Journal of Information Technology Management

Print ISSN: 2008-5893

Online ISSN: 2423-5059

Estimating the Parameters for Linking Unstandardized References with the Matrix Comparator

Awaad Al-Sarkhi

*Corresponding author, University of Arkansas at Little Rock, USA. E-mail: aalsarkhi@ualr.edu

John R. Talburt

Associate Professor, University of Arkansas at Little Rock, USA. E-mail: jrtalburt@ualr.edu

Abstract

This paper discusses recent research on methods for estimating configuration parameters for the Matrix Comparator used for linking unstandardized or heterogeneously standardized references. The matrix comparator computes the aggregate similarity between the tokens (words) in a pair of references. The two most critical parameters for the matrix comparator for obtaining the best linking results are the value of the similarity threshold and the list of stop words to exclude from the comparison. Earlier research has shown that the standard deviation of the token frequency distribution is strongly predictive of how useful stop words will be in improving linking performance. The research results presented here demonstrate a method for using statistics from token frequency distribution to estimate the threshold value and stop word selection likely to give the best linking results. The model was made using linear regression and validated with independent datasets.

Keywords: Entity resolution, Record linking, Matrix comparator, Stop words, Token frequency, F-measure.

DOI: 10.22059/jitm.2019.274871.2332

Introduction

The process of Entity Resolution (ER) is measuring whether two references to real-world objects in an information system are referring to the same purpose, or different objects (Talburt, 2011; Hernández, & Stolfo, 1995; Wang, 1998). References to the same entity are called equivalent references. The goal of ER is to link two references if, and only if, the references are equivalent (Kobayashi, Eran & Talburt, 2014; Alsarkhi, & Talburt, 2018; Elmagarmid, Ipeirotis, & Verykios, 2007; Kobayashi, & Talburt, 2018; Agichtein, & Ganti, 2004; Moustakides, & Verykios, 2009). For this reason, the ER is sometimes referred to as record linking.

Precision, recall, and F-measure have generally accepted measures for assessing the quality of the linking results from an ER process. The linking precision is the ratio of true positive links (links between equivalent references) to the total number of links made. The linking recall is the ratio of the true positive links to the total number of equivalent pairs (possible true positive links). The F-measure is the harmonic mean of precision and recall.

ER logic is based on the Similarity Assumption (Talburt, & Zhou, 2015; Pullen, Wang, Talburt, & Wu, 2013) which states "the more similar two references are, the more likely they are equivalent, and the less similar they are, the less likely they are equivalent." Both deterministic and probabilistic ER systems start by first assessing the similarity of corresponding attributes in each reference such as the similarity of first names, last names, street numbers, and date-of-birth. However, this approach assumes the attribute values in both references have metadata tags to indicate their usage, and all references use the same metadata tags.

The process to create a uniform set of metadata tags is called data standardization, and ER processes typically rely on having standardized input references. However, when there are many different sources of data, the standardization process may require a great deal of time and effort to harmonize (Jurek-Loughrey & Deepak, 2018). Even if a data provider has already standardized each source, different sources may have different standardizations. For example, one source may have normalized the references to having a separate field (tag) for the street number, and another field (tag) for the street name whereas another source's standardization has the street number and street name together as a single street address field.

The organization of this paper is as follows:

- Section II describes the logic of the Matrix Comparator for performing ER on unstandardized and heterogeneously standardized references
- Section III summarizes previous research on the effectiveness of using stop words to improve the quality of ER results produced by the matrix comparator

- Section IV describes new research for predicting the value of critical parameters of the matrix comparator, in particular, the matching threshold value, and the list of stop words
- Section V assessment of prediction model
- Section VI conclusion and future work

Logic of the Matrix Comparator

An ER method for avoiding the need for transforming references into a standard layout before processing is the matrix comparator (Li, Talburt, & Li, 2018). Given a pair of references to compare, each reference, excluding the unique record identifier, is first transformed into a list of tokens (word strings). The tokens are created by treating the entire reference as a single string of characters, then replacing all non-word characters in the reference (characters other than letters and digits) with a blank. After the replacement, the reference string is split into substrings (tokens) delimited by blanks. In a final step, all letter characters in each token are converted to upper case. For example, the reference string "A087, Mary Jones-Smith, 31 Oak St #12" would give the string "Mary Jones Smith 31 Oak St 12" after non-word character replacement and discarding the record identifier. Then splitting the string on blanks and uppercasing would produce seven tokens: "MARY", "JONES", "SMITH", "31", "OAK", "ST", and "12".

In the matrix comparator, the tokens from the first reference are used to label the rows of the matrix, and the tokens from the second string label the columns of the matrix. The cell value is the similarity measure between the two tokens. There are several algorithms available to measure string similarities such as Jaccard, Levenshtein, and Jaro-Winkler. For the research described in this paper, the Levenshtein Edit Distance (LED) was used to assess the similarity between each row token and column token. Given two character strings, the LED function calculates the distance between the strings as the minimum number of canonical character operations (insert, deleted, substitute) necessary to transform one string into the other. Because the number of processes can vary from zero up to the length of the longest string (worst case), the function is often normalized (nLED) as a rating metric to produce an amount in the interval [0, 1]. Where the value is 1.0 if, and only if, the strings are equal, i.e., LED distance is zero. The formula for the nLED is

1) $$nLED(string1, string2) = ١.٠ - \frac{LED(string1, string2)}{Max\{string1.len, string2.len\}}$$

To illustrate how the matrix comparator works consider the following two references.

- A045, Smith, John, Apt 21, 345 Oak St, Anytown, NY
- B167, Jon Smith, 345 Oak Street #21, Anytown, NY

Furthermore, suppose the threshold for the comparator has been set to 0.80, and the list of stop words contains the token "NY". The resulting token matrix would then appear as shown in Figure 1. The value in each cell is an nLED similarity between the tokens labeling the row and column of the cell. For example, the nLED similarity between the tokens "JOHN" and "JON" is 0.75 because their edit distance is 1 (inserting an H in "JON" or deleting an H from "JOHN"), hence the nLED similarity between "JOHN" and "JON" is ١ − ١/٤ = ٠.٧٥.

Table 1. Example Token Matrix

	JON	**SMITH**	**345**	**OAK**	**STREET**	**21**	**ANYTONW**
SMITH		1.00			0.17		0.14
JOHN	0.75			0.25			0.14
APT		0.20			0.17		0.29
21						1.00	
345			1.00				
OAK				1.00			0.14
ST		0.40			0.33		0.14
ANYTOWN	0.29	0.14		0.14			0.71

In Table 1, the cells where the nLED value is 0.00 have been left blank for readability. The token "NY" does not appear in the matrix because it is in the list of stop word values. This example also illustrates how the matrix comparator is not dependent upon the order of the tokens in each reference. The reference A045 gives the name in last-name-first order, whereas reference B167 gives it in first-name-first order. Similarly, in A045, the apartment number comes before the street address, and in B167, it happens after.

After the cells of the matrix have been populated with nLED similarity values, the comparator systematically selects the nLED values in descending order from each row and column in an iterative process. In the first iteration, the row and column with the largest LED value for the entire matrix are identified, and this value is the starting value of the overall total. After the largest value is found in a row and column, the nLED values in the same row and same column removed. For example in Figure 1, the first maximum is 1.00 between the "SMITH" tokens. As a result, the values 1.00, 0.17 and 0.14 in Row 1, and the value 0.2 in Column 2 are removed before starting the next iteration.

In the next iteration, the largest value of the remaining similarity values in the matrix is identified, added to the overall total, and again, all of the nLED values in the same row and column are removed. The process continues in subsequent iterations until all of the values have been removed from the matrix.

The number of iterations will be equal to the number of tokens from the reference generating the fewest tokens. After the last iteration, the running total is divided by the

number of repetitions. If the calculated average value is greater than or equal to a threshold value provided by the user, then the comparator returns a "true" result and links the references. Otherwise, the comparison yields a "false" effect, and the references are not linked. At the end of the algorithm, the final matrix score for a pair of references in Table 1 is 0.83. Because 0.83 is above the 0.80 thresholds, these two references would be linked.

An essential feature of the Matrix Comparator is the optional use of stop words (Li & et al., 2018). A stop word feature is a simple form of token weighting in which the most frequently occurring words are given a weight of zero, i.e., are excluded from the matrix, and all other tokens are given a weight of one. Stop word represent a simplification of the frequency-inverse document frequency (tf-idf) technique often applied in document retrieval (Salton & Buckley, 1998). In this technique, tf represents how often a word (token) occurs in a particular document whereas idf represents the ratio of the number of documents containing the term versus the total number of documents. In applying tf-idf to ER, each reference can be considered a document.

In most cases, the reference is a relatively small document, and the term frequency value tf is almost always one or zero. Hence, the inverse document frequency is just the inverse of the frequency of the token across all references. However, for this research, binary weights were selected because, in the preliminary study, the inverse frequency weights did not produce better results than the binary weights.

The motivation to stop words the matrix comparator and tf-idf , in general, is that tokens with a very high frequency across all references are less indicative of similarity because they are as likely to co-occur in non-equivalent references (documents) as they are to co-occur in equivalent references (documents). For example, if a set of references all have addresses in the state of New York, then unless a non-standard abbreviation or misspelling occurs, the state abbreviation token "NY" will be found in almost every reference. The co-occurrence of "NY" will always contribute a similarity of 1.00 to the matrix score. Excluding "NY" should cause the average similarity score to be based on other tokens less frequently occurring and presumably more indicative of equivalence.

The primary finding of the prior research is that the use of stop words for the matrix comparator is not always warranted. There are cases where using high-frequency tokens as stop words does not significantly improve linking results, and in fact, may degrade the results.

Summary of Prior Research Method and Results

The experiments for both the prior research (Alsarkhi & Talburt, 2018). And the new study is based on annotated references taken from four different sources. Two of the sources are synthetically generated references designed to represent customer data. Different processes created the two synthetic sources. The first source was generated by the Synthetic Occupancy Generator (SOG) (Talburt, Zhou, & Shivaiah, 2009) designed to simulate the movement over

time of consumers (persons) from address-to-address and changes of name through marriage. The SOG source also included gender coding, phone numbers, social security numbers, and date-of-birth. At the same time, some of the data quality problems were deliberately injected into the SOG data to increase its realism. These included issues such as deleted (missing) values, misspellings, transpositions, truncations, and the inconsistent data and telephone formats.

Most importantly for ER research, the SOG corpus includes many redundant (duplicate) references to the same customers in different file formats and with different information and data quality problems. The SOG corpus comprises a total of 271K records in three different file layouts.

The second synthetic source also represents customer references but was generated with an R package called "rlErrorGeneratoR" (rlErrorGeneratoR, n.d.). The R-generator was also designed to produce realistic consumer data with various levels of data quality problems including duplicate records. The R corpus comprises about 800K references in total, even in three different file formats.

Both the SOG and R corpora also have a separate annotation file in the form of a crosswalk table listing every generated reference associated with its original entity identifier. While every reference in each corpus has a different record identifier, various references to the same entity (customer) will still have the same entity identifier in the crosswalk table. The crosswalk table allows an ER metrics program to quickly join ER linking output to the crosswalk table by the corresponding record identifier. This allows the ER metrics program to count the true positive links, false-positive links, and false-negative links, then calculate the precision, recall, and F-measure of the linking outcome. Because we are not assuming any constraints for minimum precision or recall, we take the maximum F-measure to represent the highest quality linking results.

Stratified samples of approximately 5,000 references were drawn from each corpus. Stratification was used because it is unlikely that records selected at random will be equivalent. To create samples exhibiting a reasonable level of linking, the first step of the sampling is to append the entity identifier from the crosswalk table to each record in the file. Next, the data was sorted by the entity identifier to bring together groups of equivalent reference. After sorting by entity identifier, a segment of 5,000 consecutive records was selected from a random starting point in the sorted file. Because references to the same customer are in adjacent records in the sorted data, this method of stratified sampling guarantees each sample would contain a significant number of true positive pairs.

Samples 1-4 and 7-8 were taken from the SOG corpus. Samples 9-17 were taken from the R corpus. In all, 15 samples were drawn from the two annotated synthetic corpora. Examples of synthetic customer records are shown here.

- A926344: ANDREW, AARON, STEPHEN, 2475 SPICEWOOD DR, WINSTON SALEM, NC, 27106, 601-70-6106, (159)-928-5341

- A930444: A, AARON, STEPHEN, 2475 SPICEWOOD DR, WINSTON SALEM, NC, 27106, 601706106, (159)9285341

The experimental data also includes two real-world data sets. Sample 5 is a file of 866 references to restaurants (businesses). The references are from two different sources, Zagat's and Fodor's restaurant guides. The references contain restaurant names, addresses, city, phone, and cuisine. The file has been manually annotated with a cross-walk table and is known to have 112 pairs of equivalent references (Tejada, n.d.). Examples of restaurant references are shown here.

- A001: Arnie Morton's of Chicago 435 S. La Cienega Blvd. Los Angeles 310-246-1501 Steakhouses
- A002: Arnie Morton's of Chicago 435 S. La Cienega Blvd. Los Angeles 310/246-1501 American

The following real-world data set comprises 4,910 references to published research papers. The references were taken from the DBLP Computer Science Bibliography and the Association for Computing Machinery (ACM) Digital Library. The file has been manually annotated with cross-walk table and is known to have 2,224 pairs of equivalent references (Reuther, 2019). Examples of DBLP references are shown here.

Conf/sigmod/SlivinskasJS01: Adaptable query optimization and evaluation in temporal middleware, Giedrius Slivinskas Christian S. Jensen Richard Thomas Snodgrass, International Conference on Management of Data, 2001

375678: Adaptable query optimization and evaluation in temporal middleware, Giedrius Slivinskas Christian S. Jensen Richard Thomas Snodgrass, International Conference on Management of Data, 2001

It is important to note that for all samples, any metadata tagging in the example was ignored. When samples had separate fields for name, address, or other features, the values for these fields were concatenated into one string representing the entire reference. Only the unique record identifier was kept as a separately tagged field to join the linking results back to the cross-walk table to enable the ER metrics program to generate the F-measure of the linking results.

The first step in the analysis of each sample was to perform a frequency analysis on the tokens in the example. This step was implemented by using a regular expression to tokenize each reference into sub-references delimited by non-word characters (regex group '\W'). The collected tokens were then converted to upper-case letters and sorted to produce a token frequency table. Finally, the tokens were sorted into descending order by frequency to identify the highest frequency tokens as candidates for stop words.

Three statistics were calculated for the frequency distribution of tokens from each sample:

1. The average frequency

٢. The standard deviation of the frequency distribution

٣. The ratio of the highest frequency to the sample size called the "top ratio."

The next step was to determine the number of stop words producing the best F-measure for each sample. Two trial-and-error processes implemented this step. The first process was to select the number of high-frequency stop words giving the best F-measure results. The starting point was a baseline of no (zero) stop words, then increasing the number of stop words in increments of 25.

For each selection of stop words, the second trial-and-error process was performed to find the matching threshold producing the best result (best F-measure) for the given set of stop words. This was done by again starting with a low threshold, running the ER process, using the cross-walk table to calculate the F-measure of the result, then incrementing the threshold and repeating the process. The ER processes for all of the experiments described here were performed using OYSTER, an open source ER system (Oyster Open Source Project, n.d.) available on BitBucket under the "Oyster Project".[١] In OYSTER, the matrix comparator is implemented as a function of the form MatrixComparator (d.dd, 'a|b|c|d…')

where d.dd represents the matching threshold given as a number from 0.00 to 1.00, and 'a|b|c|d' represents a list of stop words separated by a pipe (|) delimiter.

Table 1 shows the results obtained from this process for 17 reference samples. The Start F-Measure is the baseline showing the best measure achieved without using stop words. The Best F-measure is the best measure achieved when using stop words. The Threshold and Stop Words columns give the matrix comparator parameters yielding the best F-measure. The column labeled "Effect" records whether using stop words improved the F-measure (Pos), had little or no effect (None) or degraded the linking results (Neg).

Table 1 shows the results of using the matrix comparator to link 17 sets of annotated references. The second column labeled "Baseline" shows the best F-measure of linking results obtained using the matrix comparator without the use of a stop word list. The third column shows the best F-measure of linking results obtained when the matrix comparator used a list of stop words. Column 4 gives the threshold, and Column 5 offers the number of stop words that yielded the best F-measure given in Column 3.

Column 6 indicates the effect of using stop words. For samples 7 and 8, the use of stop words had a negative impact in effect on linking performance (Neg). Samples 3, 4, and six did not show any significant improvement in linking performance when stop words were used, and Sample 5, showed only marginal improvement (None). However, in all of the remaining samples, the use of stop words had a positive effect of substantially improving the linking results (Pos).

١ https://bitbucket.org/oysterer/oyster/

Column 7 gives the standard deviation of the token frequency distribution, and Column 8 is the average token frequency. The Top Ratio presented in Column 9 represents the ratio of the highest frequency token to the total number of references in the sample.

From Table 1 you can observe that the samples benefiting the most from the use of stop words (i.e., Column 6 is Pos) were those samples whose token frequency distributions had a relatively large standard deviation and maximum frequency. The only exception is Sample 5 which has only 866 references.

Table 2. Experimental Results from 17 Test Samples

Sample	Baseline F-Meas w/o Stop Words	F-Meas using Stop Words	Threshold using Stop Words	Nbr of Stop Words	Effect of Stop Words	Std. Dev of Freq Dist	Avg. Token Freq.	Top Ratio	Sample Size
1	0.318	0.522	0.56	1000	Pos	26.89	3.70	0.44	5,000
2	0.322	0.518	0.60	200	Pos	25.77	3.67	0.39	5,000
3	0.293	0.297	0.81	25	None	20.38	3.54	0.31	5,000
4	0.293	0.294	0.81	25	None	20.05	3.52	0.29	5,000
5	0.837	0.887	0.83	20	None	16.99	3.27	0.54	866
6	0.97	0.97	0.99	10	None	31.37	3.82	0.28	4,910
7	0.802	0.802	0.67	0	Neg	3.040	2.48	0.01	5,000
8	0.796	0.796	0.67	0	Neg	3.050	2.45	0.01	5,000
9	0.872	0.930	0.71	200	Pos	55.47	3.96	0.87	5,000
10	0.875	0.934	0.69	150	Pos	57.05	4.27	0.86	5,000
11	0.857	0.922	0.71	150	Pos	55.53	4.00	0.87	5,000
12	0.851	0.913	0.72	300	Pos	54.90	3.96	0.87	5,000
13	0.869	0.912	0.74	400	Pos	58.27	4.51	0.85	5,000
14	0.901	0.931	0.79	100	Pos	61.31	4.74	0.82	5,000
15	0.872	0.930	0.70	200	Pos	67.16	4.18	0.87	7,000
16	0.847	0.89	0.71	100	Pos	72.49	4.31	0.86	8,000
17	0.869	0.90	0.77	200	Pos	83.42	5.07	0.85	9,000

This is illustrated in Table 2. The change in F-measure with different stop words for Sample 1 is shown in Table 2 by the “Improve” graph. The difference (or lack thereof) in F-measure for Sample 3 is shown in Table 2 by the “No Effect” graph. Finally, the chart “Degrade” in Table 2 shows the fall in F-measure exhibited by Sample 7.

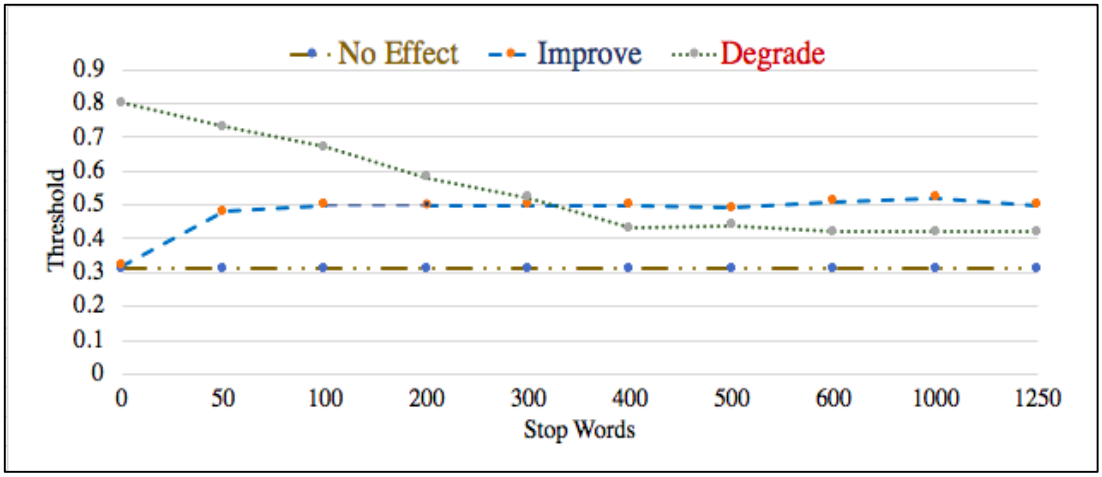

Figure 1. F-Measures by Stop Word Count for Three Samples

The primary conclusion from the prior research (Alsarkhi &, Talburt, 2019) is that the use of stop words is only warranted when the token frequencies are widely dispersed from very large to very small. Using the standard deviation as the measure of token frequency dispersion, except Sample 5, the positive effects only occurred with the standard deviation was 25 or higher. Also, the best improvements were obtained with the highest frequency token occurred in 40% or more of the references, i.e., a top ratio greater than 0.40.

Research to Predict Threshold and Stop Word Counts

While the results of the prior research are insightful, they do not help with one of the most fundamental questions in using the matrix comparator. Given that the token frequency distribution of a set of reference has a significant standard deviation and top ratio, what matching threshold and the number of stop words should be used to obtain the best linking results? In the real world, ER practitioners do not have the annotated truth set and cross-walk table to perform the repeated trials and measurements used in this research.

The approach used to try and answer this question was to revisit the results from the 17 samples analyzed previously. The first step was to remove the five samples where no improvement in F-measure was obtained. This left 12 samples where stop words had a positive effect on the outcome.

Our goal was to create a predictive model based on the characteristics of the references data. In our case, linear regression was selected as the predictive model with standard deviation, top ratio, sample size, and average frequency as candidates for the independent variables to predict the threshold, count of stop words, and best F-measure.

After experimenting with several combinations of the four independent variables, the best model was based on the standard deviation and the sample size as the independent

variables. The linear regression formulas (Hu, 2014) obtained for each of the three dependent variables were

- Best F-Measure = (0.6904) + (0.0130) x Std_Dev + (-9.75e-05) x Sample_Size
- Number of Stop Word = (0.0728) + (-0.0023) x Std_Dev + (-1.591e-05) x Sample_Size
- Match Threshold = (0.6904) + (0.0050) x Std_Dev + (-4.704e-05) x Sample_Size

OLS Regression Results

Dep. Variable:	F-Measure	R-squared:	0.939
Model:	OLS	Adj. R-squared:	0.926
Method:	Least Squares	F-statistic:	69.74
Date:	Fri, 18 Jan 2019	Prob (F-statistic):	3.32e-06
Time:	10:18:06	Log-Likelihood:	22.723
No. Observations:	12	AIC:	-39.45
Df Residuals:	9	BIC:	-37.99
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.6904	0.038	18.320	0.000	0.605	0.776
Size	-9.75e-05	1.1e-05	-8.902	0.000	-0.000	-7.27e-05
Freq Std Dev	0.0130	0.001	11.751	0.000	0.010	0.015

Omnibus:	0.033	Durbin-Watson:	1.709
Prob(Omnibus):	0.984	Jarque-Bera (JB):	0.260
Skew:	0.035	Prob(JB):	0.878
Kurtosis:	2.283	Cond. No.	1.78e+04

Figure 2. Regression Results for Modeling Best F-Measures

Figure 2 shows the linear regression results for the model to predict the Best F-Measure based on the standard deviation of the token frequency distribution and the size of the sample.

Assessment of Prediction Model

The final step of the research was to validate the regression model by creating new samples independent of the 12 samples used to build the regression model. For this purpose, four new stratified samples were drawn from the SOG, R corpora and the GeCo Synthetic Data Generation (Tran, Vatsalan, & Christen, 2013). For each Sample, the threshold and number of stop words yielding the best F-measure were found using the truth set cross-walk table as described previously to find optimal parameters.

Table 2 summarizes the results. For each sample, we first use the trial-and-error process to find the threshold and number of stop words giving the highest F-measure. These are the columns labeled “Best Threshold,” “Best Stop Words,” and “Best F-Measure,” respectively in

Table 2. Next, the Sample Size and Std Deviation were input into the model to predict a threshold, the number of stop words, and F-measure in columns labeled "Predicted Threshold," "Predicted Stop Words," and "Predicted F-Measure," respectively in Table 2. Finally, the sample was linked using OYSTER with the predicted threshold and stop word count, and the F-measure was recorded in the column labeled "F-measure with Predicted".

Table 3. Experimental Results Using the Predicted Values from the Regression Model for Four Test Samples

Test Sample	Sample Size	Std. Dev.	Best Stop Words	Best Threshold	Best F-Meas	Predicted Threshold	Predicted Stop Words	Predicted F-Meas	F-measure using Predicted
A	3,000	40.27	100	0.73	0.891	0.75	84	0.921	0.879
B	4,000	48.97	100	0.73	0.894	0.76	95	0.937	0.884
C	7,000	66.84	300	0.71	0.860	0.71	213	0.877	0.855
D	20,000	135.9	100	0.76	0.932	0.77	63	0.940	0.936

For example, using the annotated cross-walk for Test Sample A, trial-and-error established the best F-measure of 0.891 obtained with a threshold of 0.73 and 100 stop words. The regression model predicted the best threshold should be 0.75, the best stop word count to be 84, and the F-measure to be 0.921 when the matrix comparator was used link Sample A in OYSTER using the predicted parameters, the F-measure of the result 0.879. The predicted threshold of 0.75 is very close to the best threshold of 0.73, and the predicted stop word count of 84 is close to the best stop word count of 100. The F-measure of 0.879 is also close to, but smaller than best F-measure of 0.891. For Sample A, the model somewhat over-predicted F-measure as 0.921.

Table 3 shows in all four cases, the model tended to an over-predict threshold, and under-predict the stop word count. The model also tends to over-predict the best F-measure that can be obtained, especially for the smaller samples A and B. At the same time, the actual F-measures obtained by using the predicted threshold and stop word counts for Samples A, B, and C were only slightly smaller than the F-measures obtained by trial-and-error. For Sample D, the predicted parameters yielded a marginally higher F-measure than by trial-and-error. The predicted parameters were able to outperform the trial-and-error parameters for Sample D because the trial-and-error threshold and stop word counts were tested only at specific increments, and as can be seen in the results for Sample D, do not necessarily represent in the absolute best results possible.

Conclusion and Future Work

The matrix comparator can be a practical approach to ER for unstandardized or heterogeneously standardized references. The results of our research shows four datasets exhibiting particular characteristics, the quality of the matrix comparator results can be further improved through the use of stop words. More importantly, the work has demonstrated that for these datasets, it is possible to predict the values of the two critical parameters of the matrix comparator necessary to produce the highest quality linking results, the matching threshold and the number of stop words. The approximations of the optimal values for these two parameters can be predicted using a linear regression model where the independent variables are the standard deviation of the token frequency distribution and the number of references in the dataset.

Our research will continue to validate the results presented here with additional benchmark datasets from different sources, different types of entities, and larger datasets. We also plan to look at increasing the predictive accuracy of the current linear regression model using additional dataset characteristic and to explore other predictive models.

Another line of research is the use of token weights based on the full tf-idf model. As noted earlier, the current work only uses a simplification of this model with two weight values, zero and one. Changes to the matrix comparator logic to apply scaled weight on the interval [0, 1] have yet to be fully implemented and tested.

All of the current research has been based on using the normalized Levenshtein Edit Distance function as the measure of token similarity. One of the weaknesses of this comparator is its penalty for aliases and abbreviations. For example, in customer entities, nicknames such as “Bill” for “William” and “Bob” for “Robert” are considered semantically very similar, but are not syntactically identical. This can occur for other types of entities such as product references where the description of quantity as “dozen” may be represented by the abbreviation “dz” or where “diameter” is represented by “dia” or “diam.”

One approach to similarity is comparator “chaining” or “stacking” where a series of comparators are applied to the same pair of tokens taking the best (maximum) similarity, for example, stacking nLED with Nickname. In this case, the logistic results of Nickname (True/False) would have to convert to a numeric score such as 0.95 for True and 0.00 for false. In this case, the comparator stack nLED+Nickname would produce a value of 0.95 when comparing “Bill” with “William” instead of 0.375 obtained by using nLED alone.

References

Agichtein, E., & Ganti, V. (2004). Mining reference tables for automatic text segmentation. *The Tenth ACM Sigkdd International Conference on Knowledge Discovery and Data Mining*, August, pp. 20-29.

Alsarkhi, A., & Talburt, J. R. (2018). A method for implementing probabilistic entity resolution. *International Journal of Advanced Computer Science and Applications*, 9(11), 7-15.

Alsarkhi, A., & Talburt, J. R. (2018). An analysis of the effect of stop words on the performance of the matrix comparator for entity resolution. *The Journal of Computing Sciences in Colleges*, 34(7), 64-71.

Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering*, 19(1), 1-16.

Hernández, M. A., & Stolfo, S. J. (1995). The merge/purge problem for large databases. *ACM Sigmod Record*, ٢٤(٢), 127-138.

Hu, L. (2014). Research on the application of regression analysis method in data classification. *Journal of Networks*, 9(11), 3151-3157.

Jurek-Loughrey, A., & Deepak, P. (2018). Semi-supervised and unsupervised approaches to recording pairs classification in multi-source data linkage. In *Linking and Mining Heterogeneous and Multi-view Data* (P. Deepak & A. Jurek eds.), pp. 55-78.

Kobayashi, F., Eram, A., & Talburt, J. (2014). Entity resolution using logistic regression as an extension to the rule-based oyster system. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, April 2008, pp. 146-151.

Li, X., Talburt, J.R., & Li, T. (2018). Scoring Matrix for Unstandardized Data in Entity Resolution. *Proceedings of the International Conference on Computational Science and Computational Intelligence CSCI 2018*, pp. 1087-1092.

Moustakides, G. V., & Verykios, V. S. (2009). Optimal stopping: A record-linkage approach. *Journal of Data and Information Quality*, 1(2), 9.

Pullen, D., Wang, P., Talburt, J., & Wu, N. (2013). Mitigating data quality impairment on entity resolution errors in student enrollment data. In *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*.

Reuther, P. (2019) *DBLP-ACM Bibliographic benchmark dataset*. Retrieved April 13, 2019, https://dbs.uni-leipzig.de/en/research/projects/object_matching/fever/benchmark_ datasets_for_ entity_resolution

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), 513-523.

Talburt, J. R. (2011). *Entity resolution and information quality*. Elsevier.

Talburt, J. R., & Zhou, Y. (2015). *Entity information life cycle for big data: Master data management and information integration*. Morgan Kaufmann.

Talburt, J. R., Zhou, Y., & Shivaiah, S. Y. (2009). SOG: A Synthetic Occupancy Generator to Support Entity Resolution Instruction and Research. *MIT International Conference on Information Quality*, pp. 91-105.

Tejada, S. (n.d.). *Restaurant Benchmark Dataset*. Retrieved April 13, 2019, http://www.cs.utexas. edu/users/ml /riddle/data.html

Tran, K. N., Vatsalan, D., & Christen, P. (2013). GeCo: an online personal data generator and corruptor. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 2473-2476.

Wang, R. Y. (1998). A product perspective on total data quality management. *Communications of the ACM*, 41(2), 58-66.

Bibliographic information of this paper for citing:

Al-Sarkhi, Awaad, & Talburt, John R. (2018). Estimating the Parameters for Linking Unstandardized References with the Matrix Comparator. *Journal of Information Technology Management*, 10(4), 12-26.