Approximate String Comparison and its Effect on an Advanced Record Linkage System

Edward H. Porter, Bureau of the Census
William E. Winkler, Bureau of the Census

KEY WORDS: string comparator, bigram, assignment algorithm, EM algorithm, latent class.

Record linkage, sometimes referred to as information retrieval (Frakes and Baeza-Yates 1992), is needed for the creation, unduplication, and maintenance of name and address lists. This paper describes string comparators and their effect in a production matching system. Because many lists have typographical errors in more than 20% of first names and also in last names, effective methods for dealing with typographical error can greatly improve matching efficacy. The enhanced methods of approximate string comparison deals with typographical variations and scanning errors. The values returned by the string comparator are used in a statistical model for adjusting parameters that are automatically estimated by an expectation-maximization algorithm for latent class, log linear models of the type arising in the Fellegi-Sunter model of record linkage (1969). Overall matching efficacy is further improved by linear assignment algorithm that forces 1-1 matching.

Modern record linkage represents a collection of methods from three different disciplines: computer science, statistics, and operations research. While the foundations are from statistics, beginning with the seminal work of Newcombe (Newcombe et al. 1959, also Newcombe 1988) and Fellegi and Sunter (1969), the means of implementing the methods have primarily involved computer science. Record linkage begins with highly evolved software for parsing and standardizing names and addresses that are used in the matching. Name standardization identifies components such as first names, last names (surnames), titles, and middle initials. Address standardization locates components such as house numbers, street names, PO Boxes, apartment numbers, and rural routes. With good standardization, effective comparison of corresponding components of information and the advanced methods described in this paper become possible.

Because pairs of strings often exhibit typographical variation (e.g., Smith versus Smoth), the record linkage needs effective string comparator functions that deal with typographical variations. While approximate string comparison has been a subject of research in computer science for many years (see survey article by Hall and Dowling 1980), some of the most effective ideas in the record linkage context were introduced by Jaro (1989; see also Winkler 1985, 1990). Budzinsky (1991), in an extensive review of twenty string comparison methods, concluded that the original Jaro method, the extended method due to Winkler (1990), and a widely used computer science method called bigrams worked well. This paper describes two new enhancements to the string comparators used at the Census Bureau. The first, due to McLaughlin (1993), adds logic for dealing with scanning errors (e.g., "I" versus "1") and certain common keypunch errors (e.g., "V" versus "B"). The second due to Lynch and Winkler (1994) makes adjustments for pairs of long strings having a high proportion of characters in common. We also describe the method of computing bigrams and present results comparing them with the other string comparators of this paper.

Our record linkage system uses the Expectation-Maximization (EM) algorithm (Dempster, Laird, and Rubin) to estimate optimal matching parameters. We use a linear sum assignment

procedure (lsap) to force 1-1 matching. Jaro (1989) introduced the lsap as a highly effective means of eliminating many pairs that ordinarily might be clerically reviewed. With a household data source containing multiple individuals in a household, it effectively keeps the four pairs associated with father-father, mother-mother, son-son, and daughter-daughter pairs while eliminating the remaining twelve pairs associated with the household.

The next section describes the string comparator. In the third section, we provide a summary of the parameters that are obtained via the EM algorithm. The results of section four provide empirical examples of how matching efficacy is improved for three, small pairs of high quality lists. The final section consists of a summary and conclusion.

Approximate String Comparison

Dealing with typographical error can be vitally important in a record linkage context. If comparisons of pairs of strings are only done in an exact character-by-character manner, then many matches may be lost. An extreme example is the Post Enumeration Survey (PES) (Winkler and Thibaudeau 1991, also Jaro 1989) in which, among true matches, almost 20 percent of last names and 25 percent of first names disagreed character-by-character. If matching had been performed on a character-by-character basis, then more than 30 percent of matches would have been missed by computer algorithms that were intended to delineate matches automatically. In such a situation, required manual review and (possibly) matching error would have greatly increased.

Jaro (1989) introduced a string comparator that accounts for insertions, deletions, and transpositions. In a small study, Winkler (1985) showed that the Jaro comparator worked better than some others from computer science. In a large study, Budzinsky (1991) concluded that the comparators due to Jaro and Winkler (1990) were the best among twenty in the computer science literature. The basic Jaro algorithm is: (1) compute the string lengths, (2) find the number of common characters in the two strings, and (3) find the number of transpositions. The definition of common is that the agreeing character must be within ½ the length of the shorter string. The definition of transposition is that the character from one string is out of order with the corresponding common character from the other string. The string comparator value (rescaled for consistency with the practice in computer science) is:

```
jaro(s1,s2) = 1/3( #common/str_len1 + #common/str_len2 +
          0.5 #transpositions/#common), (2.1)
```

where s1 and s2 are the strings with lengths str_len1 and str_len2, respectively.

The new string comparator algorithm begins with the basic Jaro algorithm and then proceeds to three additional loops corresponding to the enhancements. Each enhancement makes use of information that is obtained from the loops prior to it.

The first enhancement due to McLaughlin (1993) assigns value 0.3 to each disagreeing but similar character. Each exact agreement gets value 1.0 and all exact agreements are located prior to searching for similar characters. Similar characters might occur because of scanning errors ("1" versus "l") or keypunch ("V" versus "B"). The number of common characters (#common) in equation (2.1) gets increased by 0.3 for each similar character, is denoted by #similar, and #similar

is substituted for #common in the first two components of equation (2.1).

The second enhancement due to Winkler (1990) gives increased value to agreement on the beginning characters of a string. It was based on ideas from a very large empirical study by Pollock and Zamora (1984) for the Chemical Abstracts Service. The study showed that the fewest errors typically occur at the beginning of a string and the error rates by character position increase monotonically as the position moves to the right. The enhancement basically consisted of adjusting the string comparator value upward by a fixed amount if the first four characters agreed; by lesser amounts if the first three, two, or one characters agreed. The string comparator examined by Budzinsky (1991) consisted of the Jaro comparator with only the Winkler enhancement.

Table 2.1 Comparison of String Comparators Using Last Names, First Names, and Street Names

Two strings		String comparator values				
		Jaro	Wink	McLa	Lynch	Bigram
SHACKLEFORD	SHACKELFORD	0.970	0.982	0.982	0.989	0.925
DUNNINGHAM	CUNNIGHAM	0.896	0.896	0.896	0.931	0.917
NICHLESON	NICHULSON	0.926	0.956	0.969	0.977	0.906
JONES	JOHNSON	0.790	0.832	0.860	0.874	0.000
MASSEY	MASSIE	0.889	0.933	0.953	0.953	0.845
ABROMS	ABRAMS	0.889	0.922	0.946	0.952	0.906
HARDIN	MARTINEZ	0.000	0.000	0.000	0.000	0.000
ITMAN	SMITH	0.000	0.000	0.000	0.000	0.000
JERALDINE	GERALDINE	0.926	0.926	0.948	0.966	0.972
MARHTA	MARTHA	0.944	0.961	0.961	0.971	0.845
MICHELLE	MICHAEL	0.869	0.921	0.938	0.944	0.845
JULIES	JULIUS	0.889	0.933	0.953	0.953	0.906
TANYA	TONYA	0.867	0.880	0.916	0.933	0.883
DWAYNE	DUANE	0.822	0.840	0.873	0.896	0.000
SEAN	SUSAN	0.783	0.805	0.845	0.845	0.800
JON	JOHN	0.917	0.933	0.933	0.933	0.847
JON	JAN	0.000	0.000	0.860	0.860	0.000
BROOKHAVEN	BRROKHAVEN	0.933	0.947	0.947	0.964	0.975
BROOK HALLOW	BROOK HLLW	0.944	0.967	0.967	0.977	0.906
DECATUR	DECATIR	0.905	0.943	0.960	0.965	0.921
FITZRUREITER	FITZENREITER	0.856	0.913	0.923	0.945	0.932
HIGBEE	HIGHEE	0.889	0.922	0.922	0.932	0.906
HIGBEE	HIGVEE	0.889	0.922	0.946	0.952	0.906
LACURA	LOCURA	0.889	0.900	0.930	0.947	0.845
IOWA	IONA	0.833	0.867	0.867	0.867	0.906
1ST	IST	0.000	0.000	0.844	0.844	0.947

The final enhancement due to Lynch and Winkler (1994) adjusts the string comparator value if the strings are longer than six characters and more than half the characters beyond the first four

agree. The final enhancement was based on detailed comparisons between versions of the comparator. The comparisons involved tens of thousands of pairs of last names, first names, and street names that did not agree on a character-by-character basis but were associated with truly matching records.

A common string comparison methodology is comparing the bigrams that two strings have in common. A bigram is two consecutive letters with a string. Hence the word "bigram" contains the bigrams "bi" "ig" "gr" "ra", and "am". The bigram function also returns a value between 0 and 1. The return value is the total number of bigrams that are in common divided by the average number of bigrams in the two strings. Bigrams are known to be a very effective, simply programmed means of dealing with minor typographical errors. They are widely used by computer scientists working in information retrieval (Frakes and Baeza-Yates 1992).

Table 2.1 illustrates the effect of the new enhanced comparators on last names, first names, and street names, respectively. To make the value returned by bigram weighting function more comparable to the other string comparators, we make a functional adjustment. If x is the value returned by the bigram weighting function, we use $f(x) = x^{0.2435}$ if x is greater than 0.8 and 0.0 otherwise. If each string in a pair is less than four characters, then the Jaro and Winkler comparators return the value zero. The Jaro and Winkler comparator values are produced by the loop from the main production software (e.g., Winkler and Thibaudeau 1991) which is only entered if the two strings do not agree character-by-character. The return value of zero is justified because if each of the strings has three or less characters, then they necessarily disagree on at least one.

In record linkage situations, the string comparator value is used in adjusting the matching weight associated with the comparison downward from the agreement weight toward the disagreement weight. Using crude statistical modeling techniques, Winkler (1990) developed downweighting functions for last names, first names, street names, and some numerical comparisons that generalized the original downweighting function introduced by Jaro.

Data and Matching Weights - Parameters

In this section, we describe the fields and the associated matching weights that are used in the record linkage decision rule. We do not give details of the EM algorithm or the assignment algorithm because they have been given elsewhere (Winkler 1994).

The fields used in the creation of mailing list during the 1995 test census are first name, last name (surname), sex, month of birth, day of birth, year of birth, race, and Hispanic origin. The census file is linked with an update file. These update files have been either I.R.S., driver's license, or school records. Only fields whose housing unit identifier agreed are compared in the first pass. The housing unit identifiers were calculated by the Census Bureau's geography division's address standardization software. It consists of a State Code, County Code, TIGER Line Id (e.g. a city block), Side Id (right or left), house number, and apartment number. In the 1995 test census of Oakland, California, 95.0% of the records file were geocoded with housing unit identifier. Also, 94.7% of the I.R.S. file records for the corresponding area were geocoded with housing unit identifier. The names were standardized at a 95.2% rate in the test census file and 99.0% rate in the I.R.S. file.

Each parameter was assigned an agreement and disagreement weight. Certain parameters such as first name are assigned a higher agreement weight. Since matching was done within a household, surname carried had less distinguishing power than first name. After initial trial runs

and research of the output, the expectation-maximization software (EM) was run to produce the parameters for the test.

Table 3.1 Parameters used in matching for the 1995 test census of Oakland, California.

Parameter	Agreement weight	Disagreement weight
first	4.3385	-2.7119
last(surname)	2.4189	-2.5915
sex	0.7365	-3.1163
month	2.6252	-3.8535
day	3.5206	-2.9652
year	1.7715	-4.1745
Hispanic	0.2291	-0.3029
race	0.5499	-0.5996

String comparators were only used with first names and surnames. For example if the first names were Martha and Marhta. The matching weight would be computed as follows.

	Jaro	Wink	McLa	Lynch
Comparator Value	0.944	0.961	0.961	0.971
Matching Weight	3.943	4.063	4.063	4.134

The piecewise linear function that uses the value returned by the different string comparators to adjust the matching agreement weight downward is detailed in Winkler (1990).

Results

Results are presented in two parts. In each part, the different string comparators are substituted in the string comparison subroutine of an overall matching system. The matching weights returned by the EM algorithm are held constant. Two different versions of a linear sum assignment procedure are used. For the description of the lsap, see Winkler 1994. The main empirical data consists of three pairs of files having known matching status. In the first part, we show how much the string comparators can improve the matching results. The second part provides an overall comparison of matching methods that utilize various combinations of the new and old string comparators and the new and old assignment algorithms.

Exact Matching Versus String Comparator Enhanced Matching

In Figure 4.1, we illustrate how much string comparators improve matching in comparison with exact matching. After ordering pairs by decreasing matching weight in the first and third of the empirical data files, we plot the proportion of false matches against the total number of pairs. We see that, if matching is adjusted for bigrams and the string comparators, then error

rates are much lower than those obtained when exact matching is used. Since exact matching is not competitive, remaining results are only presented when string comparators are used.

Table 4.1 Matching Results At Different Error Rates
1st Pair of Files with 4539 and 4859 records
38795 Pairs Agreeing on Block and First Char Last

Link Error Rate	Link match/nonm	Clerical match/nonm
0.002		
base	3172/ 6	242/64
s_c	3176/ 6	236/64
as	3176/ 6	234/64
os_l	3174/ 6	242/64
bigram	3224/ 7	174/63
0.005		
base	3363/17	51/53
s_c	3357/17	55/53
as	3357/17	53/53
os_l	3364/17	52/53
bigram	3327/17	71/53
0.010		
base	3401/34	13/36
s_c	3396/34	16/36
as	3396/34	14/36
os_l	3402/34	14/36
bigram	3376/34	22/36
0.020		
base	3414/70	0/ 0
s_c	3411/70	0/ 0
as	3410/70	0/ 0
os_l	3416/70	0/ 0
bigram	3398/70	0/ 0

Overall Comparison of Matching Methods

The baseline matching is done under 3-class, latent class models under the conditional independence assumption. The 3-class models are essentially the same ones used in Winkler (1994). Results are reported for error rates of 0.002, 0.005, 0.01, and 0.02, respectively. *Link*, *Nonlink*, and *Clerical* (*or Possible Link*) are the computer designations, respectively. *Match* and *Nonmatch* are the true statuses, respectively. The baseline results (designated by *base*) are

Table 4.2 Matching Results At Different Error Rates
2nd Pair of Files with 5022 and 5212 records
37327 Pairs Agreeing on Block and First Char
Last

Link Error Rate	Link match/nonm	Clerical match/nonm
0.002		
base	3475/ 7	63/65
s_c	3414/ 7	127/65
as	3414/ 7	127/65
os_l	3477/ 7	63/65
bigram	3090/ 7	461/66
0.005		
base	3503/18	35/54
s_c	3493/18	48/54
as	3493/18	48/54
os_l	3505/18	36/54
bigram	3509/18	42/55
0.010		
base	3525/36	13/36
s_c	3526/36	15/36
as	3526/36	15/36
os_l	3527/36	14/36
bigram	3543/36	8/73
0.020		
base	3538/72	0/ 0
s_c	3541/72	0/ 0
as	3541/72	0/ 0
os_l	3541/72	0/ 0
bigram	3551/73	0/ 0

produced using the existing lsap algorithm and the previous string comparator (see e.g., Winkler 1990) but use the newer, 3-class EM procedures for parameter estimation (Winkler 1994). The results with the new string comparator (designated *s_c*) are produced with the existing string comparator replaced by the new one. The results with the new assignment algorithm (designated *as*) use both the new string comparator and the new assignment algorithm. For comparison, results produced using the previous string comparator but with the new assignment algorithm (designated by *os_l*) are also given. Finally, results using the bigram adjustments are denoted by *bigram.*

Matching efficacy improves if more pairs can be designated as links and nonlinks at fixed error rate levels. In Tables 4.1-3, computer-designated links and clerical pairs are subdivided into (true) matches and nonmatches. Only the subset of pairs produced via 1–1 assignments

Table 4.3 Matching Results At Different Error Rates
3rd Pair of Files with 15048 and 12072 Records
116305 Pairs Agreeing on Block and First
Character of Last Name

Link Error Rate	Link match/nonm	Clerical match/nonm
0.002		
base	9696/19	155/182
s_c	9434/19	407/182
as	9436/19	406/182
os_l	9692/19	157/182
bigram	9515/19	335/182
0.005		
base	9792/49	59/152
s_c	9781/49	60/152
as	9783/49	57/152
os_l	9791/49	58/152
bigram	9784/49	66/152
0.010		
base	9833/99	18/102
s_c	9822/99	19/102
as	9823/99	17/102
os_l	9831/99	18/102
bigram	9823/99	27/102
0.020		
base	9851/201	0/ 0
s_c	9841/201	0/ 0
as	9842/201	0/ 0
os_l	9849/201	0/ 0
bigram	9850/201	0/ 0

are considered. In producing the tables, pairs are sorted by decreasing weights. The weights vary according to the different model assumptions and string comparators used. The number of pairs above different thresholds at different link error rates (0.002, 0.005, 0.01, and 0.02) are presented. False match error rates above 2 percent are not considered because the sets of pairs above the cutoff threshold contain virtually all of the true matches from the entire set of pairs when error rates rise to slightly less than 2 percent. In each line, the proportion of nonmatches (among the sum of all pairs in the Link and Clerical columns) is 2 percent.

The results generally show that the different string comparators improve matching efficacy. In all of the best situations, error levels are very low. The new string comparator produces worse results than the previous one (see e.g., Winkler 1990) and the new assignment algorithm (when combined with the new string comparator) performs slightly worse (between 0.1 and 0.01 percent) than the existing string comparator and lsap algorithm. In all situations (new or old string

comparator), the new assignment algorithm slightly improves matching efficacy.

To test the effect of the Winkler variant of the Jaro string comparator and bigrams on more recent files, we use 1995 test census files from Oakland, California. The match rates were as follows. In the first matching pass, we only used pairs of records that agreed on housing unit ID. Those that were not matched were processed in a second pass. Blocking during the second pass was on house number and first character of the first name. The results generally show that either string comparator produces good results. The variant of the Jaro string comparator yields a slightly smaller clerical review region.

Table 4.4 First Pass--Housing Unit Identifier match. Matching results of a pair files with 226,713 and 153,644 records, respectively

	Jaro String Comparator		Bigram	
	Links	Clerical	Links	Clerical
	78814	5091	78652	5888
Estimated False Match Rate	0.1%	30%	0.1%	35%

Second Pass--House Number and First Character of first name. Matching results of a pair of files with 132,100 and 64,121 records, respectively

	Links	Clerical
	16893	7207
Estimated False Match Rate	0.3%	40%

Summary and Conclusion

Application of new string comparator functions can improve matching efficacy in the files having large amounts of typographical error. Since many of the files typically have high typographical error rates, the string comparators can yield increased accuracy and reduced costs in matching of administrative lists and census.

References

Budzinsky, C. D. (1991), "Automated Spelling Correction," Statistics Canada.

Fellegi, I. P., and Sunter, A. B. (1969), "A Theory for Record Linkage," *Journal of the American Statistical Association*, **64**, 1183-1210.

Frakes, W. B., and Baeza-Yates, R. (ed.) (1992), Information Retrieval: Data Structures & Algorithms, Upper Saddle River, NJ: Prentice-Hall PTR.

Hall, P. A. V., and Dowling, G. R. (1980), "Approximate String Comparison," *Computing Surveys*, **12**, 381-402.

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*, **89**, 414-420.

Lynch, M. P., and Winkler, W. E. (1994), "Improved String Comparator," technical report, Statistical Research Division, Washington, DC: U.S. Bureau of the Census.

McLaughlin, G. (1993), Private communication of C-string-comparison routine.

Newcombe, H. B. (1988), *Handbook of Record Linkage: Methods for Health and Statistical Studies, Administration, and Business*, Oxford: Oxford University Press.

Newcombe, H. B., Kennedy, J. M., Axford, S. J., and James, A. P. (1959), "Automatic Linkage of Vital Records," *Science*, **130**, 954-959.

Pollock, J. and Zamora, A. (1984), "Automatic Spelling Correction in Scientific and Scholarly Text," *Communications of the ACM*, **27**, 358-368.

Winkler, W. E. (1985), "Preprocessing of Lists and String Comparison," in W. Alvey and B. Kilss, (eds.) *Record Linkage Techniques*- 1985, U.S. Internal Revenue Service, Publication 1299 (2-86), 181-187.

Winkler, W. E. (1990), "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage," *Proceedings of the Section on Survey Research Methods, American Statistical Association*, 354-359.

Winkler, W. E. (1994), "Advanced Methods for Record Linkage," technical report, Statistical Research Division, Washington, DC: U.S. Bureau of the Census.

Winkler, W. E. (1995), "Matching and Record Linkage," in B. G. Cox (ed.) *Survey Methods for Businesses, Farms, and Institutions*, New York: J. Wiley.

Winkler, W. E., and Thibaudeau, Y. (1991), "An Application of the Fellegi-Sunter Model of Record Linkage to the 1990 U.S. Decennial Census," Statistical Research Division Report 91/09, Washington, DC: U.S. Bureau of the Census.

Figure 4.1. Effects of String Comparators

Proportion of False Matches versus Total Pairs
Sorted By Decreasing Matching Weight