Journal of Information & Computational Science 12:3 (2015) 957–964 February 10, 2015
Available at http://www.joics.com

A Comparative Evaluation of String Similarity Metrics for Ontology Alignment ⋆

Yufei Sun [a,*], Liangli Ma [a], Shuang Wang [b]

[a] *Department of Computer Engineering, Naval University of Engineering, Wuhan 430033, China*
[b] *Collage of Liberal Arts, Hunan Normal University, Changsha 410205, China*

Abstract

Ontology alignment is regarded as the most perspective way to achieve semantic interoperability among heterogeneous data. The majority of state of art ontology alignment systems used one or more string similarity metrics, while the performance of these metrics were not given much attention. In this paper we first analyze naming variations in competing ontologies, then we evaluate a wide range of string similarity metrics, from the experimental result we can get some heuristic strategies to achieve better alignment results with regard to effectiveness and efficiency.

Keywords: Ontology Alignment; Name Matching; String Similarity Metrics; Comparative Evaluation

1 Introduction

The rapid progress of information and communication technologies have made available a huge of heterogeneous information on the web. To overcome the semantic heterogeneity and integrate distributed information sources, various solutions have been proposed, among which ontology is the most perspective. While since the decentralized nature of semantic web, there must have some variations in representation, so ontologies themselves are heterogeneous even in the same domain. Therefore, if we want to achieve information interoperability, firstly, we must find semantic relevance between entities of competing ontologies, that is ontology alignment(for a comprehensive survey of ontology alignment, see [1]).

Existing approaches of ontology alignment are mostly based on calculating similarities between entities of two ontologies by utilizing various types of features [2]. Exploiting name features is the most direct and effective way, string similarity metrics are pervasive in almost all state of art ontology alignment systems. RiMOM [2] used edit distance, vector space and WordNet, Falcon [3] used ISUB [4], TF-IDF, and YAM++ [5] used Jaro, Smith-Waterman, and so on.

⋆Project supported by the Pre-research Funds from PLA General Armament of China (No. 9140A27040413JB 11407).

*Corresponding author.
Email address: sharesorrows@163.com (Yufei Sun).

1548–7741 / Copyright © 2015 Binary Information Press
DOI: 10.12733/jics20105420

The metrics usually used for ontology alignment are mostly derived from some other research community, such as duplicated record detecting, record linkage, object identification, information integration and nature language processing. So, it's valuable to analyze these metrics' performance for ontology alignment tasks, in particular, this paper seek to answer the questions below:

(1) What is the most effective and efficient string similarity metrics for ontology alignment task?

(2) Does the hybrid method indeed improve the performance in terms of effectiveness and efficiency?

(3) When faced with a certain ontology alignment task, how to select string similarity metrics taking the accuracy and efficiency into account?

In the following we discuss these points on the basis of our experiences from the comparison of these metrics, and discussed rules to select metrics with regard to accuracy and efficiency. In Section 2 we discuss entities' name variations in competing ontologies. The similarity metrics usually used in ontology alignment are discussed in Section 3. The comparative experimental evaluation and results are addressed in Section 4. We conclude this paper in Section 5.

2 A Taxonomy of Name Variations

From our informal analysis performed on a large amount of heterogeneous ontologies, 8 primary categories of the entities names' variations were classified:

Type 1. Syntactic similar but different naming conventions(punctuation, capitalization, spacing, and so on) are used. e.g., "E-mail" vs. "email", "url" vs. "U.R.L.", etc.

Type 2. Synonym. e.g., "Participant" vs. "Attendee", or they have similar meaning in a specific domain, e.g., "contribution" vs. "paper" in the conference organization domain, etc.

Type 3. Word omissions. e.g., "email" vs. "hasEmail", "Regular_author" vs. "author", etc.

Type 4. Abbreviations. e.g., "PC_Member" vs."ProgramCommitteeMember", or acronym, e.g., "WWW" vs. "World Wide Web", etc.

Type 5. Misspelling. e.g., "sponsor" vs. "sponzor", etc.

Type 6. Two names are tokenizable and there tokens (or only a part of tokens) is syntactic or meaning similar. e.g., "hasSurname" vs. "has_the_last_name", "Camera_ready_contribution" vs. "Final_manuscript", etc.

Type 7. Do not belonging to any categories above, but they surely denote the same entity. e.g., "has_a_review_expertise" vs. "hasRating", "Speaker" vs. "Active_conference_participant", etc.

Type 8. Irregular. This can be caused by synthetic replace with random strings, or just use an identifier to denote entities. e.g., "volume" vs. "zsbdgz", etc.

Sometimes even, entities names' variations are composed by several kinds of name variations discussed above. Among these variations, from our experiences, type 1, type 3, type 6 happen first place, type 2, type 4, type 7 come second, and type 5, type 8 at least.

3 String Similarity Metrics

String similarity metrics usually used in ontology alignment systems can be classified into two groups: string-based and language-based. We covered some typical metrics in detail that will be evaluated in Section 4 (show in bold).

3.1 String-based Similarity Metrics

String-based similarity metrics calculate syntactic similarity of two names, they can divide into two groups: character-based metrics measures similarity depending only on the appearance and sequence of characters; while token-based string similarity metric first tokenize two strings into sets of tokens (words), then compute the similarity between two sets.

3.1.1 Character-based

- **Edit distance** The edit distance between two strings is the minimum cost of edit operations need to transform one string into another. Each operation has a cost function associated, in the simplest form, each has cost 1, this is also referred to Levenstein distance. The edit distance can address the typographical errors of name variations very well, and the distance can be transformed to similarity by subtracting normalized distance by 1:

$$sim(s_1, s_2) = 1 - \frac{Levenstein(s_1, s_2)}{\max\{|s_1|, |s_2|\}} \tag{1}$$

- **Monge-Elkan** The Monge-Elkan is a variation of Smith-Waterman with the match matrix scores all entries 0 except that an exact match scores 5, and approximate matches score 3. Two characters are approximate match if they fall into one of the following set: {d t} {g j} {l r} {m n} {b p v} {a e i o u} {, .}. It uses an affine gap model with gap start penalties -5 and gap continual penalties -1. The Monge-Elkan is suit for abbreviations of name variations mostly. More information about this metric can find in [6].

- **Jaro-Winkler** Given two strings $s_1 = a_1...a_N$ and $s_2 = b_1...b_L$, define a character a_i in s_1 to be an common if there exist a character $b_j = a_i$ in s_2, such that $|i-j| < \min(|s1|, |s2|)/2$. Let $s'_1 = a'_1...a'_m$ be the common characters in s_1 (in the same order), and Let $s'_2 = b'_1...b'_m$ be the common characters in s_2, define a transposition of s'_1 and s'_2 to be a position i such that $ai \neq bi$. Let m be number of the common characters and t be half the number of transpositions, The Jaro distance is defined as:

$$Jaro(s_1, s_2) = \begin{cases} 0 & \text{if } m = 0, \\ 3 \cdot \left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{otherwise.} \end{cases} \tag{2}$$

Winkler proposed a variant of this metric that prefer to two strings has common prefix, let l' be the longest common prefix of s_1 and s_2, $l = \max(4, l')$, Jaro-Winkler is defined as:

$$Jaro - Winkler(s_1, s_2) = Jaro(s_1, s_2) + l \cdot p(1 - Jaro(s_1, s_2)) \tag{3}$$

where p is a variable, and $p < 0.25$, usually set to 0.1.

- **ISUB** ISUB [4] is a so-called new metric specially developed for ontology alignment, it argue that similarity is based on commonalities as well as differences between two strings being compared. ISUB is defined as:

$$sim(s_1, s_2) = comm(s_1, s_2) - diff(s_1, s_2) + winkler(s_1, s_2) \tag{4}$$

The $comm(s_1, s_2)$ first finds the longest common substring, then removes it and searches for the next longest common substring repeatedly until there is no one retain. The sum of the lengths of i iterations' substrings is then scaled by the length of original strings:

$$comm(s_1, s_2) = \frac{2 \cdot \sum_i |maxComSubstring_i|}{|s_1| + |s_2|} \tag{5}$$

The $diff(s_1, s_2)$ is defined as:

$$diff(s_1, s_2) = \frac{uLen_{s1} * uLen_{s2}}{p + (1-p) * (uLen_{s1} + uLen_{s2} - uLen_{s1} * uLen_{s2})} \tag{6}$$

where $uLen$ is a function return the length of unmatched substring leaved by the iteration step from the initial strings scaled with the original strings' length. p is a factor, the author argue that 0.6 to be a good choice.

The $winkler(s_1, s_2)$ is the improvement of the result using the method introduced above. It should be noted that this metric return a similarity value range from -1 to 1.

- **N-gram** N-gram first convert each string into a set of n-grams, for example, if the string is "paper" and n=3, the result would be pap, ape, per, an improvement is to append special characters prior to the start of and after the end of string, this would result {##p, #pa, pap, ape, per, er%, r%%}. Then the Dice coefficient will used resulting a similarity:

$$sim(set_1, set_2) = \frac{2 \cdot |set_1 \cap set_2|}{|set_1| + |set_2|} \tag{7}$$

We implemented the improved variation with n equal to 3 in the experiment.

3.1.2 Token-based

- **Jaccard** The Jaccard metric is defined as the intersection of the two sets of words dividing by the union of them, let A and B are the sets of words of s_1 and s_2 respectively, then:

$$Jaccard(s_1, s_2) = \frac{|A \cap B|}{|A \cup B|} \tag{8}$$

- **Level2** Monge and Elkan also proposed a set-based similarity metric in their paper [6]. The SecondString [7] library implements this metric as Level2, so that is why we call it Level2 here. Let sim' be a base similarity metric (e.g., Levenstein, Jaro-Winkler, etc.), and $\{a_1, a_2 ... a_n\}$, $\{b_1, b_2 ... b_m\}$ be the set of words of s_1 and s_2, respectively. Then the similarity is defined as:

$$sim(s_1, s_2) = \frac{1}{|s_1|} \sum_{i=1}^{m} \max_{j=1}^{n} \{sim'(a_i, b_j)\} \tag{9}$$

In order to ensure symmetry, $|s_1|$ must be no more than $|s_2|$. We used the Level2 with Jaro-Winkler in the experiment.

- **Soft TF-IDF** TF-IDF stands for term frequency-inverse document frequency, which was widely used in information retrieval community. Let $TF_{w,s}$ be the number of times word $w \in s$ appears in two ontologies, divided by the total number of words in this ontologies, IDF_w be the inverse of the fraction of names in ontologies that contain word w. So $w \in s$ has weight $V'(w,s) = \log(TF_{w,s}+1) \cdot \log(IDF_w)$. Then each weight in a string is scaled:

$$V(w,s) = V'(w,s) \Big/ \sqrt{\sum_{w' \in s} V'(w',s)} \tag{10}$$

Finally, using the cosine similarity, this metric can be defined as:

$$TFIDF(s_1,s_2) = \sum_{w \in s_1 \cap s_2} V(w,s_1) \cdot V(w,s_2) \tag{11}$$

Soft TF-IDF [7] is a variation of TF-IDF, in which similar tokens (using an base similarity metric) are only measured once in word sets s_1 and s_2, and similar tokens also considered in $s_1 \cap s_2$. Again let sim' be a secondary similarity metric, then the similarity is defined as:

$$sim(\mathrm{s}_1,s_2) = \sum\nolimits_{w \in \mathrm{s}_1} V(w,s_1) \cdot V(v,s_2) \cdot \max_{v \in s_2}\{sim'(w,v)\} \tag{12}$$

where $\max_{v \in s_2}\{sim'(w,v)\}$ finds the maximal similarity word (which must larger than a threshold θ) $v \in s_2$ with w. We used the Soft TF-IDF with Levenstein 0.9 in the experiment.

3.2 Language-based Similarity Metrics

We only discuss the semantic similarity based on WordNet, for a comprehensive survey of WordNet based semantic similarity, please see [8]. One known approach is **Lin** [9]. Lin's conceptual semantic similarity is defined as:

$$sim(c_i,c_j) = \frac{2 \times \log p(lso(c_i,c_j))}{\log p(c_i) + \log p(c_j)} \tag{13}$$

where $lso(c_i,c_j)$ is the lowest super-ordinate of c_i and c_j in the taxonomy, $p(c_i)$ is the probability of node c_i by which are calculated statistics from a large corpus.

It is important to note that, semantic similarity metrics measure the similarity between two concepts rather than two words, it's need to determine the meaning of words before calculate their similarity. Let $s(w_i)$ be the set of concepts of word w_i, there still has an compromise:

$$sim(w_i,w_j) = \max_{c_i \in s(w_i), c_j \in s(w_j)}\{sim(c_i,c_j)\} \tag{14}$$

3.3 Hybrid Methods

- **Soft TF-IDF with Lin and Levenstein (SLL)** Different from the soft TF-IDF metric, when calculate the similarity between tokens, it first use the Lin metric, in the case of Lin metric return a value less than a predefined threshold (e.g., 0.8), then use the Levenstein metric. We implemented Soft TF-IDF with Lin and Levenstein 0.9 in the experiment.

- **Max** This method uses the maximal similarity value of two entities computed by some metrics as an aggregated similarity. In the experiment, we used the 9 metrics mentioned in Section 3.1 and 3.2.
- **Sigmoid** This method uses the sigmoid function to emphasize high individual similarity values and deemphasizes low individual similarity values. The aggregated similarity is the average of sigmoid individual values of different similarities. We used the function $f(x) = 1/(1 + \exp(-5(x - 0.5)))$ and again the 9 metrics in the experiment.

4 Experimental Evaluation

4.1 Experimental Setup

To compare these metrics' performance for ontology alignment tasks, we conducted experiments that implement with java using the Alignment API [10], the SecondString library [7] and the JWS (Java WordNet Similarity) library. The experiment methodology works as follows. First each entities' lexicons is extracted from the two ontologies, the lexicons are entities' id (in the case of id is null, we use the label instead), and we preprocess them by tokenizing for token-based string similarity metrics. Then for each string similarity algorithm in the library, we compute a similarity score for each pair of entities that belongs to two ontologies respectively. We extract mapping results using the naive descendant extraction algorithm [11]. The experiments were conducted on Intel Core i5 M480 2.67 GHz processors and 6 GB RAM under Windows 7.

In the experiment, part of the benchmarks tests (#204, #301, #302, #303 and #304) and the conference tests from OAEI ontology alignment campaign [12] are used. The #204 test case is a naming conventions modified ontology of the reference ontology, and the #30x are four real-life ontologies, these five ontologies are compared to #101. The conference data set include 16 real world ontologies describing the domain of organizing conferences and 21 tests which reference alignments are open.We follow the evaluation criteria from OAEI campaign in terms of precision, recall and F-measure.

4.2 Experimental Result

For each string similarity metric, after the mapping results are extracted, a filter threshold value range from 0.1 to 1 in interval of 0.1 is used to select the candidate mappings, and at each level of this threshold the average F-measure among all tests was computed. So we can get the highest F-measure at a certain threshold, this best F-measure and corresponding precision as well as recall will be compared with each other. Meanwhile, the exact matching is used as the baseline, the results on two data sets are shown in Fig. 1 and Fig. 2 (in order to save space, we use Levens, MongeE, JaroW instead). The average computation time of two tests each metric spends are illustrated in Fig. 3 and Fig. 4.

Fig. 1 and Fig. 2 reveal a wide disparity among the performance of string similarity metrics. On benchmarks data set, N-gram, ISUB, SoftTFIDF, Jaccard, and Sigmoid have the top performance in term of F-measure, and Lin get the worst since its recall is quite low. On conference data set, the SoftTFIDF, Sigmoid, and Jaccard get the top F-measure, and to our surprised, Max, Monge-Elkan have the worst F-measure. One possible reason is Max and Monge-Elkan return higher

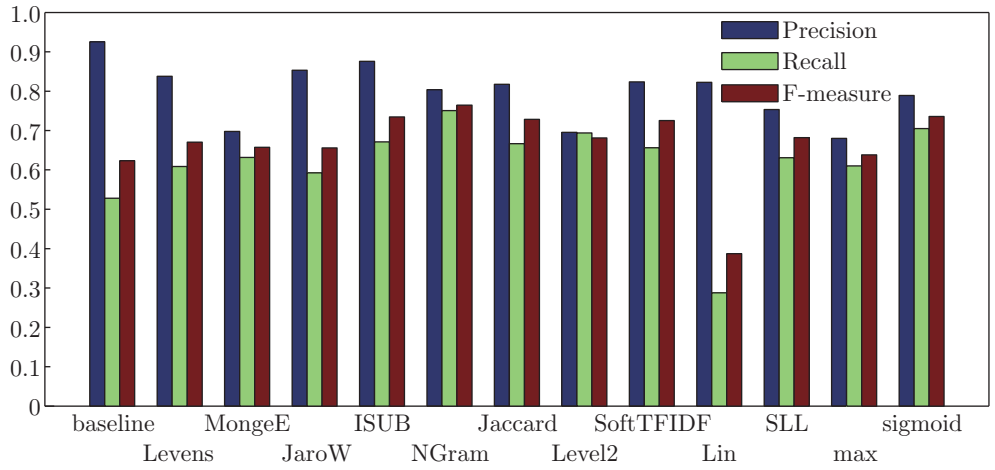

Fig. 1: Performance of string similarity metrics on benchmarks data set

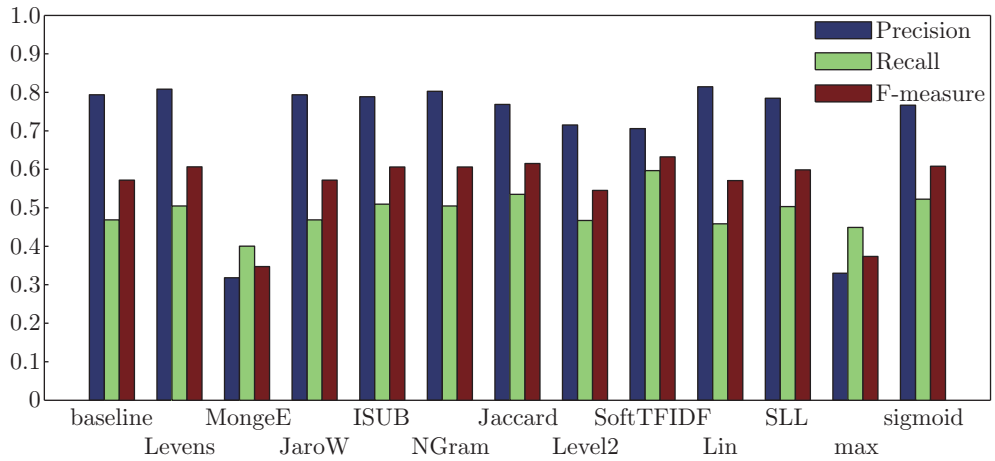

Fig. 2: Performance of string similarity metrics on conference data set

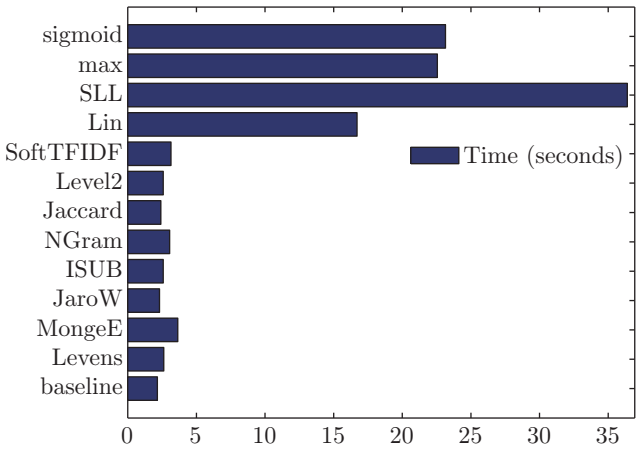

Fig. 3: Computation time on benchmarks data set

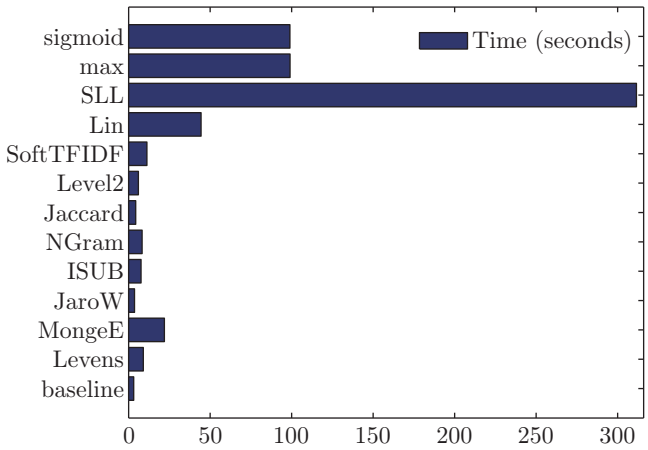

Fig. 4: Computation time on conference data set

similarity to false positive. While if we take precision and recall into account, on benchmarks data set, N-gram has the best recall and baseline get the best precision since it return correspondences as correct by exact match of strings. And on conference data set, Soft-TFIDF get the best recall

and Lin get the best precision. It's clear that hybrid metrics usually used do not improve the performance much actually. Meanwhile, as we expected, illustrated by Fig. 3 and 4, the hybrid methods spend too much time compared to non-hybrid methods. Among non-hybrid methods, since Lin need access the whole WordNet database, it spend more computation time.

We can inspire from the results that, when faced with an ontology alignment task, some information encoding in ontologies and strings can be analyzed to select or combine the appropriate metrics and strategies taking the precision, recall or time cost into account. For example, we should not use the hybrid metrics if the efficiency is important. If the tokens of every entities' string are less than 2, we should choose the character-based metrics, or, if two ontologies share many synonyms, we should select semantic similarity metrics.

5 Conclusion

In order to analyze the string similarity metrics' performance for ontology alignment tasks, in this work, we first discussed the name variations in heterogeneous ontologies, then introduced a wide range of metrics addressing different source of name variations. Finally we conducted experiments to evaluate the performance of these metrics. From the experimental results, some inspiration on how to select or combine these metrics are discussed. We need to compare these metrics' performance on a wider variety of data sets. Those will be our future work.

References

[1] P. Shvaiko, J. Euzenat, Ontology matching: State of the art and future challenges [J], IEEE Transactions on Knowledge and Data Engineering, 25(1), 2013, 158-176

[2] J. Li, J. Tang, Y. Li et al., RiMOM: A dynamic multistrategy ontology alignment framework [J], IEEE Transactions on Knowledge and Data Engineering, 21(8), 2009, 1218-1232

[3] W. Hu, Y. Qu, G. Cheng, Matching large ontologies: A divide-and-conquer approach [J], Data and Knowledge Engineering, 67(1), 2008, 140-160

[4] G. Stoilos, G. Stamou, S. Kollias, A string metric for ontology alignment [C], The Semantic WebCSWC 2005, Berlin: Springer Heidelberg, 2005, 624-637

[5] D. H. Ngo, Z. Bellahsene, R. Coletta, YAM++–Results for OAEI 2011 [C], The 6th International Workshop on Ontology Matching, ISWC'11, 2011, 228-235

[6] A. E. Monge, C. Elkan, The field matching problem: Algorithms and Applications [C], KDD, 1996, 267-270

[7] W. Cohen, P. Ravikumar, S. Fienberg, A comparison of string metrics for matching names and records [C], KDD Workshop on Data Cleaning and Object Consolidation, 3, 2003, 73-78

[8] A. Budanitsky, G. Hirst, Evaluating wordnet-based measures of lexical semantic relatedness [J], Computational Linguistics, 32(1), 2006, 13-47

[9] D. Lin, An information-theoretic definition of similarity [C], Proceedings of the International Conference on Machine Learning, Madison, USA: Morgan Kaufmann, 1998, 296-304

[10] J. David, J. Euzenat, F. Scharffe et al., The alignment api 4.0 [J], Semantic Web, 2(1), 2011, 3-10

[11] C. Meilicke, H. Stuckenschmidt, Analyzing mapping extraction approaches [C], OM, 2007

[12] B. C. Grau, Z. Dragisic, K. Eckert et al., Results of the ontology alignment evaluation initiative 2013 [C], Proc. 8th ISWC Workshop on Ontology Matching (OM), 2013, 61-100