Undefined 1 (2009) 1–5
IOS Press

Quality-Based Model For Effective and Robust Multi-User Pay-As-You-Go Ontology Matching

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Isabel F. Cruz [a,*], Francesco Loprete [b], Matteo Palmonari [b], Cosmin Stroe [a] and Aynaz Taheri [a]
[a] *University of Illinois at Chicago, USA*
E-mail: {ifc,cstroe1,ataher2}@cs.uic.edu
[b] *Università di Milano-Bicocca, Italy*
E-mail: {f.loprete@campus.unimib.it,matteo.palmonari@disco.unimib.it}

Abstract.
Using a pay-as-you-go strategy, we allow for a community of users to validate mappings obtained by an automatic ontology matching system using consensus for each mapping. The ultimate objectives are *effectiveness*—improving the quality of the obtained alignment (set of mappings) measured in terms of F-measure as a function of the number of user interactions—and *robustness*—making the system as much as possible impervious to user validation errors. Our strategy consisting of two major steps: *candidate mapping selection*, which ranks mappings based on their perceived quality, so as to present first to the users those mappings with lowest quality, and *feedback propagation*, which seeks to validate or invalidate those mappings that are perceived to be "similar" to the mappings already presented to the users. The purpose of these two strategies is twofold: achieve greater improvements earlier and minimize overall user interaction. There are three important features of our approach. The first is that we use a dynamic ranking mechanism to adapt to the new conditions after each user interaction, the second is that we may need to present each mapping for validation more than once—*revalidation*—because of possible user errors, and the third is that we propagate a user's input on a mapping immediately without first achieving consensus for that mapping. We study extensively the effectiveness and robustness of our approach as several of these parameters change, namely the error and revalidation rates, as a function of the number of iterations, to provide conclusive guidelines for the design and implementation of multi-user feedback ontology matching systems.

1. Introduction

The ontology matching problem consists of mapping concepts in a *source* ontology to semantically related concepts in a *target* ontology. The resulting set of mappings is called an *alignment* [9], which is a subset of the set of all possible mappings, which we call the *mapping space*. As ontologies increase in size, automatic matching methods, which we call *matchers*, become necessary. The matching process also requires feedback provided by users: in real-world scenarios, and even in the systematic ontology matching benchmarks of the Ontology Alignment Evaluation Initiative (OAEI), alignments are neither correct nor exhaustive when compared against a *gold standard*, also called *reference alignment*. An important consideration is that domain experts such as those with whom

*Corresponding author. E-mail:ifc@cs.uic.edu. This paper significantly extends the paper titled "Pay-As-You-Go Multi-User Feedback Model for Ontology Matching" accepted to EKAW 2014.

we collaborated in the geospatial domain [6], require the ability to verify the correctness of a subset of the mappings. In this paper we propose a semi-automatic ontology matching approach that supports feedback provided by multiple domain experts to match two ontologies. Our approach first computes an alignment using automatic matching methods and then allows for the domain experts to request a mapping to validate. In the rest of the paper, the term *users* refers to the domain experts, not to casual users often called *workers* in crowdsourcing terminology. The fact that our users are domain experts will influence some of our assumptions.

When a user requests a mapping to validate a feedback loop is triggered. A mapping is selected among the entire set of possible mappings using a *candidate selection strategy* and is presented to the user, who can validate the mapping by labelling it as correct or incorrect. A *feedback propagation* method is used to update the similarity of the validated mapping as well as of other for"similar" mappings, thus saving users' effort. In the last step of the loop, a new alignment is selected that satisfies a desired cardinality. The matching process continues iteratively by selecting new candidate mappings and presenting them to users for validation, with the alignment being updated at every iteration.

When different users are allowed to take part in the interactive matching process, they may disagree upon the label to assign to a mapping [1]. Our approach assumes that mappings labeled as correct (resp. incorrect) by a majority of users are correct (resp. incorrect), thus allowing for mislabeling by users. The candidate selection strategy and the feedback propagation method are designed to maximise the improvement of the alignment while reducing the users' effort. In such a multi-user scenario, they have to be devised also to mitigate the propagation of users' errors and reach consensus about the mappings in the alignment.

To this end, we define a model to dynamically estimate the quality of the mappings at each iteration, which consists of five different measures. These measures, which consider the mapping similarity and the feedback collected in previous iterations, are combined into two candidate selection strategies to present to the users the mappings that are estimated to have lower quality first.

This approach allows for the system to quickly adjust and is devised to run in a pay-as-you-go fashion, where we may stop the iterative process at any stage. A proportion of mappings is presented to the users for validation, which have been already validated in previous iterations. This proportion, called *revalidation rate*, can be configured to tune the robustness of the approach against users' errors. Our pay-as-you-go strategy is in opposition to first collecting a pre-determined number of validations n for each mapping, considering the majority vote after that, and only then propagating the user-provided feedback. During those n iterations, we would only be progressing on a single mapping. Following our approach, during n iterations we will be making progress on as many as n mappings and propagating the user-provided feedback at each iteration.

Previous approaches to ontology matching assume that feedback is given by individual users or that users always validate a mapping correctly [8,19,5]. Errors are considered only in one approach [10]. However, this approach does not consider the activity of different users, which plays a crucial role in our work. Therefore, we want to show that a high-quality alignment can be attained by involving multiple users so as to reduce the effort required by each individual user while allowing for user error.

To evaluate our approach we simulate the user feedback considering different *error rates*, which measure the errors made by the users in a sequence of iterations. We conduct experiments with the OAEI Benchmarks track to evaluate the gain in quality (measured in terms of F-measure) and the robustness (defined as the ratio between the quality of the alignment for a given error rate and the quality of the alignment when no errors are made) as a function of the number of validations for different error and revalidation rates. Our results highlight complex trade-offs and point to the benefits of adjusting the revalidation rate.

In Section 2, we describe the architecture of the multi-user feedback ontology matching system and give an overview of the combined automatic and manual process. In Section 3, we describe the key elements of the proposed approach: a model for the evaluation of the quality of the mappings, the ranking functions used for candidate mapping selection, and the method used for feedback propagation. In Section 4, we present the results of our experiments conducted on the OAEI Benchmarks track. In Section 5, we describe related work. Finally, in Section 6, we draw some conclusions and describe future work.

2. Approach Overview

We assume that as members of a community, domain users are committed to an ontology matching task and are overall reliable. Therefore, we do not deal with problems such as the engagement of users or the assessment of their reliability, which have been investigated in crowdsourcing approaches [18]. Even if we consider possible errors in validating mappings, thus causing inconsistency among users, we assume consistency for the same user, thus we do not present the same mapping more than once to the same user. We also do not distinguish among users although some users may make fewer errors than others. Instead we consider an overall error rate associated with a sequence of validated mappings. We assume that given a group of users whose reliability is known (or can be estimated), we can determine the corresponding error rate.

The validation of a mapping m by a user assigns a label l to that mapping. We define the homonymous function *label*, such that *label(m)* has value 1 or 0 depending on whether the user considers that m is or is not part of the alignment, respectively. When more than one user is involved, we use a consensus-based approach to decide whether a mapping belongs to an alignment. Consensus models include a simple majority vote, a sophisticated weighted majority vote, or more complex models such as tournament selection [2]. In this paper, we consider a simple majority vote, where *V* is an odd number of validations considered sufficient to decide by majority (we do not require that all the users vote on each mapping); thus, *minimum consensus*, $\mu = \lfloor (V/2) + 1 \rfloor$, is the minimum number of similar labels that is needed to make a correct decision on a mapping. For example, if $V = 5$ is the number of validations considered sufficient to decide by majority, a correct decision on a mapping can be taken when $\mu = 3$ similar labels are assigned to a mapping by the users.

We restrict our focus to equivalence mappings. Differently from other interactive techniques for ontology matching [14], our approach is independent from the cardinality of the alignment, because the desired cardinality can be set at the end of feedback loop.

The architecture of our multi-user ontology matching strategy can be built around any ontology matching system. In our case, we use AgreementMaker [4]. We list the steps of the feedback loop workflow:

Step 1: Initial Matching. During the first iteration, before feedback is provided, all data structures are created. A set of k matchers is run, each one creating a *local similarity matrix* where the value of each element (i, j) is the similarity score associated with mapping $m_{i,j}$ of element i of the source ontology to element j of the target ontology. For each mapping we can then define a *signature vector* with the k similarity scores computed for that mapping by the k individual matchers [5]. The results of the individual matchers are combined into a *global similarity matrix* where the value of each element represents the similarity between two concepts, which is computed by aggregating the scores of individual matchers into a final score [3]. An optimization algorithm is run to select the final alignment so as to maximize the overall similarity [4] and satisfy the mapping cardinality.

Step 2: Validation Request. A user asks for a mapping to validate, triggering the feedback loop.

Step 3: Candidate Selection. For each user who requests a mapping to validate, a mapping is chosen using two different candidate selection strategies combined by one meta-strategy (explained in detail in Section 3.2). Each strategy uses quality criteria to rank the mappings. The highest ranked mappings are those mappings that are estimated to have lowest quality, the expectation being that they are the more likely to be incorrect. The mapping quality is assessed at each iteration. The first strategy ranks the mappings that have not been validated by any user in previous iterations, while the second strategy ranks the mappings that have been previously validated at least once. When a user requests a mapping for validation, the meta-strategy selects one candidate selection strategy and presents the highest-ranked mapping to the user. Our approach is inspired by active learning methods and aims to present to the users those mappings that are most informative for the ontology matching problem. Mappings that are wrongly classified by the system at a current iteration are considered to be informative, because the result can be improved as long as the error is corrected [19,5].

Step 4: User Validation. The selected mapping is validated by the user. The user can label a mapping as being correct or incorrect but can also skip that particular mapping when unsure of the label to assign to the mapping.

Step 5: Feedback Aggregation. A *feedback aggregation matrix* keeps track of the feedback collected for each mapping and of the users who provided that feedback. The data in this matrix are used to compute mapping quality measures in the candidate selection and feedback propagation steps.

Step 6: Feedback Propagation. This method updates the *global similarity matrix* by changing the similarity score for the validated mapping and for the mappings whose signature vector is close to the signature vector of the mapping that was just validated, according to a distance measure.

Step 7: Alignment Selection. An optimization algorithm [4] used in **Step 1**, is run on the updated *similarity matrix* as input, and a refined alignment is selected. At the end of this step, we loop through the same steps, starting from **Step 2**.

3. Quality-Based Multi-User Feedback

In this section we describe the Candidate Selection and Feedback Propagation steps, which play a major role in our model. First, we explain the Mapping Quality Model, which is used by both steps.

3.1. Mapping Quality Model

We use a mapping quality model to estimate the quality of the candidate mappings, which uses five different mapping quality measures. The quality of a mapping estimated by a measure is represented by a score, which is higher for the mappings that are considered of higher quality. The score assigned to the mappings is always normalized in the interval $[0,1]$, which has two advantages. First, for every quality measure Q, we can define a measure Q^- in the same interval $[0,1]$, where the score for a mapping m is obtained by subtracting the quality score $Q(m)$ from 1. While a quality measure Q is used to rank mappings in increasing order of quality, a measure Q^- is used to rank mappings in decreasing order of quality. Rankings defined with a measure Q^- are inverted compared to rankings defined with a quality measure Q. Second, scores estimated with different measures can be easily combined with aggregate functions, e.g., maximum or average.

Automatic Matcher Agreement (AMA). It measures the agreement of the similarity scores assigned to a mapping by different automatic matchers and is defined as $\mathit{AMA(m)} = 1 - \mathit{DIS(m)}$, where *DIS(m)* is the *Disagreement* associated with mapping m. It is defined as the variance of the similarity scores in the signature vector and is normalized to the range $[0,1]$ [5]. Since *Disagreement* plays an important role in our approach, we will use the notation *DIS* instead of the superscript notation that we use for other measures.

As an example, given a mapping m with a signature vector $\langle 1,1,0,0\rangle$, where each value represent a similarity score returned by one automatic matcher, $\mathit{AMA}(m) = 0$ indicates that there is no agreement among the automatic matchers.

Cross Sum Quality (CSQ). Given a source ontology with n concepts, a target ontology with p concepts, and a matrix of the similarity scores between the two ontologies, for each mapping $m_{i,j}$ the *cross sum quality* sums all the similarity scores $\sigma_{i,j}$ in the same ith row and jth column of the matrix. The sum is normalized by the maximum sum of the scores per column and row in the whole matrix, respectively denoted by max_R and max_C, as defined in Equation 1.

$$\mathit{CSQ}(m_{i,j}) = 1 - \frac{\sum_{h=1}^{p} \sigma_{i,h} + \sum_{k=1}^{n} \sigma_{k,j}}{max_R + max_C} \quad (1)$$

This measure assigns a higher quality score to a mapping that does not conflict with other mappings, a conflict occurring when there exists another mapping for the same source or target concept. This measure takes into account the similarity score of the mappings, assigning a lower quality to mappings that conflict with mappings of higher similarity.

Table 1

An example of a similarity matrix. Empty cells have value 0.

i \ j	0	1	2	3	4	5
0	0.45					0.70
1					0.30	
2			0.60			
3		0.50			0.90	
4				0.80		
5	0.40		0.10			0.90

Table 2

Examples for the *Consensus (CON)* and *Feedback Stability (SF)* quality measures with $\mu = 3$.

Mapping	T_{m_i}	F_{m_i}	*CON*(m_i)	*SF*(m_i)
m_1	1	1	0.00	0.00
m_2	1	0	0.33	0.33
m_3	2	1	0.33	0.5

For the matrix of Table 1, the values of $\mathit{CSQ}(m_{3,4})$ and $\mathit{CSQ}(m_{2,2})$ are:

$$\mathit{CSQ}(m_{3,4}) = 1 - \frac{1.2 + 1.4}{1.4 + 1.6} = 0.13$$

$$\mathit{CSQ}(m_{2,2}) = 1 - \frac{0.6 + 0.7}{1.4 + 1.6} = 0.57$$

Mapping $m_{2,2}$ has higher quality than $m_{3,4}$ because $m_{2,2}$ has only one conflict with $m_{5,2}$ while $m_{3,4}$ has two conflicts, $m_{1,4}$ and $m_{3,1}$. Also, the conflicting mapping $m_{5,2}$ has lower similarity than the conflicting mappings $m_{1,4}$ and $m_{3,1}$, further contributing to the difference in quality between $m_{3,4}$ and $m_{2,2}$.

Similarity Score Definiteness (SSD). This measure ranks mappings in increasing order of quality. It evaluates how close the similarity σ_m associated with a mapping m is to the similarity scores' upper and lower bounds (respectively 1.0 and 0.0) using Equation 2.

$$SSD(m) = |\sigma_m - 0.5| * 2 \quad (2)$$

SSD will assign higher quality to the mappings considered more definite in their similarity score. The least definite similarity score is 0.5.

For the matrix of Table 1, the values of $SSD(m_{0,0})$ and $SSD(m_{3,4})$ are:

$$SSD(m_{0,0}) = |0.45 - 0.5| * 2 = 0.1$$

$$SSD(m_{3,4}) = |0.9 - 0.5| * 2 = 0.8$$

Consensus (CON). This measure ranks mappings in increasing order of quality. In the multi-user ontology matching scenario, a candidate mapping may be labeled as correct by some users and as incorrect by others. In our approach we assume that the majority of users are able to make the correct decision. The *consensus (CON)* quality measure uses the concept of minimum consensus μ, as defined in Section 2 to capture the user consensus gathered on a mapping at a given iteration. Let T_m and F_m denote respectively the number of times when a mapping has been labeled respectively as correct and incorrect. Given a mapping m, *CON*(m) is maximum when the mapping is labeled at least μ times as correct, as defined in Equation 3.

$$CON(m) = \begin{cases} 1 & \text{if } T_m \geq \mu \text{ or } F_m \geq \mu \\ \frac{T_m - F_m}{\mu} & \text{otherwise} \end{cases} \quad (3)$$

Three examples of *CON* quality evaluation are shown in Table 2. According to the consensus gathered among the users, the quality of mappings m_2 and m_3 is higher than the quality of mapping m_1.

Feedback Stability (FS). Given the current set of user validations received by the system at some iteration, *FS* estimates the impact of future user validations on the similarity evaluation in the feedback propagation step of the loop. Using the concept of *minimum consensus* (μ), *FS* tries to identify the mappings that are more stable in the system. Intuitively, mappings are more stable when minimum consensus has been reached, or when they have been assigned one label (correct or incorrect) a higher number of times than the other. In addition, the number of similar labels assigned to a mapping tells us how close the system is to reaching minimum consensus on that mapping. Instead, the more unstable mappings are the ones that have been assigned a label correct and incorrect an equal number of times. For these mappings, a new validation will bring more information into the system. By breaking a "tie" in user validations, the system come closer to make a decision. Defining $\Delta T_m = \mu - T_m$ and $\Delta F_m = \mu - F_m$, then:

$$FS(m) = \begin{cases} 1 & \text{if } T_m = \mu \text{ or } F_m = \mu \\ 1 - \frac{min(\Delta T_m, \Delta F_m)}{max(\Delta T_m, \Delta F_m)} & \text{otherwise} \end{cases} \quad (4)$$

The fraction in Equation 4 measures the instability of a mapping, defined as the ratio between the minimum and the maximum distances from minimum consensus of the number of similar labels assigned to a mapping. For $\Delta T_m = \Delta F_m$ this fraction is always equal to 1, meaning that a mapping m will be assigned a quality $FS(m) = 0$. We also observe that *FS* is always defined in the interval $[0, 1]$, and that when minimum consensus on a mapping m has not been reached, $FS^{-}(m) = \frac{min(\Delta T_m, \Delta F_m)}{max(\Delta T_m, \Delta F_m)}$.

Considering the examples in Table 2, mapping m_1 has the lowest *SF* score because we are in a tie situation and new feedback on that mapping is required. Mapping m_3 has a high *SF* score because the number of times it was labeled as correct is close to μ. Mapping m_2 has medium *SF* because, despite $T_{m_2} - F_{m_2} = T_{m_3} - F_{m_3}$ the number of times that m_2 has been validated as correct is more distant from μ. As can be seen from the example in Table 2, the intuition captured by *SF* is slightly different from the one captured by *CON*. While $CON(m_2) = CON(m_3) = 1/3$, m_2 and m_3 have different *SF* scores.

3.2. Quality-Based Candidate Selection

We combine the proposed quality measures using well-known aggregation functions to define two different candidate selection strategies: *Disagreement and Indefiniteness Average (DIA)*, which is used to select unlabeled mappings (mappings that have not been validated by any user in previous iterations) and *Revalidation (REV)*, which is used to select already labeled mappings (mappings that have been validated in previous iterations). Both strategies use quality measures that change over time and rank mappings at each iteration.

The *DIA* strategy uses the function:

$$DIA(m) = AVG(DIS(m), SSD^{-}(m)) \quad (5)$$

It favors mappings that are at the same time the most disagreed upon by the automatic matchers and have the most indefinite similarity values. The two measures *CON* and *SF* cannot be used in this strategy because they consider previous validations. After an experimental evaluation of different combinations of the other quality measures, discussed in detail in Section 4.2, we found that the combination of *DIS* and *SSD* (without *CSQ*) is the best combination of measures to find those mappings that were misclassified by the automatic matchers.

The second strategy, *Revalidation (REV)*, ranks mappings using the function:

$$REV(m) = AVG(CSQ^{-}(m), CON^{-}(m), SF^{-}(m)) \quad (6)$$

This strategy favors mappings with lower consensus and that could have changed significantly, and harmfully, the quality of the current alignment. The analysis of the users' activity, which is explicitly captured by *CON* and *SF*, is crucial to this strategy. In addition, since several mappings might have similar *CON* and *SF* in the first iterations, *REV* favors also mappings with potential conflicts with other mappings leveraging the *CSQ* measure. In this strategy, *CSQ* is preferred to *DIS* and *DSS* because: i) to rank already labeled mappings, disagreement among users, measured with *CON* and *SF*, is more informative than disagreement among automatic matchers, measured by *DIS*, ii) labeled mappings will have very definite similarity scores, and, therefore, very similar *DSS* scores, and iii) more potential conflicts can emerge as more feedback is collected.

The mapping that is presented to the user is selected by a parametric meta-strategy, which picks the top mapping from one of the *DIA* or *REV* rankings. This meta-strategy uses two probabilities, p_{DIA} and p_{REV}, such that $p_{DIA} + p_{REV} = 1$, which are associated respectively to the *DIA* and *REV* strategies. The parameter p_{REV} is called *revalidation rate* and is used to specify the proportion of mappings presented to the users for validation that have been already validated in previous iterations. We consider a constant revalidation rate, because we do not have empirical data that shows whether the users make more (or fewer) errors as the matching process unfolds. If such evidence is found, the revalidation rate can be changed accordingly. The meta-strategy verifies also that the same mapping (chosen from the *REV* list) is not presented for validation to the same user more than once.

3.3. Quality-Based Feedback Propagation

When the selected mapping is validated by a user, the feedback is propagated by updating a subset of the Similarity Matrix. We experimentally evaluated several feedback propagation methods, including a method used in our previous work [5], a method based on learning similarity scores with a multiple linear regression model, and a method based on our quality measures. For our experiments, we use this last method, which we call *Quality Agreement (QA) Propagation*, because it achieves the best trade-off between speed and robustness.

The method we used in previous work sets the similarity of the cluster of mappings that have the signature vector equivalent to the vector of the mapping validated by the user. The similarity of this cluster of mappings is set to 1 or 0 depending on the label given to the validated mapping by the user. This method has the disadvantage of propagating the user feedback on a very limited number of mappings. The method based on the multiple linear regression model learns the dependency between the values in the signature vectors of the mappings and the similarity values in the global similarity matrix. We found that this method has the disadvantage of requiring many user inputs before producing meaningful predictions.

In QA Propagation, the similarity of the validated mapping is set to 1 or 0 depending on the label assigned by the user. To propagate the similarity to other mappings, we compute the Euclidean distance between the signature vectors of the validated mapping, denoted by v, and the signature vectors of all the mappings for which consensus has not been reached. A distance threshold θ is used to identify the class of mappings most similar to the mapping labeled by the user. The mappings in this class have their similarity increased if the validated mapping is labeled as correct, and decreased otherwise. The change is proportional to: 1) the quality of the labeled mapping v and of the mappings m in the similarity class, measured respectively by two quality measures Q and Q', and 2) a *propagation gain* defined by a constant g such that $0 \leq g \leq 1$, which regulates the magnitude of the update. This constant will determine how much the quality of the labeled mapping will affect the quality of the mappings in the similarity class. Let $\delta = Q(v) * Q'(m) * g$ be this change factor. After the propagation of a validation $label(v)$, the similarity σ_m^t of a mapping m in the similarity class of v at an iteration t is defined by:

$$\sigma_m^t = \begin{cases} \sigma_m^{t-1} + min\left(\delta, 1 - \sigma_m^{t-1}\right) & \text{if } label(v) = 1 \\ \sigma_m^{t-1} - min\left(\delta, \sigma_m^{t-1}\right) & \text{if } label(v) = 0 \end{cases} \quad (7)$$

We adopt a conservative approach to propagation to make the system more robust to erroneous feedback. We define $Q(v) = \mathit{CON}(v)$ and $Q'(m) = \mathit{AVG}(\mathit{AMA}(m), \mathit{SSD}(m))$. Thus, the similarity of the mappings in this class is increased/decreased proportionally to: i) the consensus on the labeled mapping, and ii) the quality of the mappings in the similarity class. For example, for $\mathit{CON}(m_v) = 0$, the similarity of other mappings in the class is not updated. In addition, when $g = 0$, the propagation function changes the similarity of the validated mapping but not the similarity of other mappings in the class.

4. Experiments

We conduct several experiments to evaluate our multi-user feedback loop model. In a first set of experiments we evaluate the performance of the proposed pay-as-you-go method by analyzing the performance of different system configurations under various error rates and comparing it to the performance of a baseline approach. In a second set of experiments, we compare the performance of our mapping quality measures to the performance of other quality measures proposed in related work.

Table 3

AUGC for ontologies 101-303.

	ER	0.0	0.05	0.1	0.15	0.2	0.25
RR=0.0	NoGain	6.6	5.6	4.5	3.9	2.8	1.6
RR=0.0	Gain	**6.8**	**6.0**	**4.9**	**4.3**	2.9	2.1
RR=0.1	NoGain	6.0	5.3	4.4	3.6	3.0	2.5
RR=0.1	Gain	6.2	5.7	4.7	4.0	3.4	2.4
RR=0.2	NoGain	5.5	5.1	4.3	3.6	3.3	2.5
RR=0.2	Gain	5.7	5.0	4.7	3.8	2.9	2.9
RR=0.3	NoGain	5.2	4.9	4.4	4.2	3.8	**3.4**
RR=0.3	Gain	5.3	4.7	4.6	4.1	**3.9**	3.0
RR=0.4	NoGain	4.6	4.3	4.1	3.8	3.5	3.3
RR=0.4	Gain	4.7	4.4	4.2	3.8	3.7	3.0
RR=0.5	NoGain	4.1	3.9	3.7	3.6	3.6	3.2
RR=0.5	Gain	4.1	4.1	3.7	3.6	3.5	**3.4**

4.1. Performance under Different Error Rates

4.1.1. Experimental Setup

Our experiments are conducted using four matching tasks in the Benchmarks track of OAEI 2010, which consist of real-world bibliographic reference ontologies that include BibTeX/MIT, BibTeX/UMBC, Karlsruhe and INRIA, and their reference alignments. We chose these ontologies because they have been used in related studies [8,19,5,18].

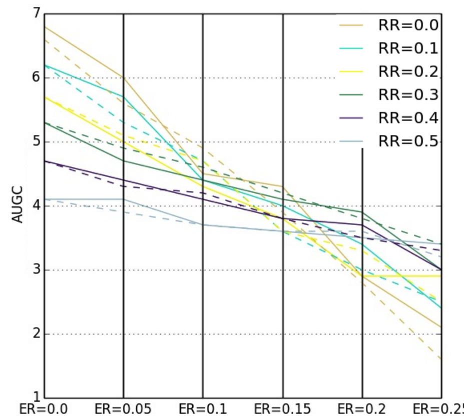

Fig. 2. Parallel coordinates of AUGC for ontologies 101-303.

In the evaluation we use two measures based on F-Measure:

Gain at iteration t, Δ*F-Measure*(t), is the difference between the F-Measure at iteration t as evaluated after the Candidate Selection Step and the F-Measure at the Initial Matching Step (see Section 2).

Robustness at iteration t, *Robustness*(t), is the ratio at iteration t of the F-Measure obtained under error rate *er*, $\mathit{FM}_{ER=er}(t)$, and the F-Measure obtained with zero error rate, $\mathit{FM}_{ER=0}(t)$, for the same configuration. A robustness of 1.0 means that the system is impervious to error.

The above measures characterize the behavior of the system in time. We need to consider two additional measures to represent this behavior with a single aggregate value, so as to ease the comparison among different configurations. The Area Under the Curve (AUC) can be used to describe a variable measured at different points in time, e.g., gain at iteration t, with an aggregate value. This value is defined by the area of the curve obtained by plotting the variable over time. The two aggregate measures based on AUC used in our experiments are defined as follows.

Area Under the Gain Curve, *AUGC*, is a measure that provides an aggregate representation of the gain in F-Measure until a fixed iteration *n*:

$$\mathit{AUGC} = \sum_{t=1}^{n} \Delta \mathit{F\text{-}Measure}(t) \quad (8)$$

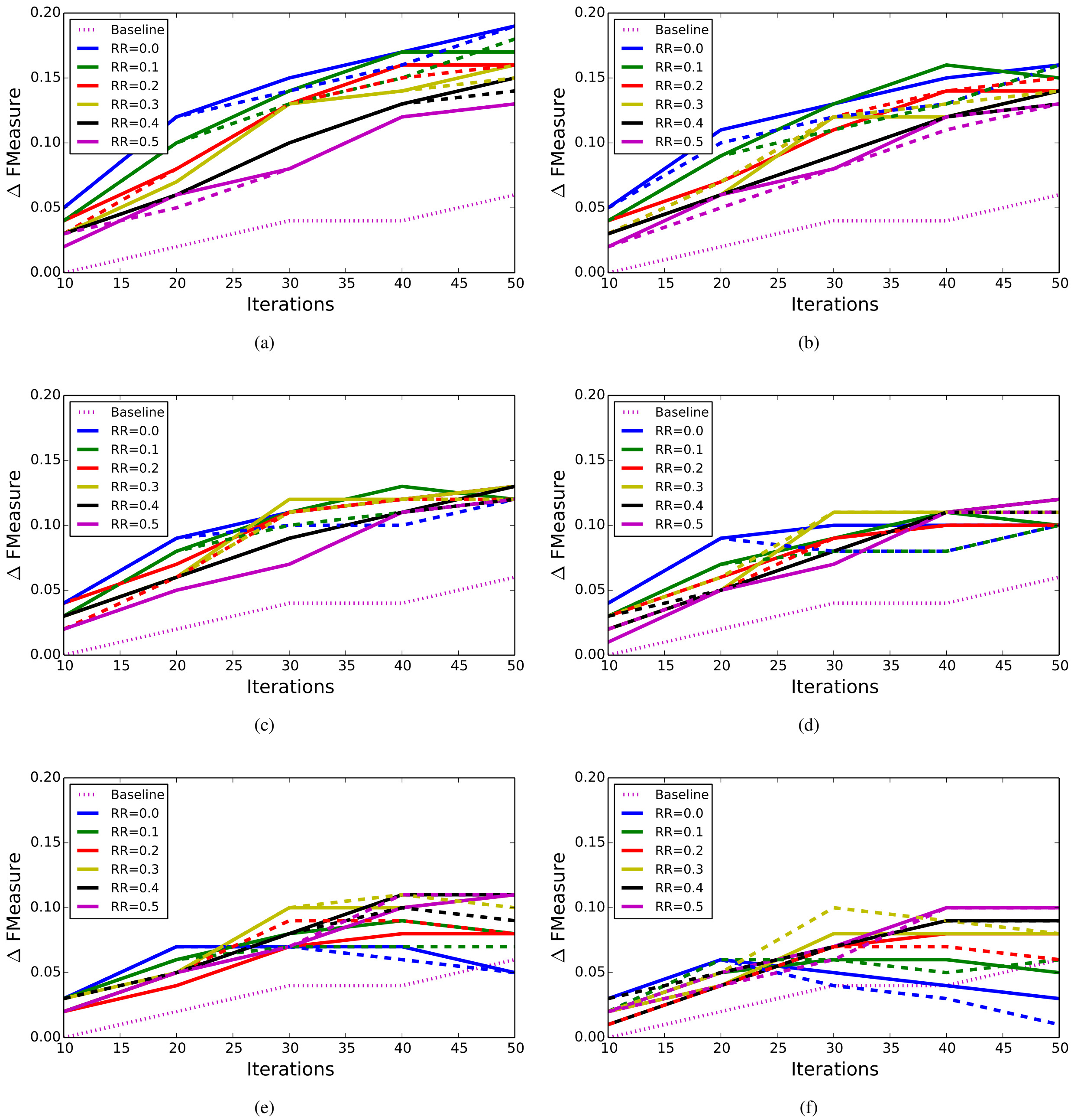

Fig. 1. Each chart presents $\Delta F\text{-}Measure(t)$ obtained for ontologies 101-303 with a different error rate (ER): (a) ER = 0.0; (b) ER = 0.05; (c) ER = 0.1; (d) ER = 0.15; (e) ER = 0.2; (f) ER = 0.25. The dashed lines represent a propagation gain equal to zero.

This measure is similar to Area Under the Learning Curve (AULC), which has been recently proposed to evaluate interactive ontology matching systems [17]. In AULC, absolute F-Measure is used instead of gain in F-Measure and the iteration axis uses a logarithmic scale to reward a quicker increase of F-Measure. We use gain in F-Measure to better emphasize the difference from the initial F-Measure. We also do not adopt a logarithmic scale because this would not adequately penalize a decrease in F-Measure after a certain number of iterations, which can happen when user errors are considered.

Area Under the Robustness Curve, *AURC*, is a measure that provides an aggregate representation of the Robustness until iteration *n*:

$$AURC = \sum_{t=1}^{n} Robustness(t) \quad (9)$$

We conduct our experiments by simulating the feedback provided by the users. Our focus is on the evaluation of the methods minimize the users' overall effort and make the system robust against users' errors. This kind of simulation is needed to comparatively assess the effectiveness of different candidate selection and propagation methods before performing experiments with real users, where presentation issues play a major role. We consider a community of 10 users, and simulate their validation at each iteration using the reference alignment. We note that we have made two assumptions that can be revised as they do not alter the substance of the method. The first reflects the fact that we do not distinguish among users as mentioned in Section 2 and therefore consider a constant error rate for each sequence of validated mappings. A Constant error rate has been applied to other interactive ontology matching approaches [10]. The study of a community of users might uncover an appropriate probability distribution function for the error (e.g., Gaussian). The second assumption is related to the choice of the number of validations V considered sufficient to decide by majority, which we set to 5, and therefore $\mu = 3$. Studying the users could lead to setting V so as to guarantee a desired upper bound for the error rate. Without this knowledge, we considered several error rates while keeping V constant.

In the Initial Matching Step we use a configuration of AgreementMaker that runs five lexical matchers in parallel. The LWC matcher [4] is used to combine the results of five lexical matchers, and two structural matchers are used to propagate the similarity scores. The similarity scores returned by these matchers are used to compute the signature vectors. In our experiments we compute the gain and robustness at every iteration t from 1 to 100, with six different error rates (ER) (0.05, 0.1, 0.15, 0.2, 0.25) and twelve different system configurations. The configurations stem from the six different revalidation rates (RR) (0.0, 0.1, 0.2, 0.3, 0.4, 0.5) used in candidate selection strategy, and two different feedback propagation gains, $g = 0$ and $g = 0.5$. When $g = 0$, the propagation step affects only the mapping validated by the user, that is, it does not change the similarity of other mappings. We set the threshold used for cluster selection $\theta = 0.03$. This value is half the average Euclidean distance between the signature vectors of the first 100 validated mappings and the remaining mappings with a non-zero signature vector. Remarkably, this value was found to be approximately the same for all matching tasks, thus being a good choice. In the Alignment Selection Step we set the cardinality of the alignment to 1:1. The evaluation randomly simulates the labels assigned by the users according to different error rates. Every experiment is therefore repeated twenty times to eliminate the bias intrinsic in the randomization of error generation. In the analysis of the results we will report the average of the values obtained in each run of the experiments.

We also want to compare the results obtained with our model, which propagates the user feedback at each iteration in a pay-as-you-go fashion, with a model that adopts an *Optimally Robust Feedback Loop (ORFL)* workflow, inspired by CrowdMap, a crowdsourcing approach to ontology matching [18]. In their approach, similarity is updated only when consensus is reached on a mapping, which happens after five iterations when $V = 5$. To simulate their approach we modify our feedback loop in such a way that a correct validation is generated every five iterations (it is our assumption that the majority decision is correct). CrowdMap does not use a candidate selection strategy because all the mappings are sent in parallel to the users. We therefore use our candidate selection strategy with $RR = 0$ to define the priority with which mappings are validated and do not propagate the similarity to other mappings.

4.1.2. *Result Analysis*

We ran our first experiment on two of the OAEI Benchmarks ontologies, 101 and 303. We chose these ontologies because their matching produced the lowest initial F-Measure (0.73) when compared with the results for the other matching tasks 101-301 (0.92), 101-302 (0.86) and 101-304 (0.93). Thus we expect to see a higher gain for 101-303 than for the others. Table 4 shows for each matching task the number of correct mappings, false positives, false negatives, and F-Measure after the initial matching step.

Figure 1 shows the gain in F-Measure after several iterations using different configurations of our model and the ORFL approach. Each chart presents results

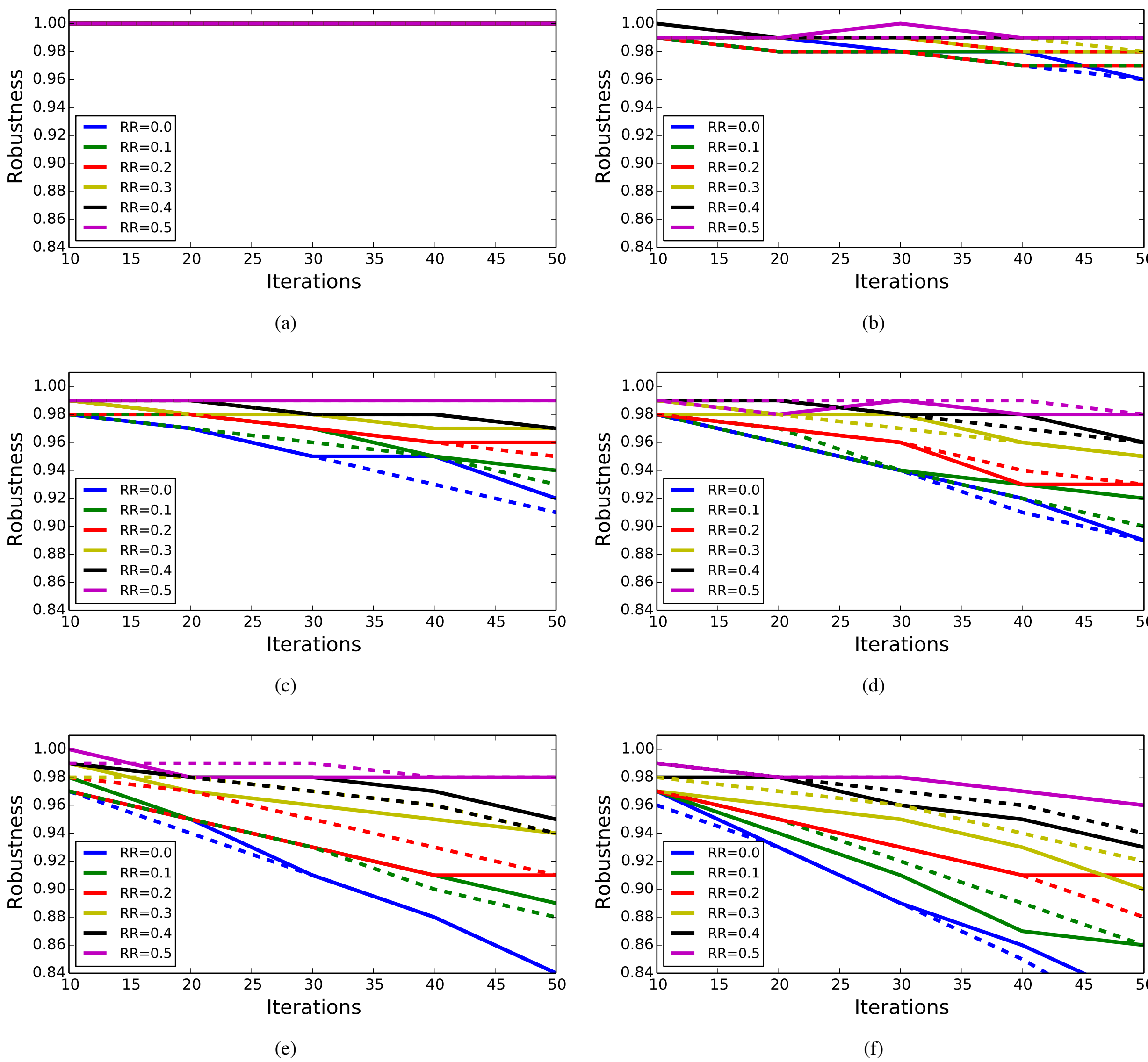

Fig. 3. Each chart presents *Robustness*(t) obtained for ontologies 101-303 with a different error rate (ER): (a) ER = 0.0; (b) ER = 0.05; (c) ER = 0.1; (d) ER = 0.15; (e) ER = 0.2; (f) ER = 0.25. Dashed lines represent a propagation gain equal to zero.

Table 4

Results after the initial matching step.

Matching Task	# Correct Mappings	# False Positives	# False Negatives	F-Measure
101-301	50	6	2	92.31
101-302	36	5	5	86.11
101-303	40	23	4	72.73
101-304	74	9	2	92.90

for a specific error rate (ER). Solid lines represent configurations with propagation gain $g = 0.5$, while dashed lines represent configurations with zero propagation gain. Different colors are associated with different revalidation rates (RR). The dotted line represent the results obtained with the ORFL approach. In the charts, the steeper a curve segment between two iterations, the faster the F-measure gain between those iter-

Table 5

AURC for ontologies 101-303.

	ER	0.0	0.05	0.1	0.15	0.2	0.25
RR=0.0	NoGain	**50.0**	48.8	47.4	46.8	45.4	44.3
RR=0.0	Gain	**50.0**	49.0	47.7	46.9	45.5	44.7
RR=0.1	NoGain	**50.0**	48.9	47.9	47.1	46.3	45.9
RR=0.1	Gain	**50.0**	49.1	48.2	47.3	46.6	45.5
RR=0.2	NoGain	**50.0**	49.3	48.4	47.8	47.4	46.4
RR=0.2	Gain	**50.0**	48.9	48.6	47.7	46.7	46.7
RR=0.3	NoGain	**50.0**	49.4	49.0	48.5	48.3	47.7
RR=0.3	Gain	**50.0**	49.3	48.9	48.5	48.1	47.1
RR=0.4	NoGain	**50.0**	49.5	49.1	48.9	48.4	48.4
RR=0.4	Gain	**50.0**	**49.6**	49.1	49.0	48.7	48.0
RR=0.5	NoGain	**50.0**	49.5	**49.5**	**49.4**	**49.3**	**48.8**
RR=0.5	Gain	**50.0**	**49.6**	**49.5**	49.2	49.2	**48.8**

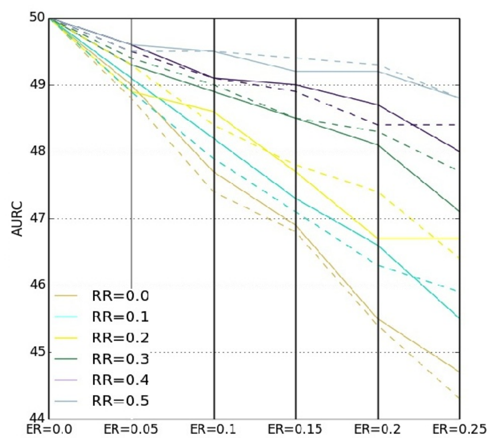

Fig. 4. Parallel coordinates of AURC for ontologies 101-303.

ations. It can be observed that our approach is capable of improving the quality of the alignment over time. However, it is also the case that as time increases the quality can decrease especially for higher error rates, that is, primarily for charts (d), (e), (f) of Figure 1. We can see that lower revalidation rates obtain better Δ*F-Measure*(t) with lower error rates. However, as error rate increases, e.g., for ER=0.2 and ER=0.25, better results are obtained with higher revalidation rates. Therefore, we infer that our *REV* strategy is effective in counteracting high error rates. Moreover, our approach is performing better than *ORFL* in all situations except the one with highest error rate and lowest revalidation rate.

Table 3 shows *AUGC* for the charts presented in Figure 1. *AUGC* is also plotted using parallel coordinates in Figure 2, where each parallel line represents a different error rate. It is evident from Table 3 that propagation always helps to obtain the maximum AUGC for every error rate. However, for some revalidation rates and some error rates, *AUGC* is higher when the feedback is not propagated, remarkably for RR=0.2 and ER=0.2, RR=0.3 and ER=0.25, RR=0.4 and ER=0.25. Propagation is more frequently effective for lower error rates, e.g., for an error rate up to 0.1, which can be explained by the higher probability of error propagation when the error rate increases. Finally, it can be seen from Figure 1 that AUGC decreases monotonically for every configuration as the error rate increases, but this decrease is less prominent for higher revalidation rates (represented by gentler AUGC curves). This observation indicates that our *REV* strategy helps to make the feedback loop more tolerant to user errors.

Figure 3 shows the robustness of different configurations evaluated at different iterations, varying both the error and the revalidation rates. Each chart presents results for a specific error rate (ER). Solid lines represent configurations with propagation gain $g = 0.5$, while dashed lines represent configurations with zero propagation gain. Different colors represent results obtained with different revalidation rates. Robustness decreases as time increases and revalidation rate decreases, more noticeably for high error rates. However, robustness decreases at a much lower rate with high revalidation rate, as shown by the gentler curves in Figure 3.

Table 5 shows *AURC* for the charts presented in Figure 3. *AURC* is also plotted using parallel coordinates in Figure 4. As error rates increase, we see a sharp monotonic decrease in robustness. However, as the revalidation rates increase, robustness always increases, except in one case for RR=0.2 and ER=0.05. This observation indicates that with high revalidation rates the system becomes less sensitive to the error rate. Moreover, it can be seen from Table 5 that configurations with propagation gain are more robust than configurations with zero propagation gain for low revalidation and error rates. When error rate increases and a high revalidation rate is used, configurations with zero propagation gain are more robust than configurations with propagation gain.

We ran further experiments with three other matching tasks of the OAEI 2010 Benchmarks track. Table 6 contains the results for the three other tasks (101-301, 101-302, 101-304) and shows Δ*F-Measure*(t) at different iterations under two different error rates (0.0 and 0.1), two different revalidation rates (0.2 and 0.3), in different configurations with or without gain (Gain or NoGain), for our pay-as-you-go workflow, together with a comparison with ORFL. We discuss the results for an error rate up to 0.1 because the initial F-Measure in these matching tasks is high (0.92, 0.86, and 0.93,

Table 6
Δ*F-Measure*(t) for the matching tasks with higher initial F-Measure.

ER	RR	CONF	101-301(0.92)				101-302(0.86)				101-304(0.92)			
			@10	@25	@50	@100	@10	@25	@50	@100	@10	@25	@50	@100
0.0	0.2	NoGain	**0.03**	**0.05**	**0.05**	**0.05**	**0.03**	0.05	**0.06**	**0.08**	**0.0**	**0.05**	**0.05**	**0.05**
0.0	0.2	Gain	**0.03**	0.04	0.04	**0.05**	**0.03**	**0.06**	**0.06**	**0.08**	**0.0**	**0.05**	**0.05**	**0.05**
0.0	0.3	NoGain	0.02	**0.05**	**0.05**	**0.05**	**0.03**	0.05	**0.06**	**0.08**	**0.0**	0.04	**0.05**	**0.05**
0.0	0.3	Gain	0.02	0.04	0.04	**0.05**	**0.03**	0.05	**0.06**	**0.08**	**0.0**	0.03	**0.05**	**0.05**
0.1	0.2	NoGain	**0.03**	**0.04**	0.01	-0.01	0.02	0.01	0.0	-0.02	**0.0**	**0.03**	0.03	0.00
0.1	0.2	Gain	**0.03**	0.03	0.01	**0.0**	0.02	**0.03**	**0.01**	**0.01**	**0.0**	**0.03**	0.03	0.00
0.1	0.3	NoGain	0.02	**0.04**	**0.02**	**0.0**	**0.03**	0.02	0.00	**0.01**	**0.0**	**0.03**	**0.04**	**0.02**
0.1	0.3	Gain	0.02	0.03	0.01	**0.0**	**0.03**	**0.03**	**0.01**	**0.01**	**0.0**	**0.03**	**0.04**	0.01
-	0.0	ORFL	0.0	0.02	0.04	0.05	0.01	0.03	0.05	0.05	0.0	0.0	0.0	0.05

respectively), therefore we do not expect that users will make more errors than automatic matchers. In the absence of error, our model always improves the quality of the alignment for the three tasks faster than ORFL (except for iteration 100 of 101-304 where both methods have the same gain of 0.05). For an error rate of 0.1, our model performs better than ORFL for $t = 10$ for every matching task, and for $t = 25$ in two of them. For $t = 50$ it performs worse than ORFL for two of the tasks and better for one of the tasks. For $t = 100$, ORFL always performs better.

4.2. Comparison with Quality Measures Proposed in Related Work

We establish a comparison between our mapping quality model and the measures used in the candidate selection of the single user approach of Shi et al. [19]. We want to determine which quality model performs better in our feedback loop workflow. The candidate selection strategy used by Shi et al. uses three measures, *Contention Point, Multi-Matcher Confidence,* and *Similarity Distance*, whose intent is close to that of our quality measures *CSC, DIS,* and *SSD*.

We ran an experiment with all the four matching tasks of the OAEI 2010 Benchmarks track (101-301, 101-302, 101-303, 101-304), in an error-free setting (like the one considered by Shi et al.) with no propagation gain. We consider the measures of our model that are meaningful in an error-free setting, i.e., *CSQ*$^-$, *DIS*, and *SSD*$^-$. We compare *DIA* (see Equation 6) with several selection strategies defined using individual measures and significant combinations of them, i.e., maximum, minimum and average. For the evaluation we look at the list of top-100 ranked mappings returned by each strategy and we measure: the number of false positives and false negatives found in the list, Δ*F-Measure*(t) obtained after validating the mappings in the list, and the *Normalized Discounted Cumulative Gain* (NDCG) of the ranked list.

NDCG is a well known measure used to evaluate the quality of a ranked list of results [12]. Discounted Cumulative Gain measures the gain of an item in a list based on its relevance and position. The gain is accumulated from the top of a result list of n elements to the bottom, with the gain of each result discounted at lower ranks:

$$\mathit{DCG} = rel_1 + \sum_{i=2}^{n} \frac{rel_i}{\log_2 i} \tag{10}$$

NDCG is defined by normalizing the cumulated gain by the gain of an ideal ranking:

$$\mathit{NDCG} = \frac{\mathit{DCG}}{\mathit{IDCG}} \tag{11}$$

In a list of mappings to present to the user for validation, a mapping will be validated at iteration t when holds the position t in the list. A mapping is considered relevant if it is misclassified by the system, i.e., if it is either a false positive or a false negative. The ideal ranking for a candidate selection is the ranking in which all the misclassified mappings are ranked on top of the mapping list. A candidate selection strategy has higher quality measured by *NDCG* when it ranks a high number of mappings in the first positions.

Table 7 shows the result of our experiments on four matching tasks (101-301, 101-302, 101-303, 101-304). We refer to the set of measures in Shi et al. as SLTXL, using the first letters of the names of each author. For each candidate selection strategy, Table 7 shows the number of misclassified mappings (#FP and #FN), Δ*F-Measure*(t) and *NDCG*. The values of Δ*F-Measure*(t) and *NDCG* for Candidate selection strategies based on our mapping quality measures significantly outperform the strategies based on the measure proposed by Shi et al.. All the quality measures are more effective in finding false positives than false negatives, but a limited number of false negatives is found by our measures in every matching task. *DIA* is the strategy that performs on average better, with an average $\Delta F\text{-}Measure(t)$ equal to 0.11.

4.3. Conclusions

From our experiments with four different matching tasks characterized by different initial F-Measure values, we draw the following conclusions:

1. When users do not make errors, our method improves the quality of the alignment much faster in every matching task than an optimally robust feedback loop (ORFL) method that labels a mapping only after having collected from the users every validation needed to reach consensus.
2. An increasing error rate can be counteracted by an increasing revalidation rate, still obtaining very good results for an error rate as high as 0.25 and a revalidation rate of 0.5.
3. In the presence of errors, our approach is particularly effective when the initial alignment has lower quality and includes a higher number of false positives (see Table 4). In the matching task with lower initial F-Measure, every configuration of our method improves the quality of the alignment much faster than the optimally robust feedback loop method, even when error rates are as high as 0.25. Propagating the feedback to mappings other than the mapping labeled by the user at the current iteration shows a higher gain in F-Measure in several of the experiments.
4. In the presence of errors, the F-Measure gain decreases after a certain number of iterations, unless a high revalidation rate is used. The number of iterations after which the gain in F-Measure decreases, which is clearly correlated with the error rate, appears to also be correlated with the quality of the initial alignment and, in particular, with the number of false positives (see Table 4). For example, using a revalidation rate of 0.3 and an error rate of 0.1, the F-Measure gain starts to decrease after 25 iterations in matching tasks with at most six false positives in the initial alignment (101-301, 101-302), and does not decrease before the 50th iteration in matching tasks where the initial alignment contains at least nine false positives (101-303, 101-304).
5. When the error rate is unknown, a revalidation rate equal to 0.3 achieves a good trade-off between F-measure gain and robustness because of the "stability" of the results as displayed in the (d) charts of Figures 1 and 3. We note that propagation leads to better results for the F-measure gain than for robustness.
6. Propagation leads in general to better results (F-measure gain and robustness) than no propagation (See Table 3 and Table 5). There are however, a few exceptions. The most notorious in Table 3 is for ER=0.2 and RR=0.2. In this case, it appears that errors get propagated, without being sufficiently counteracted by revalidation. When revalidation rate increases to RR=0.3 then the results with propagation wins. Another example is when we have ER=0.25 and RR=0.3 in Table 5. The result without propagation is much more better than with propagation. However, when the revalidatio rate increases, the results become better with propagation. Finally, the RR=0.5 wins.
7. To avoid decreasing in the amount of robustness, we have to use high revalidation rates without considering the error rate according to figure 4. However, we should consider the error rate when we want to configure the system to avoid decreasing in FMeasure gain. Since figure 2 indicates that lower revalidation rates provide better results with lower error rates, and higher revalidation rates provide better results with higher error rates.
8. According to our results, the revalidation rate should be changed over time, starting with a lower revalidation rate and then switching to a higher revalidation rate. The higher the error, the sooner the switch should occur.

5. Related Work

Leveraging the contribution of multiple users has been recognized as a fundamental step in making user feedback a first class-citizen in data integration systems, such as those for schema and ontology matching [1, 18]. Ontology matching approaches relying on the feedback provided by a single user are a precursor to multi-user systems. They include the work of Shi et al. [19], Duan et al. [8], Cruz et al. [5], Noy et al. [15], To et al. [20], Jirkovsky et al. [11] and Jiménez-Ruiz et al. [10]. We discuss the single user approaches in two groups based on the content of their candidate selection method, and then we explain about the multi-user approaches.

The first group of single user approaches includes those systems which have a static candidate selection strategy. They provide the ranked list of candi-

Table 7

Comparison of different quality measures and their combinations, showing retrieved false positives, retrieved false negatives, $\Delta F\text{-}Measure(t)$ and NDCG at iteration 100.

	301				302				303				304			
	#FP	#FN	ΔFM	NDCG	#FP	#FN	ΔFM	NDCG	#FP	#FN	ΔFM	NDCG	#FP	#FN	ΔFM	NDCG
Contention Point	6	0	0.05	0.19	4	1	0.08	0.39	9	0	0.07	0.19	1	1	0.01	0.05
Multi Matcher Confidence	2	0	0.01	0.09	1	1	0.04	0.06	5	0	0.03	0.11	0	0	0.00	0.0
Similarity Distance	0	0	0.00	0.00	0	0	0.00	0.00	0	0	0.00	0.00	0	0	0.00	0.00
MAX(SLTXL)	0	0	0.00	0.00	0	0	0.00	0.00	0	0	0.00	0.00	0	0	0.00	0.00
MIN(SLTXL)	0	0	0.00	0.00	0	0	0.00	0.00	1	0	0.007	0.02	0	0	0.00	0.00
AVG(SLTXL)	4	0	0.03	0.14	3	1	0.06	0.29	7	0	0.05	0.18	0	0	0.00	0.00
CSQ^-	6	1	0.05	0.46	5	1	0.08	0.56	20	3	0.20	0.79	9	1	**0.06**	0.53
DIS	6	1	0.05	0.18	4	2	0.06	0.27	18	3	0.20	0.65	8	1	0.05	**0.79**
SSD^-	6	0	0.05	0.49	4	0	0.05	0.25	17	0	0.13	0.60	3	0	0.01	0.12
AVG(CSQ^-,DIS,SSD^-)	6	1	0.05	0.45	4	0	0.03	0.31	20	1	0.18	0.65	8	0	0.04	0.38
MAX(CSQ^-,DIS,SSD^-)	2	0	0.01	0.16	3	0	0.02	**0.75**	3	0	0.02	0.58	2	0	0.01	0.21
MIN(CSQ^-,DIS,SSD^-)	0	0	0.00	0.00	0	0	0.00	0.00	1	0	0.007	0.02	0	0	0	0.00
DIA	6	1	**0.06**	**0.69**	5	3	**0.11**	0.60	23	3	**0.26**	**0.80**	9	1	**0.06**	0.45

date mappings at the beginning of their process, and they do not change it during the iterations. These approaches do not consider error rate for the user except LogMap 2. Shi et al. use an active learning approach to determine an optimal threshold for mapping selection and propagate the user feedback using a graph-based structural propagation algorithm. Duan et al.ũse a supervised method to learn an optimal combination of both lexical and structural similarity metrics [19]. Cruz et al. use signature vectors that identify the mappings for which the system is less confident and propagate the validated mappings based on the similarity of signature vectors; the overall goal is to reduce the uncertainty of the mappings [5]. LogMap 2 is another interactive ontology matching system that is proposed by Jiménez-Ruiz et al. [10]. They try to find reliable and non-reliable mappings using some lexical, structural and reasoning-based techniques. They discard most of the non-reliable mappings, and request feedback for the remaining ones. They reject all the automatically obtained mappings if there is a conflict with user provided feedback. This kind of "propagation" is therefore completely different from our own. LogMap2 considers error rates in their evaluation, but they do not obtain the consensus over the feedback.

The second group of single user approaches includes those systems which have a dynamic candidate selection. They update the ranked list of candidate mappings at each iteration. None of them considers the error rate for users. Noy et al. use an interactive component in the PROMPT suite for ontology merging and mapping [15]. They ask users for feedback based on the structure of the ontologies, inconsistencies and potential problems in the alignment. Their candidate selection method is entirely different from ours because we rank candidate mappings based on the combination of quality measures, each with a particular emphasis. To et al. propose an adaptive machine learning framework for ontology matching using user feedback [20]. They use two kinds of user feedback in their approach: pre-alignment and relevance feedback. Pre-alignment is used at the beginning of the mapping process to train the learner, and relevance feedback is used in a semi-supervised method to iteratively improve the learner. The semi-supervised method suggests candidate mappings to the users at each iteration. They do not have a propagation method, but find actively the candidate mappings. Jirkovsky et al. propose MAPSOM, an interactive ontology matching approach [11]. They train a classifier using a neural network to combine their basic similarity measures. Then they use user feedback to improve their classifier and find the best configuration for it. Their approach uses user feedback to aggregate the initial matchers. However, our approach applies user feedback after the combination of initial matchers. They have a candidate selection method, which is based on their classifier boundary.

In multi-user scenarios, several opportunities arise, such as the possibility of gathering consensus on mappings, as well as challenges, such as the need to deal with noisy feedback [1,18]. Many multi-user scenarios use crowdsourcing on a web platform: for example, CrowdMap [18] for ontology matching and ZenCrowd [7] for data linking. As in our multi-user feedback approach, both CrowdMap and ZenCrowd engage multiple workers to solve a semantic-based matching task and use revalidation. However, CrowdMap does not integrate automatic matching methods with user feedback and does not investigate methods for candidate mapping selection nor feedback propagation.

Workers may not have specific skills nor a specific interest in the task that they perform other than the monetary reward that they get. Therefore, strategies are needed to assess their performance. For example, McCann et al. [13] classify workers as trusted or untrusted. Another example is provided by Osorno-Gutierrez et al. [16], who investigate the use of crowdsourcing for mapping database tuples. They address the workers' reliability, identifying both workers whose answers may contradict their own or others'. Meilicke et al. [14] propose a reasoning approach to identify the inconsistencies after manual mapping revision by human experts. One of their strategies is to remove some mappings from the search space based on the cardinality of the alignment (e.g., using the 1:1 cardinality assumption). Our feedback model works prominently on the similarity matrix: a desired cardinality constraint can be specified by configuring the alignment selection algorithm (Step 7).

Similarly to some single-user feedback strategies, the recent crowdsourcing approach of Zhang et al., aims to reduce the uncertainty of database schema matching [21] measured in terms of the entropy computed using the probabilities associated with sets of tuple correspondences, called matchings. They proposed two algorithms that generate questions to the crowd. Best candidates are those that can obtain highest certainty with lowest cost. In comparison with our approach, they do not obtain consensus on a mapping and each mapping is only validated once.

6. Conclusions and Future Work

A multi-user approach needs to manage inconsistent user validations dynamically and continuously throughout the matching task, while aiming to reduce the number of mapping validations so as to minimize user effort. In this paper, we presented a mapping model that uses quality measures in the two main steps of the system: the Candidate Mapping Selection and the Feedback Propagation steps. In the first step, a dynamic mechanism ranks the candidate mappings according to those quality measures so that the mappings with lower quality are the first to be presented for validation, thus accelerating the gain in quality. In the second step similarity among mappings is used to validate mappings automatically without direct user feedback, so as to cover the mapping space faster.

Our experiments brought clarity on the trade-offs among error and revalidation rates required to minimize time and maximize robustness and F-measure. Our strategies show under which circumstances we can afford to be "aggressive" by propagating results from the very first iterations, instead of waiting for a consensus to be built.

Future work may consider user profiling, so that there is a weight associated with the user validations and how they are propagated depending on the feedback quality. In this paper, we tested different constant error rates to model a variety of users' behavior as an aggregate. Other models may take into account the possibility that users' engagement decreases along time due to the repetitiveness of the validation task, thus leading to an increasing error rate, or that in certain situations users learn with experience and make fewer errors, thus leading to a decreasing error rate. We therefore plan to perform studies to determine the impact of users' behavior along time on the error distribution so as to change the candidate selection meta-strategy accordingly. Our overall strategy could also be modified to present one mapping together with several mapping alternatives. In this case, the visualization of the context for those alternatives could prove beneficial. This visualization can be included in a visual analytics strategy for ontology matching [5] modified for multiple users.

Acknowledgments

This work was supported in part by NSF Awards CCF-1331800, IIS-1213013, IIS-1143926, and IIS-0812258, by a UIC-IPCE Civic Engagement Research Fund Award, and by the EU FP7-ICT-611358 COMSODE Project.

References

[1] Khalid Belhajjame, Norman W. Paton, Alvaro A. A. Fernandes, Cornelia Hedeler, and Suzanne M. Embury. User Feedback as a First Class Citizen in Information Integration Systems. In *CIDR Conference on Innovative Data Systems Research*, pages 175–183, 2011.

[2] Julien Bourdaillet, Shourya Roy, Gueyoung Jung, and Yu-An Sun. Crowdsourcing Translation by Leveraging Tournament Selection and Lattice-Based String Alignment. In *AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, volume WS-13-18. AAAI, 2013.

[3] Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *PVLDB*, 2(2):1586–1589, 2009.

[4] Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. Efficient Selection of Mappings and Automatic Quality-driven Combination of Matching Methods. In *ISWC International Workshop on Ontology Matching (OM)*, volume 551 of *CEUR Workshop Proceedings*, pages 49–60, 2009.

[5] Isabel F. Cruz, Cosmin Stroe, and Matteo Palmonari. Interactive User Feedback in Ontology Matching Using Signature

Vectors. In *IEEE International Conference on Data Engineering (ICDE)*, pages 1321–1324. IEEE, 2012.

[6] Isabel F. Cruz and William Sunna. Structural Alignment Methods with Applications to Geospatial Ontologies. *Transactions in GIS, Special Issue on Semantic Similarity Measurement and Geospatial Applications*, 12(6):683–711, 2008.

[7] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. ZenCrowd: Leveraging Probabilistic Reasoning and Crowdsourcing Techniques for Large-scale Entity Linking. In *International World Wide Web Conference (WWW)*, pages 469–478, New York, NY, USA, 2012. ACM.

[8] Songyun Duan, Achille Fokoue, and Kavitha Srinivas. One Size Does Not Fit All: Customizing Ontology Alignment Using User Feedback. In *International Semantic Web Conference (ISWC)*, volume 6496 of *Lecture Notes in Computer Science*, pages 177–192. Springer, 2010.

[9] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007.

[10] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Yujiao Zhou, and Ian Horrocks. Large-scale interactive ontology matching: Algorithms and implementation. In *European Conference on Artificial Intelligence (ECAI)*, volume 242, pages 444–449, 2012.

[11] VÃąclav JirkovskÃ¡ and Ryutaro Ichise. Mapsom: User involvement in ontology matching. In *Joint International Semantic Technology conference (JIST)*, pages 348–363. Springer, 2014.

[12] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[13] Robert McCann, Warren Shen, and AnHai Doan. Matching Schemas in Online Communities: A Web 2.0 Approach. In *IEEE International Conference on Data Engineering (ICDE)*, pages 110–119. IEEE, 2008.

[14] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Supporting Manual Mapping Revision Using Logical Reasoning. In *National Conference on Artificial Intelligence (AAAI)*, pages 1213–1218. AAAI Press, 2008.

[15] Natalya F. Noy and Mark A. Musen. The PROMPT Suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59:983–1024, 2003.

[16] Fernando Osorno-Gutierrez, Norman W. Paton, and Alvaro A. A. Fernandes. Crowdsourcing Feedback for Pay-As-You-Go Data Integration. In *VLDB Workshop on Databases and Crowdsourcing (DBCrowd)*, volume 1025, pages 32–37, 2013.

[17] Heiko Paulheim, Sven Hertling, and Dominique Ritze. Towards evaluating interactive ontology matching tools. In *European Semantic Web Conference (ESWC)*, pages 31–45. Springer, 2013.

[18] Cristina Sarasua, Elena Simperl, and Natalya Fridman Noy. CrowdMap: Crowdsourcing Ontology Alignment with Microtasks. In *International Semantic Web Conference (ISWC)*, volume 7649 of *Lecture Notes in Computer Science*, pages 525–541. Springer, 2012.

[19] Feng Shi, Juanzi Li, Jie Tang, Guotong Xie, and Hanyu Li. Actively Learning Ontology Matching via User Interaction. In *International Semantic Web Conference (ISWC)*, volume 5823 of *Lecture Notes in Computer Science*, pages 585–600. Springer, 2009.

[20] Hoai-Viet To, Ryutaro Ichise, and Hoai-Bac Le. An adaptive machine learning framework with user interaction for ontology matching. In *International Joint Conferences on Artifical Intelligence, Workshop on Information Integration on the Web*, pages 35–40, 2009.

[21] Chen Jason Zhang, Lei Chen, H. V. Jagadish, and Chen Caleb Cao. Reducing Uncertainty of Schema Matching via Crowdsourcing. *PVLDB*, 6(9):757–768, 2013.