Towards Evaluating Interactive Ontology Matching Tools

Heiko Paulheim[1], Sven Hertling[2], and Dominique Ritze[1]

[1] University of Mannheim, Germany
Research Group Data and Web Science
{heiko,dominique}@informatik.uni-mannheim.de
[2] Technische Universität Darmstadt, Germany
Knowledge Engineering Group
hertling@ke.tu-darmstadt.de

Abstract. With a growing number of ontologies used in the semantic web, agents can fully make sense of different datasets only if correspondences between those ontologies are known. Ontology matching tools have been proposed to find such correspondences. While the current research focus is mainly on fully automatic matching tools, some approaches have been proposed that involve the user in the matching process. However, there are currently no benchmarks and test methods to compare such tools. In this paper, we introduce a number of quality measures for interactive ontology matching tools, and we discuss means to automatically run benchmark tests for such tools. To demonstrate how those evaluation can be designed, we show examples on assessing the quality of interactive matching tools which involve the user in matcher selection and matcher parametrization.

1 Introduction

Ontologies are used for describing information in the semantic web as well as for assigning meaning to data. Until now, there has been no commonly agreed upon a universal ontology, and it is unlikely that such an ontology will ever exist. On the contrary, there is a wide spectrum of ontologies used in the semantic web. For example, in the Linked Open Data cloud, more than half of the 295 datasets use their own ontologies.[1]

To use information from those ontologies in a reasonable way, ontology alignments, i.e., links between those ontologies, are necessary. Ontology matching tools are capable of finding such alignments. In the past, research on ontology matching tools has been focused on developing fully automatic ontology matching tools to a large extent.

Performing ontology matching fully automatically in high quality is hard. For many real-world datasets, fully automatic state of the art tools still yield results at a quality level that is unsatisfying for many use cases. At the recent ontology alignment evaluation initiative (OAEI),[2] the best fully automatic ontology matching tools have yielded a result quality[3] of about 70% for single-language and 40% for multi-lingual matching tasks from the conference domain [1].

[1] http://wifo5-03.informatik.uni-mannheim.de/lodcloud/state/

[2] http://oaei.ontologymatching.org/

[3] In terms of F-measure, i.e., the harmonic mean of recall and precision

Our hypothesis implies that there is an upper bound to the quality of the alignment which is hard to exceed by fully automatic ontology matching tools. As stated by Falconer and Noy, ontology matching is “a very challenging problem for both man and machine” [11], which calls for semi-automatic approaches combining the strengths of automatic matching algorithms and the expertise of domain experts in the matching process. On the other hand, domain experts are a scarce and expensive resource. This makes it necessary to a) design tools that draw maximum benefits from as little user interaction as possible, b) define suitable evaluation measures that capture those benefits and interactions, as well as the trade-off between them, and c) provide automatic evaluation approaches for such tools.

Incorporating user interaction in ontology matching tools is still a major challenge in ontology matching today [29]. Furthermore, unlike the OAEI benchmarks that measure the quality of fully automatic ontology matching tools in terms of recall, precision, and F-measure, there are no commonly agreed upon quality measures for interactive ontology matching tools, let alone comparative evaluations [11].

This paper introduces a number of quality measures for interactive ontology matching tools, inspired by a similar field of research in the area of machine learning, i.e., *active learning* [26]. Furthermore, we discuss how that quality can be measured in fully automatic test settings.These test settings will form the basis of a new branch of tests in future OAEI campaigns.

The rest of this paper is structured as follows. In Section 2, we discuss previous approaches to interactive ontology matching and evaluations. Section 3 introduces a general framework for describing interactive ontology matching tools. Based on that framework, we discuss a number of measures in Section 4. To illustrate how interactive ontology matching tools can be evaluated, we show two experiments for interactive matching in Section 5, that deal with matcher selection and parameterization. We conclude with a summary and an outlook on future work.

2 Related Work

Interactive ontology matching is closely related to active learning, a problem class in machine learning where the algorithm actively presents instances to label to a user [26]. The idea behind active learning is that a learning algorithm can minimize the workload to label examples if it is able to choose examples that are “interesting” for learning, e.g., borderline cases. Active learning has been successfully applied to other fields, related to ontology matching. Isele et al. discuss an active learning approach to generate linkage rules for the Web of Data [15], de Freitas et al. provide a method for detecting data duplication in databases [5] and Rodler et al. use it for ontology debugging [27].

In general, there are several possibilities how and at which point of time to involve the user in the matching process. This can be either before, during or after the matching process. Common examples to improve the matching by involving the user include defining system configurations, creating anchor mappings, correcting suggested correspondences, or evaluating the created alignment.

Several matching systems provide a configuration which can be adapted by the user according to the actual matching task. Since defining a configuration is a difficult task for domain experts, some approaches ask users for example mappings or validation of generated correspondences instead [24, 28].

Most ontology matching systems combine different matching strategies. Some of the systems implementing machine learning for selecting strategies and fully automatically learn how to best combine the matching methods, e.g. *GLUE* [7]. Other approaches, e.g. [8], even combine several different matching systems. Beside automatic combination, some systems involve the user by asking for validation of created correspondences [6, 9, 17] or request an initial list of correspondences and non-correspondences [33]. Such approaches are capable of outperforming conventional systems (in terms of quality), but it is not clear how the additional effort of the training or even user interaction pays off and to which amount, nor how to measure that trade-off between user efforts and improvement in alignment quality.

Besides the configuration and combination of matching systems, users can help to improve the alignment, or the systems can support the users to generate an alignment. The improvement can either be done a posteriori or (interactively) during the process.

One idea to perform an improvement after the matching is to let the user rate the correspondences such that other users can take advantage of this rating [23]. This strategy is independent of the applied matching system, however, cannot help to improve future mappings since the rating is not fed back into the matching systems. Approaches such as *PROMPT* perform the alignment generation interactively, ask the user for feedback, indicate conflicts [21], and/or provide a proper visualization to support the decision [3]. Moreover, they usually try to reduce the number of user interactions.Whenever a domain expert is asked to manually create mappings, several tools can be taken into account to support the process, e.g., by showing partial results when certain matching rules are applied [2, 20] or to point the user to places where attention is required [19].

Since collaborative strategies are getting very popular, some matching systems even apply concepts like crowd sourcing [25] or gamification [31] to generate ontology alignments. Obviously, these approaches require a lot of user interactions.

The proposed systems clearly differ in the kind and amount of user involvement. Most of the corresponding papers provide some evaluation, but they are rarely comparable since they often apply different data sets or focus on various measures, e.g. quality, runtime, and/or amount of user interaction.

Evaluation of ontology matching tools, such as the ontology alignment evaluation initiative (OAEI), have focused on non-interactive aspects of the systems so far, and do not take any user interaction into account. In contrast, Lambrix and Edberg [18] performed an evaluation to compare tools with user involvement. They compared two interactive matching systems with respect to availability, stability, representation language, functionality, assistance, precision and recall of the mapping suggestions and time, and also evaluated user satisfaction with a questionnaire. However, an evaluation campaign involving many tools and including a certain amount of users would result in an enormous effort and is thus hardly feasible.

Falconer and Storey [12] proposed a theoretical framework with several principles and corresponding software requirements for ontology matching systems. They introduce several functionalities each a tool should provide. These functionalities are concerned with user analysis and decision making, interaction, analysis and generation as well as representation. Their work only shows a theoretical framework but does not provide any benchmarks or evaluation techniques.

So far, several methods for interactive ontology matching have been proposed. However, as stated in [11], "evaluation of such [semi-automatic ontology matching] tools is still very much in its infancy." With this paper, we aim at closing that gap by providing a set of measures and a toolkit for fully automatic evaluation of interactive ontology matching tools.

3 Generic Framework

Non-interactive ontology matching tools do not provide any point of interaction between their invocation and the delivery of the final alignment. In the standard model introduced by Euzenat and Shvaiko [10], ontology matching tools take two ontologies and an (optional) alignment as input, and, optionally based on some parameters and external resources, deliver a final alignment. We extend that model to include interaction with a user.

To describe those interactions, we use the following convention: an alignment is a set of triples, also called correspondences

$$< o_1\#e_1, o_2\#e_2, r >, \tag{1}$$

where $o_1\#e_1$ is an element from ontology one, $o_2\#e_2$ is an element from ontology two, and r is a relation that holds between the two, such as equality or subsumption. While most ontology matching tools also deliver a confidence score for each triple, we consider that confidence score as meta-information on the alignment rather than part of the alignment itself. Furthermore, we only consider *simple* mappings that relate single elements, and not *complex* mappings, which might call for different forms of interaction. Examples for possible interactions include, but are not limited to:

Asking for validation of a candidate alignment. The tool provides a mapping element $< o_1\#e_1, o_2\#e_2, r >$ and asks the user if that mapping is correct or not.

Asking for definition of the relation in a candidate alignment. The tool provides a mapping element $< o_1\#e_1, o_2\#e_2, X >$ and asks the user for filling in the variable X for the relation that holds between e_1 and e_2, e.g. *broader than* or *equivalent to*.

Asking for completion of an element in a candidate alignment. The tool provides a mapping element with a variable, e.g., $< o_1\#e_1, X, r >$ and asks the user to fill in the variable X, if there is a sensible substitution.

Figure 1 depicts our generic framework. The tool may issue a hypothesized partial alignment (which may contain variables, as discussed above) to a domain expert, and asks for carrying out a certain action, such as validation or completion. The domain expert returns a correct partial alignment (which may be empty, in case a hypothesized alignment is completely discarded by the user). After a certain number of interactions (which may be after a fixed a number of interactions or when the tool has derived an alignment it considers to be stable), the tool delivers a final alignment.

Note that the list of interactions above is not fixed, and that the interactions may go beyond purely validating the system's output. For example, users may also be asked for

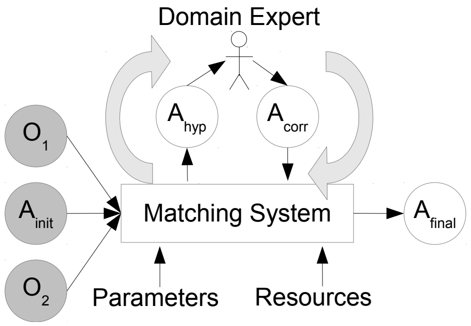

Fig. 1. A generic framework for interactive ontology matching, as an extension of [10] and [11]. Unlike fully automatic matching systems, interactive ontology matching systems support (an arbitrary number of) interactions between the system and a domain expert, triggered by the matching system. Typically, the ontology matching system will provide a hypothesized, partial alignment, possibly containing variables, which is validated and completed by the domain expert.

arbitrary example mappings (using the third type of interaction with two variables), or for the confirmation of completeness of a partial alignment.

4 Evaluation Measures for Interactive Ontology Matching Tools

Non-interactive ontology matching tools are evaluated using recall, precision, and in particular F-measure [10]. Interactive ontology matching tools require different measures which take into account the achieved result quality (i.e., the F-measure), as well as the economic use of the domain expert's workload, e.g., the amount of mappings a user has to validate.

To compute a domain expert's workload, we assign costs c_i to each action a_i performed by the domain expert. This is a *generic cost*, which can be filled by different *actual measures* when implementing our model, such as the time consumed, the money paid to an expert, or the money spent on a crowdsourcing platform. With those values, we can compare two matchers both by the F-measure they achieve, as well as by the cost of interaction they have caused.

For an *automatic evaluation*, it is often necessary to further *simplify the cost model*. For example, an automatic evaluation scenario may only allow one type of interaction. In that case, it is possible to assign a constant weight (e.g., 1) to each interaction. In the case that different interactions are allowed, the weights have to be fixed in a more sophisticated manner (e.g., letting users perform sample tasks of each type, and computing average times for those tasks). In scenarios where different interactions are possible, those may impose different cognitive loads on the domain expert (e.g., confirming or discarding an element is less demanding than completing an incomplete mapping). Thus, those different loads have to be reflected in different costs, which of course are only an approximation of the actual costs.

While it is easy to optimize F-measure (let the domain expert do everything) and cost (do not involve the domain expert at all) on their own, achieving a reasonable

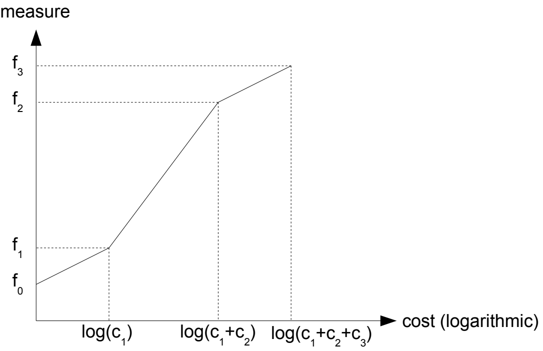

Fig. 2. An example learning curve for an ontology matcher. After each interaction step, the F-measure of the preliminary alignment is graphed against the cost consumed so far. The cost axis uses a logarithmic scale to reward fast convergence towards an optimal F-measure.

trade-off between the two is more challenging. Furthermore, depending on the actual interactive ontology matching algorithm, it may be difficult to determine when to stop, as better results may still be achieved through more user involvement. In order to account for those differences, we demand that each matching tool is capable of delivering a *preliminary alignment* A_{prelim} at any given point in time. This alignment represents the *best alignment found so far*.[4] Having preliminary alignments allows for plotting a *learning curve* of F-measure relative to the cost consumed, as shown in Fig. 2.

As discussed above, interactive ontology matching follows a similar task setting as active learning. Active learning tools are often evaluated by drawing a graph depicting the quality of the learning algorithm (e.g., its ROC value) plotted against the number of examples presented to the user. The normalized area under that curve (referred to as AUL, the area under the learning curve) is then used as a measure for comparing active learning tools [13]. A high AUL is achieved if a tool reaches a high overall F-measure and converges towards that value with few user interactions. To reward quick convergence (and, hence, efficient use of the domain expert's workload), the cost axis uses a logarithmic scale.

Given that the F-measure after the i-th user interaction is measured as f_i using the preliminary alignment A_{prelim}, and the cost of the i-th interaction is measured as c_i, the normalized AUL, using a logarithmic scale for the cost axis, can be computed as

$$AUL = \frac{1}{log \sum\limits_{i=1}^{n} c_i} \sum_{i=1}^{n} \left(log \sum_{k=1}^{i} c_k - log \sum_{k=1}^{i-1} c_k \right) \frac{f_i + f_{i-1}}{2} \tag{2}$$

In that case, f_0 is the F-measure that the tool achieves without any user interaction, such as a default initial mapping determined with a simple string-based similarity measure. When using the AUL measure for comparing two matchers performing user interactions at a different overall cost, the final F-measure of the matcher consuming

[4] Tools that do not maintain any intermediate results may simply return an empty alignment.

lower cost is used for the remaining interactions steps of the matcher consuming higher cost, so that the AUL values can be compared in a meaningful manner. Usually, the largest cost consumed is normalized to 1, so that AUL is a value between 0 and 1.

Next to AUL, the maximum F-measure value f_{max} reached during the interactive matching process, as well as the final F-measure value f_{final} are of interest. For an ideal interactive ontology matching tool, the learning curve is monotonously increasing, thus, f_{max} = f_{final} holds in that case. However, real interactive ontology matching tools will probably not always expose that behavior.

Besides new quality measures, interactive ontology matching introduces a new baseline as well. After each interaction, the human expert may have generated a partial mapping. For example, after correcting n mappings, there are is a partial mapping up to size n. The F-measure achieved with that mapping, depicted as f_{human}, serves a baseline for interactive matching tools: a matching tool that makes use of a number of interactions with a domain expert should provide a better alignment than that created by the domain expert alone.

For automatically evaluating interactive matching systems, we use the architecture depicted in Fig. 3.[5] An evaluation system which holds the reference alignment creates an oracle that answers the queries posed by the matching system. It informs the evaluation system whenever the oracle is called. The evaluation system can then ask the matching system for a preliminary alignment, as discussed above, whenever the oracle is called, in order to plot the learning curve and compute the final AUL value once the matching system has returned its final alignment.

5 Experiments

To illustrate how interactive ontology matching tools are evaluated, as well as providing some reasonable use cases for interactive ontology matching, we have conducted two experiments: interactively selecting a matcher for a given problem, and interactively tuning a matcher's parameters. These experiments use existing matchers and results from previous OAEI challenges and an illustration of how to implement our evaluation framework, rather than an in-depth study in matcher combination and parameterization, as the results strongly depend on the matchers and data used for the experiments.

5.1 Experiment 1: Matcher Selection

Not all ontology matching tools perform equally well on each ontology matching task. Thus, automatically selecting a matching tool for a given pair of ontologies is desirable to allow for optimal matching results. However, selecting a matching tool is difficult for an end user, who may be a domain expert, but not an expert for ontology matching tools [30].

As a possible solution, matchers can be selected indirectly by a domain expert in an interactive matching setting. By letting users rate individual mappings generated by

[5] An implementation of that framework, based on the Alignment API [4], is available from `http://www.ke.tu-darmstadt.de/resources/ontology-matching/interactivematching/`

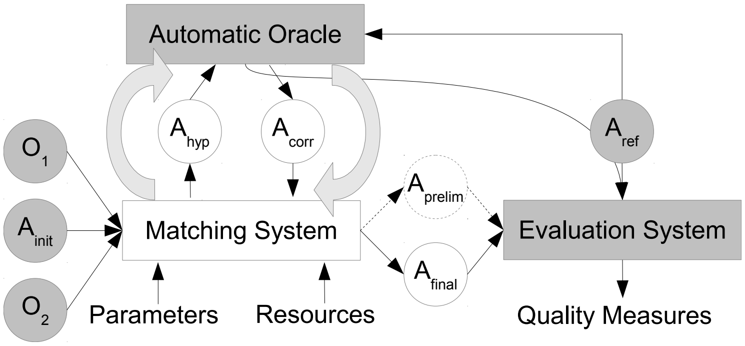

Fig. 3. Framework for evaluating interactive ontology matching tools. In order to facilitate automatic evaluation, the domain expert in Fig. 1 is replaced by an automatic oracle. The oracle is observed by the evaluation system in order to track the cost consumed. The evaluation measures are calculated based on the cost consumed and the quality of the preliminary (i.e., the best alignment found so far at a given point in time) and final alignments generated by the matching tool.

different matching tools, we can select the matching tool that receives the best ratings for a given pair of ontologies. In that case, we assume that the performance on a rated partial alignment correlates with the performance on the complete alignment, which is a valid assumption, as shown in [24].

For this experiment, we have used the dataset from the OAEI 2012 conference track.[6] The public part of this dataset consists of seven ontologies and pairwise reference alignments, resulting in a total of 21 matching problems. In order to test the matcher selection, we have used three of the currently best performing matchers on this dataset: CODI [14], LogMap [16], and Optima [32].[7]

As shown in Table 1, they all have their strengths and weaknesses: Each tool is superior on a number of individual problems (CODI: four, LogMap: seven, Optima: seven, tied: three). This shows that not every matching problem is best addressed by the same strategy: While CODI reformulates the matching problem as an optimization problem, Optima is a combination of "classic" matching techniques, such as string similarities and structural measures, and LogMap uses reasoning for computing a mapping. LogMap, the best tool among the three, achieves an average F-measure of .69, however, if we were able to always select the best tool, that selection would yield an average F-measure of .73 (and hence even beat the best tool in the competition, YAM++, which reaches an overall F-measure of .71).

The algorithm for selecting matchers is designed as follows: each matcher is run on the dataset. From the results, we collect all mappings that are found by at least one, but not by all matchers. Those mappings are put in a list in random order, and presented to the user for validation. We use one basic kind of interaction, i.e., validating if a

[6] `http://oaei.ontologymatching.org/2012/conference/`

[7] We did not take the best performing tool YAM++ into account, because it was the best tool in the majority of all cases, which makes it uninteresting for demonstrating matcher selection.

Table 1. Setup and results of the matcher selection experiment. The table depicts the individual results of the matchers included in the experiment, with the best results marked in bold. For both selection strategies, the final and the maximum F-measure, the total cost of interactions and the AUL value are depicted.

	Matcher			Selection by F-measure				Selection by scoring			
Data	CODI	LogMap	Optima	f_{final}	f_{max}	cost	AUL	f_{final}	f_{max}	cost	AUL
cmt-conf	0.56	0.54	**0.63**	0.63	0.63	14	0.59	0.63	0.63	14	0.61
cmt-confOf	0.56	0.45	**0.61**	0.61	0.61	7	0.56	0.61	0.61	7	0.55
cmt-edas	0.73	**0.76**	0.67	0.73	0.76	8	0.71	0.67	0.76	8	0.70
cmt-ekaw	**0.67**	**0.67**	0.5	0.67	0.67	11	0.67	0.67	0.67	11	0.67
cmt-iasted	**0.89**	**0.89**	0.72	0.89	0.89	3	0.89	0.89	0.89	3	0.89
cmt-sigkdd	0.75	**0.91**	0.87	0.91	0.91	7	0.91	0.91	0.91	7	0.91
conf-confOf	0.67	0.76	**0.80**	0.80	0.80	20	0.80	0.80	0.80	20	0.69
conf-edas	0.60	**0.73**	0.63	0.73	0.73	18	0.70	0.63	0.73	18	0.67
conf-ekaw	0.5	**0.56**	0.45	0.56	0.56	17	0.56	0.56	0.56	17	0.56
conf-iasted	0.4	**0.61**	0.44	0.61	0.61	17	0.61	0.61	0.61	17	0.61
conf-sigkdd	0.71	0.71	**0.77**	0.77	0.77	11	0.73	0.77	0.77	11	0.74
confOf-edas	0.51	0.67	**0.69**	0.69	0.69	13	0.67	0.69	0.69	13	0.68
confOf-ekaw	0.74	0.80	**0.83**	0.83	0.83	10	0.81	0.83	0.83	10	0.82
confOf-iasted	**0.67**	0.62	0.64	0.67	0.67	11	0.64	0.64	0.67	11	0.64
confOf-sigkdd	**0.92**	0.73	0.88	0.92	0.92	6	0.86	0.88	0.92	6	0.83
edas-ekaw	0.62	0.58	**0.64**	0.64	0.64	12	0.63	0.64	0.64	12	0.61
edas-iasted	**0.58**	0.52	0.54	0.58	0.58	13	0.55	0.54	0.58	13	0.54
edas-sigkdd	0.61	**0.64**	**0.64**	0.64	0.64	2	0.64	0.64	0.64	2	0.64
ekaw-iasted	0.67	**0.70**	0.54	0.70	0.70	15	0.70	0.70	0.70	15	0.70
ekaw-sigkdd	**0.86**	0.74	0.80	0.86	0.86	9	0.83	0.80	0.86	9	0.81
iasted-sigkdd	0.75	**0.85**	0.63	0.85	0.85	16	0.83	0.85	0.85	16	0.85
average	0.67	**0.69**	0.66	0.73	0.73	11.4	0.71	0.71	0.73	11.4	0.70

candidate mapping element is correct or not. Based on the user's response, a score for each matcher is computed. We use two different variants for scoring matchers:

F-measure Based on all true positives and false positives gathered from the user so far, we compute a partial F-measure, as described in [24].

Scoring For a true positive, all matchers that have found the element increase their score by 1, for a false positive, all matchers that have not found the element increase their score by 1.

For both variants, in case of ties, the tool with the higher a priori quality (i.e., with the better overall performance in OAEI 2012) is returned (in that case, a random selection would also be possible). Since scores are attributed after each interaction step, we can always return a preliminary alignment (i.e., the one produced by the matcher which is currently the best).

The results are shown in Fig. 4 and Table 1. It can be observed that the baseline (i.e., only using the matcher which is best on average) is constantly exceeded by both approaches after only two interactions. The value for f_{human} is always worse than the

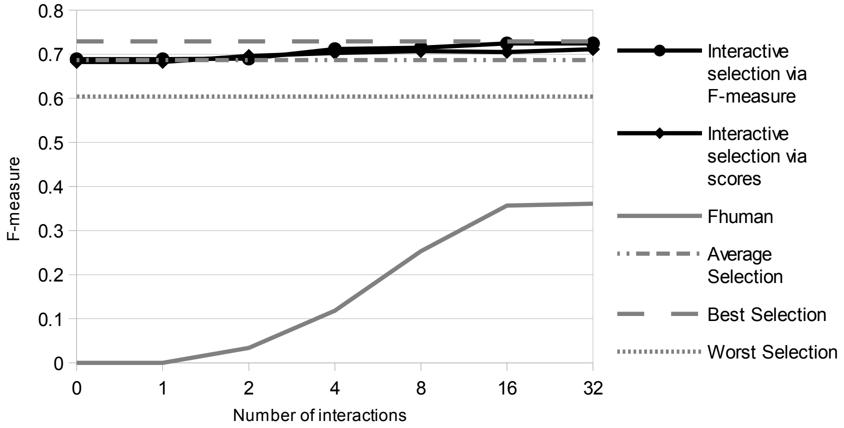

Fig. 4. Resulting learning curves for the matcher selection experiment on the OAEI conference dataset, using three matchers. The best results are achieved by selection via F-Measure. After four interactions, the default selection is outperformed, and the algorithm quickly approaches the theoretical best selection.

baseline, which shows that the approach actually makes significant use of the information gathered from the domain expert.

When comparing both variants, it can be observed that selection by F-measure is slightly superior to selection by scores, as it is less likely to diverge again once it has found an optimal matcher, which can be seen by comparing f_{max} to f_{final}. For selection based on F-measure, the best possible selection is achieved in all but one case (after a maximum of 14 interactions), i.e., $f_{max} \approx f_{final}$; the selection based on scores misses the best possible selection in six out of 21 cases., i.e., $f_{max} > f_{final}$. In both cases, f_{max} equals the best possible selection, i.e., the best matcher is found at least once in the process, but only the approach using selection by F-measure actually sticks to the best selection.

To analyze how the approach can deal with a larger number of matchers, we have repeated the experiment above with all matchers participating in OAEI 2012, again using the conference dataset. The results are depicted in Fig. 5. Here, the superiority of selection via F-measure over selection via scores is even more strongly visible: only the selection via F-measure converges towards the optimum selection, while selection via scores is not capable of outperforming the average selection baseline.

The reason why selection via scores is not optimal is that each true negative (among the false positives found by any other matcher) and each true positive equally increase the score. This can lead to skewed results in some situations. For example, a “defensive” matcher with low recall and high precision can become over-rated in the presence of a matcher producing a large number of false positives. Such effects are avoided using selection via F-measure, which provides a more accurate approximation to the final F-measure achieved in the selection process.

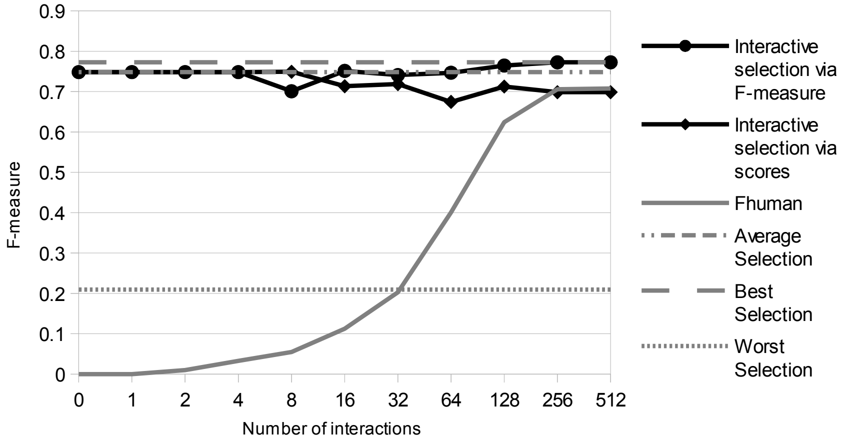

Fig. 5. Resulting learning curves for the matcher selection experiment with all matchers on the OAEI conference dataset. Only selection via F-Measure is capable of finding the best possible selection, but the number of user interactions required is fairly large.

5.2 Experiment 2: Matcher Parametrization

Like selecting a matcher that performs optimally on a dataset, setting good parameters for that matcher is a task that is hard to perform for a person who does not know about the internals of that matcher (it may even be hard for the developer of the matcher, since some parameters are hard to determine without experimentation) [24, 30]. In our second experiment, we try to let users determine parameters indirectly via rating candidate alignments instead of direct parameter manipulation.

In this experiment, we use the matching tool WeSeE [22], which, like many tools, requires a parameter for the cutoff threshold above which mappings are returned. As shown in Table 2 for the conference dataset, always selecting an optimal threshold would yield an F-measure of 0.70, while the best global threshold only yields 0.65.

We determine the threshold to select by presenting mappings to the user and collecting the feedback. The presented mappings are selected by using a search window of size w containing mappings around a given threshold.

To find the threshold, we use the following algorithm: starting with thresholds of 0, 0.5, and 1, we initially pick each w mapping element which has a confidence score as close as possible to those values. After rating the $3 \cdot w$ elements, the approximate F-measure on each of those thresholds is calculated based on all the elements rated so far, like in experiment 1. For the best threshold value, we divide the intervals left and right of that value in half and select to mapping elements at those split points for the next round of interaction. The algorithm ends if there are no more mappings elements between two split points.

All costs are set as in experiment 1. The results are shown in Fig. 6 and Table 2 for window sizes w of 1, 3, and 5. It can be seen that the baseline is exceeded in all three variants, and that the results are close to the optimum. For window sizes 1 and 3, the search algorithm is sometimes distracted in a wrong direction, thus, the F-measure

Table 2. Setup and results of the matcher parametrization experiment with WeSeE-Match. The table depicts the matcher's results with a default threshold parameter and with the best possible threshold parameter. For the three variants, the final and the maximum F-measure achieved, as well as the total cost of interactions and the AUL value are depicted.

	Threshold		$w = 1$				$w = 3$				$w = 5$			
Data	def.	best	f_{final}	f_{max}	cost	AUL	f_{final}	f_{max}	cost	AUL	f_{final}	f_{max}	cost	AUL
cmt-conf	0.58	0.62	0.58	0.58	11	0.44	0.58	0.58	16	0.45	0.58	0.58	22	0.43
cmt-confOf	0.43	0.48	0.38	0.38	9	0.34	0.48	0.48	14	0.37	0.48	0.48	22	0.37
cmt-edas	0.76	0.76	0.76	0.76	11	0.51	0.76	0.76	16	0.47	0.76	0.76	22	0.41
cmt-ekaw	0.49	0.67	0.67	0.67	9	0.48	0.67	0.67	14	0.45	0.67	0.67	19	0.42
cmt-iasted	0.89	0.89	0.89	0.89	9	0.57	0.89	0.89	17	0.45	0.89	0.89	22	0.36
cmt-sigkdd	0.82	0.92	0.92	0.92	10	0.65	0.92	0.92	19	0.59	0.92	0.92	20	0.55
conf-confOf	0.71	0.71	0.69	0.69	10	0.49	0.71	0.71	16	0.48	0.71	0.71	22	0.53
conf-edas	0.64	0.69	0.67	0.67	11	0.48	0.67	0.67	17	0.45	0.62	0.67	22	0.40
conf-ekaw	0.45	0.54	0.55	0.55	12	0.53	0.46	0.48	17	0.38	0.52	0.55	29	0.50
conf-iasted	0.38	0.42	0.44	0.44	10	0.28	0.44	0.44	18	0.23	0.35	0.35	22	0.21
conf-sigkdd	0.69	0.69	0.67	0.67	11	0.48	0.69	0.69	18	0.47	0.69	0.69	25	0.44
confOf-edas	0.65	0.69	0.69	0.69	12	0.48	0.69	0.69	16	0.47	0.69	0.69	30	0.44
confOf-ekaw	0.78	0.82	0.76	0.76	10	0.61	0.79	0.79	14	0.72	0.82	0.82	21	0.70
confOf-iasted	0.67	0.71	0.67	0.67	10	0.45	0.71	0.71	16	0.43	0.71	0.71	21	0.37
confOf-sigkdd	0.83	0.86	0.83	0.83	9	0.58	0.86	0.86	16	0.54	0.86	0.86	20	0.48
edas-ekaw	0.50	0.59	0.48	0.48	10	0.39	0.51	0.51	18	0.38	0.56	0.56	22	0.39
edas-iasted	0.56	0.63	0.62	0.62	11	0.41	0.62	0.62	22	0.37	0.62	0.62	30	0.32
edas-sigkdd	0.61	0.77	0.77	0.77	12	0.50	0.72	0.72	18	0.43	0.72	0.72	22	0.40
ekaw-iasted	0.67	0.75	0.75	0.75	20	0.48	0.67	0.67	17	0.36	0.75	0.75	22	0.38
ekaw-sigkdd	0.78	0.78	0.78	0.78	10	0.53	0.78	0.78	14	0.52	0.78	0.78	20	0.49
iasted-sigkdd	0.73	0.81	0.85	0.85	10	0.57	0.85	0.85	22	0.51	0.85	0.85	22	0.45
average	0.65	0.70	0.69	0.69	10.8	0.49	0.69	0.69	16.9	0.45	0.69	0.70	22.7	0.43

does not grow monotonously. For a window size of 5, where a lot of mapping elements are presented to the user, the curve does not grow significantly faster than F_{human}, although it starts from a higher level (the F-measure achieved using a default parameter setting). The results do not differ much in terms of the final F-measure that is reached, but larger window sizes consume more interaction cost and take longer to converge to the optimum, as reflected in the cost and AUL values.

6 Conclusion and Outlook

In this paper, we have discussed the problem of interactive ontology matching and its evaluation. We have introduced a number of evaluation measures, which are generic enough to be applied in an evaluation scenario involving end users or crowd sourcing, as well as allow for automatic evaluation.

To support fully automatic testing and comparison of interactive ontology matching tools, we have proposed a framework which can make use of existing benchmark data and emulate user behavior. We have shown how the framework can be applied to address common problems in ontology matching, such as matcher selection and parameter

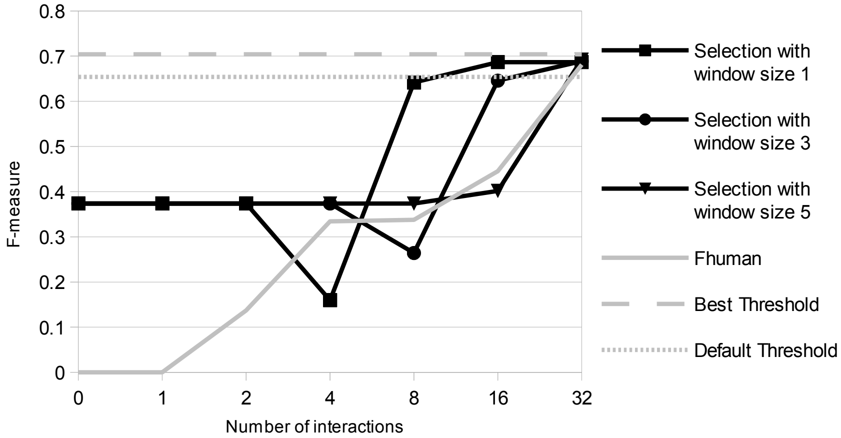

Fig. 6. Resulting learning curves for the matcher parametrization experiment with WeSeE-Match on the OAEI conference dataset. Larger window sizes result in better results, but slower convergences (i.e., more user interaction).

tuning, by interactive techniques, to compare different variants of interactive matchers, and to draw useful insights from the results provided by our evaluation measures.

In this paper, we have introduced a set of example interactions. Those may range from accepting or rejecting a candidate mapping to completing mappings. This set of examples is not an exhaustive list. For the future, we envision a full catalog of possible interactions during ontology matching. In particular, when turning to complex mappings involving more than one element on each side, the set of interactions may encompass more interactions.

Such a catalog of interactions would be even more informative with a throrough evaluation of the costs that typically come with those interactions. These could be obtained, for example, through user studies measuring the average time spent on different types of interactions. A more fine-grained weighting of interactions than simply assigning a weight to a class of interactions may also be beneficial. For example, users might be faster in rejecting many false positives (such as $Researcher \equiv Publication$), while true positives may require a closer look (such as $Researcher \equiv Scientist$).

Measuring user experience of interactive ontology matching tools has been out of scope of our work so far. The reason is that we aim at test procedures that can be fully automatized, which is difficult (if not impossible) for measuring user experience. However, for interactive ontology matching tools providing a user interface, measuring user experience is a useful complement to the measures discussed in this paper.

Based on the work presented in this paper, we are planning to introduce a new track to the next OAEI campaign, which will explicitly focus on interactive ontology matching tools. By supplying a test environment, we hope to gather insights in the qualitative comparison of existing interactive matching approaches, as well as encourage researchers in the community to develop novel approaches and algorithms for interactive ontology matching.

References

1. J. L. Aguirre, K. Eckert, J. Euzenat, A. Ferrara, W. R. van Hage, L. Hollink, C. Meilicke, A. Nikolov, D. Ritze, F. Scharffe, P. Shvaiko, O. Šváb Zamazal, C. Trojahn, E. Jiménez-Ruiz, B. C. Grau, and B. Zapilko. Results of the Ontology Alignment Evaluation Initiative 2012. In *Proc. of the 7th Int. Workshop on Ontology Matching*, 2012.
2. H. Chalupksy. OntoMorph: A Translation System for Symbolic Knowledge. In *Proc. of the 17th Int. Conference on Knowledge Representation and Reasoning*, 2000.
3. I.F. Cruz, C. Stroe, and M. Palmonari. Interactive User Feedback in Ontology Matching Using Signature Vectors. In *Proc. of the 28th Int. Conference on Data Engineering*, pages 1321–1324, 2012.
4. J. David, J. Euzenat, F. Scharffe, and C. Trojahn dos Santos. The Alignment API 4.0. *Semantic Web*, 2(1):3–10, 2011.
5. J. de Freitas, G. L. Pappa, A. S. da Silva, M. A. Gonçalves, E. S. de Moura, A. Veloso, A. H. F. Laender, and M. G. de Carvalho. Active Learning Genetic Programming for Record Deduplication. In *Proc. of IEEE Congress on Evolutionary Computation*, pages 1–8, 2010.
6. S. Duan, A. Fokoue, and . Srinivas. One Size Does Not Fit All: Customizing Ontology Alignment Using User Feedback. In *Proc. of the 9th Int. Semantic Web Conference*, pages 177–192, 2010.
7. Songyun Duan, Achille Fokoue, and Kavitha Srinivas. One size does not fit all: Customizing ontology alignment using user feedback. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *The Semantic Web – ISWC 2010*, volume 6496 of *Lecture Notes in Computer Science*, pages 177–192. Springer Berlin Heidelberg, 2010.
8. Kai Eckert, Christian Meilicke, and Heiner Stuckenschmidt. Improving ontology matching using meta-level learning. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvönen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Simperl, editors, *The Semantic Web: Research and Applications*, volume 5554 of *Lecture Notes in Computer Science*, pages 158–172. Springer Berlin Heidelberg, 2009.
9. Marc Ehrig, Steffen Staab, and York Sure. Bootstrapping ontology alignment methods with apfel. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *The Semantic Web – ISWC 2005*, volume 3729 of *Lecture Notes in Computer Science*, pages 186–200. Springer Berlin Heidelberg, 2005.
10. J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer, 2007.
11. S. M. Falconer and N. F. Noy. Interactive Techniques to Support Ontology Matching. In Z. Bellahsene, A. Bonifati, and E. Rahm, editors, *Schema Matching and Mapping*, pages 29–51. Springer, 2011.
12. Sean M. Falconer and Margaret-Anne Storey. A cognitive support framework for ontology mapping. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, volume 4825 of *Lecture Notes in Computer Science*, pages 114–127. Springer Berlin Heidelberg, 2007.
13. I. Guyon, G. C. Cawley, G. Dror, and V. Lemaire. Results of the Active Learning Challenge. *Journal of Machine Learning Research - Proceedings Track*, 16:19–45, 2011.
14. J. Huber, T. Sztyler, J. Noessner, and C. Meilicke. CODI: Combinatorial Optimization for Data Integration - Results for OAEI 2011. In *Proc. of the 6th Int. Workshop on Ontology Matching*, 2011.
15. Robert Isele, Anja Jentzsch, and Christian Bizer. Active learning of expressive linkage rules for the web of data. In Marco Brambilla, Takehiro Tokuda, and Robert Tolksdorf, editors, *Web Engineering*, volume 7387 of *Lecture Notes in Computer Science*, pages 411–418. Springer Berlin Heidelberg, 2012.

16. E. Jiménez-Ruiz, B. C. Grau, and I. Horrocks. LogMap and LogMapLt Results for OAEI 2012. In *Proc. of the 7th Int. Workshop on Ontology Matching*, 2012.
17. E. Jiménez-Ruiz, B. C. Grau, Y. Zhou, and I. Horrocks. Large-scale interactive ontology matching: Algorithms and implementation. In *Proc. of the 20th European Conference on Artificial Intelligence*, 2012.
18. P. Lambrix and A. Edberg. Evaluation of Ontology Merging Tools in Bioinformatics. In *Proc. of the Pacific Symposium on Biocomputing*, pages 589–600, 2003.
19. D. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Proc. of the 17th Int. Conference on Principles of Knowledge Representation and Reasoning*, 2000.
20. R. Miller, L. Haas, and M. Hernandez. Schema mapping as query discovery. In *Proc. of the 26th Int. Conference On Very Large Databases*, 2000.
21. N. F. Noy and M. A. Musen. The PROMPT suite: interactive tools for ontology merging and mapping. *Int. Journal of Human-Computer Studies*, 59(6):983–1024, 2003.
22. H. Paulheim. WeSeE-Match results for OEAI 2012. In *Proc. of the 7th Int. Workshop on Ontology Matching*, 2012.
23. H. Paulheim, M. Rebstock, and J. Fengel. Context-Sensitive Referencing for Ontology Mapping Disambiguation. In *Proc. of the 2007 Workshop on Context and Ontologies Representation and Reasoning*, pages 47–56, 2007.
24. D. Ritze and H. Paulheim. Towards an automatic parameterization of ontology matching tools based on example mappings. In *Proc. of the 6th Int. Workshop on Ontology Matching*, 2011.
25. Cristina Sarasua, Elena Simperl, and Natalya F. Noy. Crowdmap: Crowdsourcing ontology alignment with microtasks. In Philippe Cudré-Mauroux, Jeff Heflin, Evren Sirin, Tania Tudorache, Jérôme Euzenat, Manfred Hauswirth, JosianeXavier Parreira, Jim Hendler, Guus Schreiber, Abraham Bernstein, and Eva Blomqvist, editors, *The Semantic Web – ISWC 2012*, volume 7649 of *Lecture Notes in Computer Science*, pages 525–541. Springer Berlin Heidelberg, 2012.
26. B. Settles. Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
27. K. Shchekotykhin, G. Friedrich, P. Fleiss, and P.Rodler. Interactive ontology debugging: two query strategies for efficient fault localization. *Web Semantics: Science, Services and Agents on the World Wide Web*, 12(13):88–103, 2012.
28. Feng Shi, Juanzi Li, Jie Tang, Guotong Xie, and Hanyu Li. Actively learning ontology matching via user interaction. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *The Semantic Web – ISWC 2009*, volume 5823 of *Lecture Notes in Computer Science*, pages 585–600. Springer Berlin Heidelberg, 2009.
29. P. Shvaiko and J. Euzenat. Ontology Matching: State of the Art and Future Challenges. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):158–176, 2013.
30. Pavel Shvaiko and Jérôme Euzenat. Ten challenges for ontology matching. In Robert Meersman and Zahir Tari, editors, *On the Move to Meaningful Internet Systems: OTM 2008*, volume 5332 of *Lecture Notes in Computer Science*, pages 1164–1182. Springer Berlin Heidelberg, 2008.
31. K. Siorpaes, S. Thaler, and E. Simperl. SpotTheLink: A Game for Ontology Alignment. In *Proc. 6th Conference for Professional Knowledge Management*, 2011.
32. U. Thayasivam, T. Chaudhari, and P. Doshi. Optima+ Results for OAEI 2012. In *Proc. of the 7th Int. Workshop on Ontology Matching*, 2012.
33. H.-V. To, R. Ichise, and H.-B. Le. An Adaptive Machine Learning Framework with User Interaction for Ontology Matching. In *Proc. of the IJCAI2009 Workshop on Information Integration on the Web*, pages 35–40, 2009.