problems, relative to a learned combination of the same measures.

and Communications Security with Carnegie Mellon University.

References

Belin, T. R., and Rubin, D. B. 1997. A method for calibrating false-match rates in record linkage. In *Record Linkage – 1997: Proceedings of an International Workshop and Exposition*, 81–94. U.S. Office of Management and Budget (Washington).

Bilenko, M., and Mooney, R. 2002. Learning to combine trained distance metrics for duplicate detection in databases. Technical Report Technical Report AI 02-296, Artificial Intelligence Lab, University of Texas at Austin. Available from http://www.cs.utexas.edu/users/ml/papers/marlin-tr-02.pdf.

Cohen, W. W., and Richman, J. 2002. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002)*.

Cohen, W. W. 2000. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems* 18(3):288–321.

Dagan, I.; Lee, L.; and Pereira, F. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning* 34(1-3).

Durban, R.; Eddy, S. R.; Krogh, A.; and Mitchison, G. 1998. *Biological sequence analysis - Probabilistic models of proteins and nucleic acids*. Cambridge: Cambridge University Press.

Fellegi, I. P., and Sunter, A. B. 1969. A theory for record linkage. *Journal of the American Statistical Society* 64:1183–1210.

Galhardas, H.; Florescu, D.; Shasha, D.; and Simon, E. 2000. An extensible framework for data cleaning. In *ICDE*, 312.

Hernandez, M., and Stolfo, S. 1995. The merge/purge problem for large databases. In *Proceedings of the 1995 ACM SIGMOD*.

Jaro, M. A. 1989. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association* 84:414–420.

Jaro, M. A. 1995. Probabilistic linkage of large public health data files (disc: P687-689). *Statistics in Medicine* 14:491–498.

Joachims, T. 2002. *Learning to Classify Text Using Support Vector Machines*. Kluwer.

Lafferty, J., and Zhai, C. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *2001 ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

Larsen, M. 1999. Multiple imputation analysis of records linked using mixture models. In *Statistical Society of Canada Proceedings of the Survey Methods Section*, 65–71. Statistical Society of Canada (McGill University, Montreal).

McCallum, A.; Nigam, K.; and Ungar, L. H. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Knowledge Discovery and Data Mining*, 169–178.

Monge, A., and Elkan, C. 1996. The field-matching problem: algorithm and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*.

Monge, A., and Elkan, C. 1997. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *The proceedings of the SIGMOD 1997 workshop on data mining and knowledge discovery*.

Pasula, H.; Marthi, B.; Milch, B.; Russell, S.; and Shpitser, I. 2002. Identity uncertainty and citation matching. In *Advances in Neural Processing Systems 15*. Vancouver, British Columbia: MIT Press.

Raman, V., and Hellerstein, J. 2001. Potter's wheel: An interactive data cleaning system. In *The VLDB Journal*, 381–390.

Ristad, E. S., and Yianilos, P. N. 1998. Learning string edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(5):522–532.

Tejada, S.; Knoblock, C. A.; and Minton, S. 2001. Learning object identification rules for information integration. *Information Systems* 26(8):607–633.

Winkler, W. E. 1999. The state of record linkage and current research problems. Statistics of Income Division, Internal Revenue Service Publication R99/04. Available from http://www.census.gov/srd/www/byname.html.