Evaluating the Stability and Credibility of Ontology Matching Methods

Xing Niu[1], Haofen Wang[1], Gang Wu[2], Guilin Qi[3], and Yong Yu[1]

[1] Shanghai Jiao Tong University
{xingniu,whfcarter,yyu}@apex.sjtu.edu.cn
[2] Northeastern University
wugang@ise.neu.edu.cn
[3] Southeast University
gqi@seu.edu.cn

Abstract. Ontology matching is one of the key research topics in Semantic Web. In the last few years, many matching methods have been proposed to generate matches between different ontologies either automatically or semi-automatically. To select appropriate ones, users need some measures to judge whether a method can achieve the similar compliance even on one dataset without reference matches and whether such a method is reliable w.r.t. its output result along with the confidence. However, widely-used traditional measures like *precision* and *recall* fail to provide sufficient hints. In this paper, we design two novel evaluation measures to evaluate **stability** of matching methods and one measure to evaluate **credibility** of matching confidence values, which help answer the above two questions. Additionally, we carry out comparisons among several carefully selected methods systematically using our new measures. Besides, we report some interesting findings such as identifying potential defects of our subjects.

1 Introduction

An ontology provides a common vocabulary, which can be used to model a domain. It is a formal representation of shared knowledge between human beings and machines. Nevertheless, due to the openness of Semantic Web, the widely adoption of ontologies also exposes the heterogeneity problem to the public. Ontology matching is one of the positive efforts to reduce heterogeneity. It aims at finding matches between semantically related entities of different ontologies[7]. In recent years, many ontology matching methods and systems have been developed such as Lily[15], ASMOV[11], Anchor-Flood[9] (aflood for short), RiMOM[12], Falcon-AO[10] and AgreementMaker[2] (AgrMaker for short). With so many matching methods in hand, users need some criteria to evaluate them and select appropriate ones for their applications.

Users usually choose match candidates above a *confidence threshold* (CT). The threshold is adjusted on some training datasets with reference matches on hand. Then the trained CT is applied on test data. To achieve good matching

G. Antoniou et al. (Eds.): ESWC 2011, Part I, LNCS 6643, pp. 275–289, 2011.

performance, data characteristic or distribution of the test data is assumed to be similar to that of the training set. However, it does not always hold. Additionally, it is labor-intensive and error-prone to obtain comprehensive reference matches especially between large ontologies. In some cases, we even do not have authorizations to access the whole ontologies to get those matches. Thus, a widely-used feasible way is to train CT on small or partial datasets with representative selected samples. When trying to select the most appropriate method or system according to the user's application domain, we can only predict the matching quality of those methods working on ontologies without any reference matches. Hence, we introduce a new concept called **stability** for matching assessment. Generally speaking, a matching method with high stability indicates that it performs consistently on the data of different domains or scales.

On the other hand, a matching method outputs candidate matches sorted by their matching confidence values. We prefer methods which generate true positive matches with high confidence values (ranked high) while return false positive ones with low values (ranked low). This is another important criteria called **credibility** to guide users on ontology matching method selection.

However, existing measures only focus on judging the basic compliance like precision, recall or their extensions like relaxed precision and recall[4], and semantic precision and recall[6]. In this way, these measures fail to assess a matching method based on the two important criteria, which makes users hard to select the most suitable method or system. Some methods[5,13,14] for selecting the best matching strategy take several parameters into account, yet these parameters do not include scores for evaluating stability and credibility.

In this paper, we propose several novel measures according to the matching criteria. The main contributions are as follows:

- We propose the *STD* (STandard Deviation) score to measure matching stability by examining the fluctuation of the original *confidence threshold*s. To give an overall consideration of both the theoretical optimum matching quality of matching methods and their matching quality using the trained CTs in practical, we extend the classical *F-measure* to a comprehensive version.
- A *ROC* (Receiver Operating Characteristics) graph indicates the tradeoff between benefits (true positive matches) and costs (false positive matches) [8]. We use the *ROC-AUC* (Area Under Curve) score to measure the credibility by taking the tradeoff into account.
- We further carry out comprehensive experiments to compare several carefully selected methods based on the new measures mentioned above. By observing exceptions, e.g., the score of a matching method is well below the average, we can tell potential weak points a matching method has on particular datasets.

The rest of this paper is organized as follows. Section 2 presents the formal representation of matches and describes some typical ontology matching methods. In Section 3, we recap some basic measures as well as introduce the new measures. In Section 4, we show experimental results using the new introduced measures and provide deep analysis. Finally, we make a conclusion in Section 5.

2 Preliminaries

We will introduce some typical ontology matching methods in a nutshell. They are outstanding participants of OAEI (Ontology Alignment Evaluation Initiative) campaigns[1] and subjects of our evaluation experiments.

Lily[15] Lily realizes three main matching components: GOM (Generic Ontology Matching) uses semantic subgraph technique to combine lexical and structural matching; LOM (Large-scale Ontology Matching) is adopted to match large size ontologies without using ontology modularization; SOM (Semantic Ontology Matching) uses the Web knowledge to recognize the semantic relations through the search engine.

ASMOV[11] ASMOV incorporates a semantic validation process (remove any incorrect or invalid matches) and calculates four partial similarities: sim_L measures lexical similarity referring WordNet; sim_I and sim_E measure internal and external structure similarity respectively; sim_N measures individual one.

Anchor-Flood[9] Aflood aims at achieving high performance and resolving the scalability problem. This algorithm starts off with a pair of concepts from each ontology, gradually exploring concepts by collecting neighboring concepts. This system has two parts: one is the ontology schema matching algorithm that aligns concepts and properties; the other is the instance matching approach.

RiMOM[12] RiMOM includes several lexical matching strategies: Edit-Distance, Vector-Distance and referring to WordNet. The structural matching uses an adaptive variation of the similarity flooding. A strategy selection process is applied to improve the match accuracy. Similarity combination and propagation are also key steps.

Falcon-AO[10] Falcon-AO consists of four matchers: I-Sub and V-Doc (Virtual Documents) are two light-weight linguistic matchers which take neighboring information into consideration; GMO (Graph Matching for Ontologies) is a structural matcher, uses RDF bipartite graphs to represent ontologies and computes structural similarities; PBM (Partition-Based Block Matching) divides large-scale ontologies into blocks and finds block matches between them.

AgreementMaker[2] AgrMaker adapts three string-based techniques: BSM (Base Similarity Matcher), PSM (Parametric String-based Matcher) and VMM (Vector-based Multi-word Matcher), VMM is similar to V-Doc. A graph-based structural matcher is used which is based on the idea that if two nodes are matched with a high similarity, then their children should be similar. It also refers to WordNet to find further matches as its last step.

3 Evaluation Measures

In this section, we propose some new measures for evaluating matching methods. We first introduce three well-known measures: *Precision*, *Recall* and *F-measure*.

[1] `http://oaei.ontologymatching.org/`

Let the result set of ontology matching be divided into subset TP (true positives, means correctly proposed matches) and subset FP (false positives, means falsely proposed matches). Let FN (false negatives) be the set of missing correct matches, then it comes the definitions of $Precision$ and $Recall$:

$$Precision = |TP|/(|TP| + |FP|) \tag{1}$$

$$Recall = |TP|/(|TP| + |FN|) \tag{2}$$

$Precision$ reflects the share of correct matches among all found ones, while $Recall$ reflects the share of correctly proposed matches among all expected ones. In order to avoid the imperfection in evaluating the compliance with $Precision$ or $Recall$ alone, *F-measure*, the harmonic mean of $Precision$ and $Recall$ is proposed and widely used:

$$F\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

3.1 Comprehensive *F-measure*

A confidence value is included in the match computed from matching methods to describe the confidence about the result. The higher a confidence value is, the more similarities two entities share, and the more correct a match is likely to be. Intuitively speaking, matches with low confidence value are likely to be incorrect and will lower the $Precision$ values. However, denying all matches with low confidence values to increase $Precision$ values will lose a certain number of correct ones, which, on the contrary, will decrease the $Recall$ values.

Therefore, in practice, application systems have to determine a *confidence threshold* (CT) value to preserve only those matches with confidence values above it in a matching task. In this case, *F-measure* is a function of CT. The greedy principle of selecting the CT is to have *F-measure* reaches its maximum [7]:

$$maxF\text{-}measure = \max F\text{-}measure(CT) \tag{4}$$

$$maxFCT = \arg\max_{CT} F\text{-}measure(CT) \tag{5}$$

The *maxF-measure* reflects the theoretical optimal matching quality of matching methods but it does not consider their matching quality in practice. The design of a matching method usually takes several semantic information into account, such as Lexical knowledge, Structural knowledge, Domain knowledge, and Instance-based knowledge [1], which are quite dataset specifically (as shown in our subsequent experiments). Thus, application systems have to adjust and tune the CT dramatically to keep a matching method applicable with a high *F-measure* across different datasets. This is costly especially facing large-scale real-world datasets, and even infeasible if there are no enough reference answers for certain datasets in hand. In this case, those matching methods that generate relative stable *maxFCT*s will surpass the others, because they can ensure to

give a holistic applicable *maxFCT* value from only some selected ones among all datasets. Since validation datasets are scarce, we propose a novel measure, i.e. *uniF-measure* (*uniform F-measure*), to simulate the practical application and evaluate such stability of matching methods.

The *uniF-measure* is computed as follows. First, we compute *maxFCT*s from a set of selected datasets (named a **test unit**), and then calculate their arithmetic mean average(*maxFCT*s). Further, the *F-measure* for each corresponding dataset is re-computed under this mean value, which is then defined as the *uniF-measure*:

$$uniF\text{-}measure = F\text{-}measure(\text{average}(maxFCTs)) \tag{6}$$

The test unit mentioned above usually means a set of similar datasets describing the same domain and sharing many resemblances, but differing on details. A test unit contains several matching tasks and reference answers are necessary. It is hired to simulate featured subsets of a particular ontology.

If a matching method is stable, *maxFCT*s of a test unit should be relatively stable, i.e. *uniF-measure* value should not be much lower than *maxF-measure* value. In this way, we can observe the stability of matching methods by contrasting their *maxF-measure* values and *uniF-measure* values, or just calculating their arithmetic mean, the *comF-measure* (*comprehensive F-measure*) value. *comF-measure* is an extended *F-measure* and defined by

$$comF\text{-}measure = (maxF\text{-}measure + uniF\text{-}measure)/2. \tag{7}$$

3.2 *STD* Score

As described in Section 3.1, for some reason (such as lacking enough reference answers, or in a predicting task), there is only a limited part of the datasets (so-called the test unit) available beforehand. So we select a matching method with high stability on a limited dataset and hope this method can perform well in the coming matching tasks.

Measuring the dispersion of *maxFCT*s in a test unit can help us estimate the difficulty in obtaining *maxF-measure* for a given matching method. However, sometimes it is too strict to just focus on *maxFCT*s. In practice, we also accept those *CT* values that cause the test unit to have *F-measure* values close to *maxF-measure*. Here we use a real example as shown in Figure 1 to further explain this condition. When $CT = 0.848$, *F-measure* reaches its maximum 0.946. Actually, some smaller values near this *maxF-measure* are fully acceptable too, e.g., *F-measure* $= 0.939$ when $CT = 0.878$.

Thus, we give some grace when finally determine the expecting *CT* measuring. From all *CT*s that can get top 20 percents of *F-measure*, we choose the maximum *CT* as the *relaxedCT*. The *relaxedCT* can be formally defined similar to *maxFCT*:

$$relaxedCT = \max(\arg\operatorname*{top20pct}_{CT}(F\text{-}measure(CT))) \tag{8}$$

The dotted box in Figure 1 encloses top 20 percents of *F-measure* and we have $relaxedCT = 0.878$ now. The value of *relaxedCT* will be too far apart

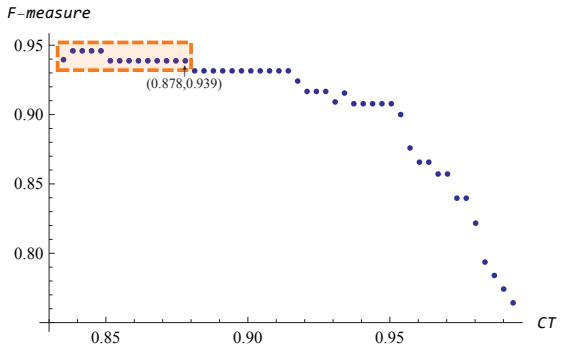


Fig. 1. Relation between *F-measure* and *CT* (result for Benchmark#304 by Falcon-AO)

from *maxFCT* if the range of *F-measure* varies widely in some cases. So we should restrict the range of *relaxedCT* under this condition. Here we choose the top 10 percents of *F-measure* if the variation range exceeds 0.1. If the variation range still exceeds 0.1 under the circumstances, only those *F-measure*s that are between *maxF-measure* and *maxF-measure*−0.1 will be taken into account.

With the definition of *relaxedCT*, we propose the *STD score* as another stability measure.

$$STD\,score = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(relaxedCT_i - \overline{relaxedCT}\right)^2} \tag{9}$$

where N is the number of matching tasks in a test unit, and $\overline{relaxedCT}$ is the mean value of *relaxedCT*s.

The *STD* score is a standard deviation to measure the dispersion of *relaxedCT*s, and hence that of *maxFCT*s. According to the probability theory and statistics, slight diversity of *relaxedCT* will cause the deviation of *STD* score apparently. Therefore, *STD* score is effective in measuring the stability of matching methods. *STD* scores are comparable when they are calculated under the same test unit.

3.3 *ROC-AUC* Score

A reasonable matching method should have a certain degree of credibility in confidence value, because the credible confidence value plays an important role in ranking matching results reasonably. With such credibility, the greater confidence value a matching method gives, the relevant match is more likely to be correct. We hire the Receiver Operating Characteristics (*ROC*) analysis technique [8] to measure such credibility.

The *ROC* analysis is used for 'visualizing, organizing and selecting classifiers based on their performance' [8]. As shown in Figure 2, *ROC* graphs are two-dimensional polygonal line graphs in which true positives rate is plotted on the Y axis, and false positives rate is plotted on the X axis.

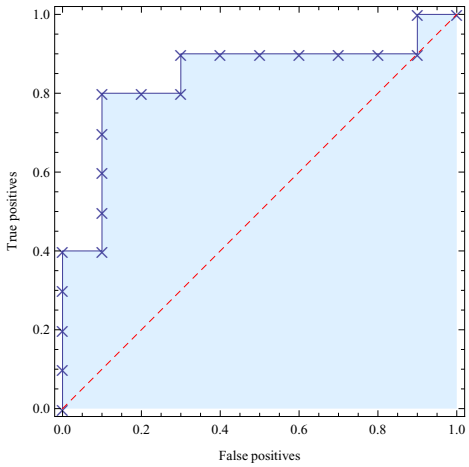


Fig. 2. An Example of *ROC* Graph

To measure the credibility of confidence value with *ROC* graph, we should first sort matches by their confidence values before drawing the *ROC* curves. Starting from the match with the highest confidence value, if it is true positive, the curve climbs up by one unit distance ($1/|TP|$); otherwise, the curve move horizontally to the right by $1/|FP|$. The whole polygonal line starts from origin and ends up at (1,1). The *ROC-AUC* (Area Under Curve) is the area surrounded by *ROC* curve, X axis and line $X = 1$. Since the domains of axes are normalized, the score of this kind of measure could be defined easily as:

$$ROC\text{-}AUC\ score = ROC\text{-}AUC \tag{10}$$

ROC-AUC score is applicable for measuring the credibility of confidence value because if confidence values are really credible, the *ROC* curve should climb straight towards (0,1) in the early stage of drawing the curve, and *ROC-AUC* should be relatively large, i.e. *ROC-AUC* score should be high. The meaning of *ROC-AUC* score could be explained by referring to an illustration from the realm of information retrieval that the users always want to receive most correct answers with high confidence value (cf. ranking score).

Ehrig [3] does not recommend this measure for the reason that different dataset results cannot be compared directly because the unit distance we use for drawing the curve is dependent on the result size. In fact, we do not directly compare matching methods using this measure, but qualitatively analyze their potential problems.

4 Experimental Results

OAEI is widely used for ontology matching evaluation. It not only assesses the strength and weakness of matching methods but also helps improving the work

on ontology matching. We choose OAEI 2009 Benchmark track and the Conference track as test datasets because relatively complete reference matches of these tracks are available for judging. In the following subsections, we will introduce main characteristics of each dataset and present experimental results using the above described evaluation measures.

4.1 Testing on the Benchmark Track

The OAEI setups benchmark test library to offer a set of tasks which are 'wide in feature coverage, progressive and stable'[2]. These tasks describe bibliographic reference.

Most ontologies of the Benchmark track orient from a complete reference ontology and there are six categories of alteration. All the benchmark tasks can be divided into five groups and we treat each group as a test unit (tasks in a test unit share many resemblances):

#101-#104 (10X for short) The descriptions of classes or properties may have some differences.

#201-#210 (20X for short) Local names and comments information are reduced while structural information remains the same.

#221-#247 (22X for short) Hierarchy, instances, properties and classes information are reduced while lexical information remains the same.

#248-#266 (24X for short) Lexical information is suppressed and structural information is reduced. We abandon this test unit because we do not think information of real-world ontologies is so deficient.

#301-#304 (30X for short) Four real-world ontologies.

The synthetic benchmark is instrumental in evaluating matching methods. Here we measure a simple matching method and five sophisticated ones. They are StringDistance, Falcon-AO, Lily, ASMOV, RiMOM and aflood respectively. More precisely, StringDistance simply compares differences between strings. We treat it as our baseline. The five sophisticated matching methods are selected due to their outstanding and stable performance in Benchmark track of OAEI campaigns in recent years (2005-2009).

10X Test Unit. This test unit is relative simple, so we provide all three kinds of scores in Table 1. In order to visually compare these scores of different matching methods, we also present results with Spider Charts (Figure 3). In Spider Charts, we can not only capture disparities of scores shown in three axes, but also learn overall quality by surveying the area surrounded by score lines. We start axes of Spider Charts at 0.5 for enlarging the disparities.

The Spider Chart for 10X test unit (Figure 3a) shows that all matching methods get high scores in *comF-measure* and $ROC\text{-}AUC$ measures except StringDistance. However, *STD* scores of Lily and ASMOV are lower than that of others.

[2] `http://oaei.ontologymatching.org/2009/benchmarks/`

Table 1. Results for Benchmark Test Units

Tools	#	*comF-measure*	*STD* score	*ROC-AUC*	#	*comF-measure*	*STD* score	*ROC-AUC*
Falcon-AO	10X	0.996537	0.999240	1.000000	22X	0.989778	0.949460	0.996392
Lily		0.993873	0.837004	1.000000		0.966045	0.767347	0.986057
ASMOV		0.991161	0.832834	1.000000		0.795300	0.742211	0.988999
RiMOM		0.965817	0.996060	1.000000		0.986734	0.947575	0.988127
aflood		1.000000	1.000000	1.000000		0.991277	1.000000	0.986153
StringDistance		0.779116	1.000000	0.638158		0.818852	1.000000	0.698579
Falcon-AO	20X	0.897292	0.633577	0.955449	30X	0.806695	0.950244	0.885652
Lily		0.973254	0.914770	0.993886		0.807346	0.836017	0.825668
ASMOV		0.940700	0.836917	0.989130		0.747131	0.762499	0.794369
RiMOM		0.822705	0.472834	0.993197		0.813504	0.892603	0.804618
aflood		0.925260	1.000000	0.973064		0.830853	1.000000	0.900738
StringDistance		0.544872	0.878835	0.439786		0.802029	0.912243	0.548727

Table 2. Ontology Information of Benchmark-20X Test Unit

Datasets	Names	Comments	Datasets	Names	Comments
203	0	N	207	F	0
204	C	0	208	C	N
205	S	0	209	S	N
206	F	F			

We draw Figure 4 which reflects the fluctuations of *relaxedCT* values, so as for further analysis on the *STD* scores. More precisely, we project each task into one unique point at X axis: 10X test unit (including 3 tasks) corresponds to point 1 to point 3, 20X test unit corresponds to 4 to 10, etc. Each horizontal dashed line in red running through a set of points represents the mean value of the corresponding *relaxedCT*s. Intuitively, the more discrete points are, the lower *STD* score will be. We do not present aflood on the figure because *relaxedCT*s of aflood are always 1s. In fact, aflood does not provide distinguishing confidence values but 1 for any match.

20X Test Unit. 20X test unit is a little more complex, and detailed ontology information is described in Table 2. Here we explain the meaning of abbreviations: 0 (stay unchanged), N (suppressed), F (strings in another language than English), C (naming conventions) and S (synonyms). We abandon tasks 201, 202 and 210 due to the same reason of abandoning 24X test unit.

We present *maxF-measure* and *uniF-measure* results (Figure 5) for this test unit because some exceptions are detected. Performances of Falcon-AO (Figure 5a), ASMOV (Figure 5c) and RiMOM (Figure 5d) draw our attention. All of them have clear disparities between *uniF-measure* scores and *maxF-measure* scores in #209. To find out the reason, we can come back to figure 4 (#209 corresponds to 10 at X axis). These three matching methods should set *CT*s really low to get *maxF-measure* scores. That is to say, when meet synonyms, these methods do not have much confidence in matching. Comparing with StringDistance we

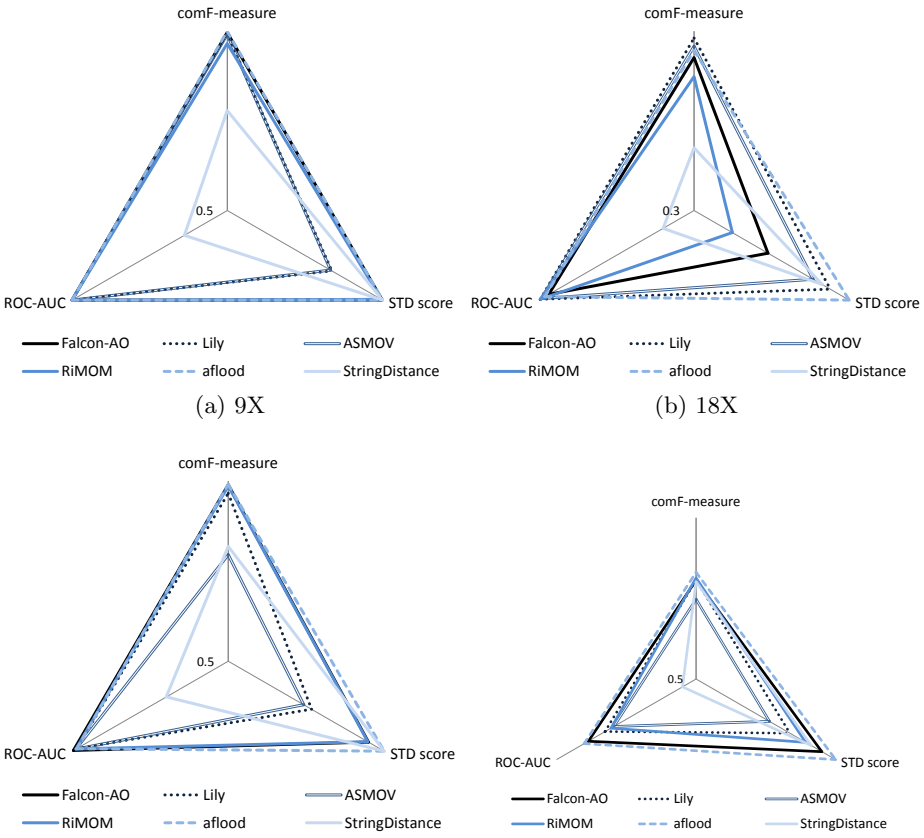


Fig. 3. Spider Charts for Benchmark Test Units

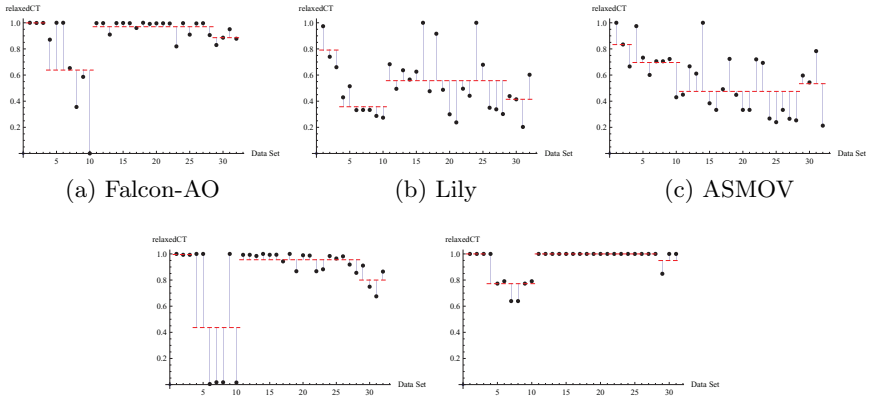
can infer that they must use structural information to get decent *maxF-measure* scores. Lily and aflood are not facing the instability problem although they also meet the challenge of lacking for lexical information.

We also observe that RiMOM does not work well in #205, #206 and #207 at the same time. Together with #209, we can find that RiMOM does not perform stable enough when Local Name information is suppressed a lot.

All evaluation results of this test unit are presented in Figure 3b. The overall matching quality can be summarized easily according to comparing areas surrounded by score lines.

22X Test Unit. The detailed ontology information of 22X test unit is described in Table 3. Meanings of abbreviations are: 0 (stay unchanged), N (suppressed), F (flattened), E (expanded) and R (random strings).

In this test unit, Lily and ASMOV get some exceptions and their *maxF-measure* and *uniF-measure* scores are presented in Figure 6.



(a) Falcon-AO (b) Lily (c) ASMOV

(d) RiMOM (e) StringDistance

Fig. 4. The Fluctuations of *relaxedCT* in Benchmark Test Units

Table 3. Ontology Information of Benchmark-22X Test Unit

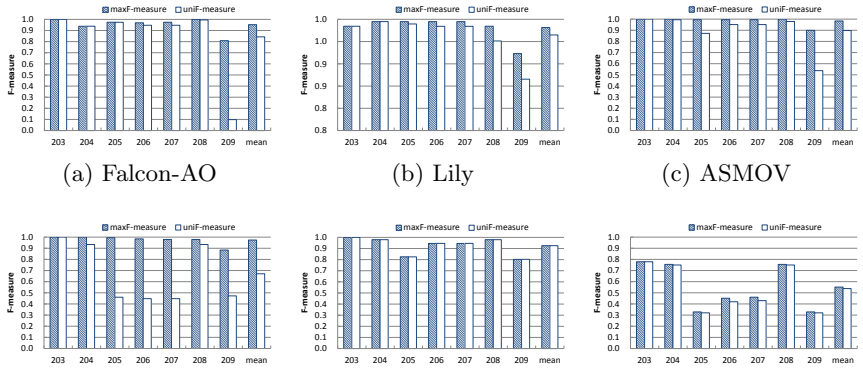
Datasets	Hierarchy	Instances	Properties	Classes	Datasets	Hierarchy	Instances	Properties	Classes
221	N	0	0	0	233	N	0	N	0
222	F	0	0	0	236	0	N	N	0
223	E	0	0	0	237	F	N	0	0
224	0	N	0	0	238	E	N	0	0
225	0	0	R	0	239	F	0	N	0
228	0	0	N	0	240	E	0	N	0
230	0	0	0	F	241	N	N	N	0
231	0	0	0	E	246	F	N	N	0
232	N	N	0	0	247	E	N	N	0

Referring to Figure 4b, Figure 4c and ontology features described in Table 3, we can find out potential flaws of these matching methods: Lily is hyper-sensitive to instances information while ASMOV is hyper-sensitive to properties information. Hyper-sensitivity means a matching method makes great different in final results (confidence values of matches) for whether or not a certain kind of information is missing. Hyper-sensitivity concededly damages stability of matching methods.

30X Test Unit. This test unit contains four real-world ontologies. According to the explanation of OAEI, the reference matches for these tasks are not perfect[3], so we will not care matching quality which is represented by *comF-measure*. Actually, *maxF-measure* scores of all matching methods, including StringDistance, are very close.

StringDistance does not provide result for #303 because this ontology lacks in local names of classes and properties and StringDistance fails to extract this information from complete URIs. Lily also faces this problem, but it considers

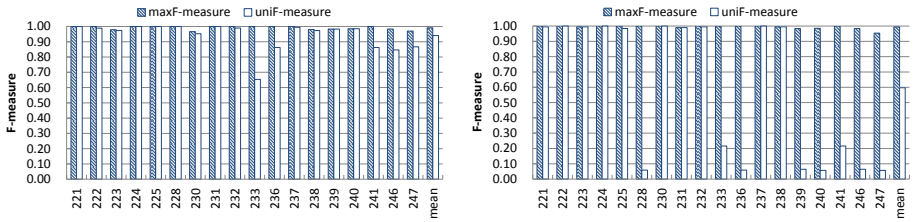
[3] `http://oaei.ontologymatching.org/2009/benchmarks/#266`



(a) Falcon-AO (b) Lily (c) ASMOV

(d) RiMOM (e) aflood (f) StringDistance

Fig. 5. *maxF-measure* and *uniF-measure* Results for Benchmark-20X Test Unit



(a) Lily (b) ASMOV

Fig. 6. Partial *maxF-measure* and *uniF-measure* Results for B-22X Test Unit

comments, so matches are found after all. Real-world ontologies usually have some special features that matching methods should take into account. Besides missing local names mentioned above, #301 also has a unique characteristic that all local names of properties start with 'has', which is a typical naming convention.

The Spider Char of 30X test unit (Figure 3d) shows that the sophisticated matching methods have no advantages over the simple StringDistance method except when we measure *ROC-AUC* scores.

4.2 Testing on the Conference Track

The ontologies of Conference track come from conference organization domain. All of these ontologies are built by different groups and have 'heterogeneous character of origin'[4]. We choose seven ontologies whose reference matches are offered. The target of this track is to match every two ontologies so there are $C_7^2 = 21$ tasks and we group all these tasks into a single test unit. A simple matching

[4] `http://oaei.ontologymatching.org/2009/conference/`

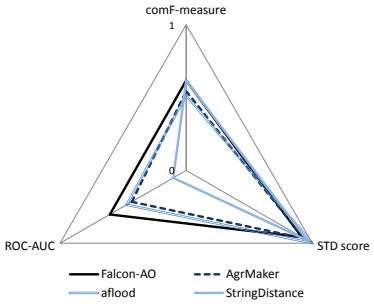


Fig. 7. Spider Chart for Conference Test Unit

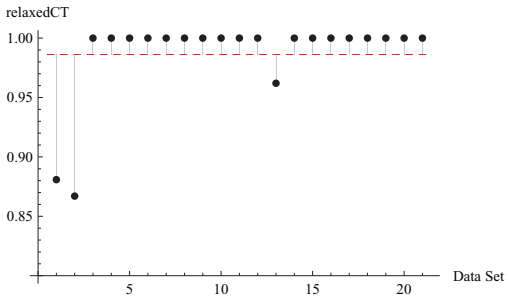


Fig. 8. The Fluctuation of *relaxedCT* in Conference Test Unit (StringDistance)

method (StringDistance) as baseline and three sophisticated ones (Falcon-AO, AgrMaker and aflood) are measured for Conference test unit. Falcon-AO performs well and stably in Conference track of OAEI 2006&2007 while AgrMaker and aflood perform better than other participants in Conference track of OAEI 2008&2009.

From the visual representation of evaluation results (Figure 7), we observe that all *STD* scores are satisfactory while all *comF-measure* scores are not that good. We even notice that simple StringDistance can nearly take the place of the sophisticated ones. However, *ROC-AUC* score holds it back. Figure 8 exhibits that most *relaxedCT*s of StringDistance are 1s, in other words, all confidence values of matches are 1s. That means it is useless to sort matches by confidence values, so users can hardly distinguish more likely correct matches among them.

Aflood faces the same problem since it always sets confidence values as 1s. This method gets normal *ROC-AUC* scores in Benchmark track so we infer that it generates confidence values for sorting during processing but does not output them in the end. Anyhow, it confuses users eventually and does not continue getting high *ROC-AUC* score in this track.

4.3 Discussion

Single *F-measure* score or even *comF-measure* score may mask potential problems of matching methods. As the detailed experimental analysis shows above, differences are manifested on *STD* scores in Benchmark track and *ROC-AUC* scores in Conference track.

Hyper-sensitivity is detected in our experiments by *STD* scores, e.g., Falcon-AO and ASMOV are hyper-sensitive to synonyms, RiMOM is hyper-sensitive to local names and properties information and Lily is hyper-sensitive to instances information. To promote stability of matching methods, researchers should pay more attention to the information that matching methods are hyper-sensitive to.

Users have the authorities to determine final confidence threshold, so matching methods ought to provide credible and diverse confidence values for different

matching tasks. Indifference confidence value is a simple cause of low *ROC-AUC* scores, yet more internal factors of matching methods remain to be dug and improved.

5 Conclusions and Future Work

Ontology Matching is one of the most popular research fields in Semantic Web. In recent years, many matching methods have been proposed. In order to assess the matching performance of these methods, it is essential to have a comprehensive benchmark which can find the potential weakness of these methods and help researchers to improve them to a certain extent.

In this paper, we design three new evaluation measures to evaluate stability of matching methods and credibility of their matching confidence values. Moreover, we identify potential defects of subjects by detecting the exception of these measure scores. The deep analysis can shed light on the selection of appropriate matching systems against the specific domain and environment. It may also help pointing out the way to improve a given matching method.

In the future, we intend to extend our evaluation measures to a comprehensive strategy for selecting matching methods and compare this strategy with existing ones. We also plan to test more matching methods under other datasets in OAEI using our proposed measures. As a result, we would like to make both stability and credibility as standard evaluation measures for ontology matching.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 60903010, the Natural Science Foundation of Jiangsu Province under Grant No. BK2009268, and the Key Laboratory of Advanced Information Science and Network Technology of Beijing under Grant No. XDXX1011.

References

1. Bouquet, P., Serafini, L., Zanobini, S.: Semantic coordination: A new approach and an application. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 130–145. Springer, Heidelberg (2003)
2. Cruz, I.F., Antonelli, F.P., Stroe, C., Keles, U.C., Maduko, A.: Using agreementmaker to align ontologies for oaei 2009: Overview, results, and outlook. In: Proceedings of the 4th International Workshop on Ontology Matching (2009)
3. Ehrig, M.: Ontology Alignment: Bridging the Semantic Gap, Semantic Web And Beyond Computing for Human Experience, vol. 4. Springer, Heidelberg (2007)
4. Ehrig, M., Euzenat, J.: Relaxed precision and recall for ontology matching. In: Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies (2005)
5. Ehrig, M., Staab, S., Sure, Y.: Bootstrapping ontology alignment methods with APFEL. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 186–200. Springer, Heidelberg (2005)

6. Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 348–353 (2007)
7. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
8. Fawcett, T.: An introduction to roc analysis. Pattern Recognition Letters 27(8), 861–874 (2006)
9. Hanif, M.S., Aono, M.: An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. J. Web Sem. 7(4), 344–356 (2009)
10. Hu, W., Qu, Y.: Falcon-ao: A practical ontology matching system. Web Semantics: Science, Services and Agents on the World Wide Web 6(3), 237–239 (2008)
11. Jean-Mary, Y., Shironoshita, E., Kabuka, M.: Ontology alignment with semantic validation. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 235–251 (2009)
12. Li, J., Tang, J., Li, Y., Luo, Q.: Rimom: A dynamic multistrategy ontology alignment framework. IEEE Trans. Knowl. Data Eng. 21(8), 1218–1232 (2009)
13. Mochol, M., Jentzsch, A., Euzenat, J.: Applying an analytic method for matching approach selection. In: Proceedings of the 1st International Workshop on Ontology Matching (2006)
14. Tan, H., Lambrix, P.: A method for recommending ontology alignment strategies. In: Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference, pp. 494–507 (2007)
15. Wang, P., Xu, B.: Lily: Ontology alignment results for oaei 2009. In: Proceedings of the 4th International Workshop on Ontology Matching (2009)