Control and Cybernetics

vol. **41** (2012) No. 4

Introducing artificial neural network in ontologies alignment process*

by

Warith Eddine Djeddi and Mohamed Tarek Khadir

LabGED, Computer Science Department, University Badji Mokhtar, PO-Box 12, 23000 Annaba, Algeria

Abstract: Ontology alignment uses different similarity measures of different categories such as string, linguistic, and structural based similarity measures to understand ontologies' semantics. A weights vector must, therefore, be assigned to these similarity measures, if a more accurate and meaningful alignment result is favored. Combining multiple measures into a single similarity metric has been traditionally solved using weights determined manually by an expert, or calculated through general methods (e.g. average or sigmoid function) that do not provide optimal results. In this paper, we propose an artificial neural network algorithm to ascertain how to combine multiple similarity measures into a single aggregated metric with the final aim of improving the ontology alignment quality. XMap++ is applied to benchmark tests at OAEI campaign 2010. Results show that neural network boosts the performance in most cases, and that the proposed novel approach is competitive with top-ranked system.

Keywords: artificial neural network, training, ontology alignment, WordNet, XMap++.

1. Introduction

The Semantic Web community concedes that having a single domain ontology shared by a number of different applications may be not feasible, since data will inevitably come from myriad domains. Ontology, a formal, explicit specification of a shared conceptualization (Gruber, 1993), has been suggested as a way to solve the problem. For this reason, before being able to combine similar ontologies, a semantic and structural mapping between them has to be established. The process of establishing such a mapping is called ontology alignment. Mapping of heterogenous ontologies in different domains is not possible without knowing the semantic mappings between them. It will become increasingly significant as the semantic web evolves, it is already an active research area and

*Submitted: October 2012; Accepted: November 2012

several automatic or semi-automatic ontology alignment tools have been proposed (e.g., Melnik et al., 2001; Noy and Musen, 2000; Noy and Musen, 2001). Most tools rely on heuristics that detect some sort of similarity in the description of the concepts and the structure of the ontology graphs, by using e.g., string and graph matching techniques. Comprehensive surveys of the ontology mapping state of the art approaches can be found in Euzenat et al. (2004), Kalfoglou and Schorlemmer (2003), Euzenat and Shvaiko (2007). Though some of these approaches have made significant progresses in ontology mapping, they suffer from many limitations. (1) Ontology mapping approaches that use multiple mapping strategies encounter the problem of aggregating multiple similarities, implying manual parameter setting (Bernstein and Melnik, 2007; Falconer and Storey, 2007; Falconer et al., 2006); this tends to be impractical due to inability to adapt to different ontology mapping tasks. (2) The problem of such approaches is the difficulty of collecting sufficient training data that may itself incur a substantial effort like using some heuristic methods, which are often applicable in many domains. (3) A further problem is that even within a domain the successful configuration for one problem does not guarantee sufficient match quality for different problems, especially for matching large schemas. Therefore, one would need methods to preselect suitable and sufficient training correspondences for a given match task, which is an open challenge. To overcome the limitations, in this paper we propose a new automatic technique to train and generate the string, linguistic and structural weights. Furthermore, the burden of manual selection of weights has been definitely eliminated. In this paper, we have combined different weights of string-based, linguistic and structural categories into one input sample. The ensemble method is an active research area leading to better performance than a single classifier (matcher) (Kittler, 1998). Some studies have shown that using a single similarity measures performing well may not be the optimal choice (Tumer and Ghosh, 1996).

Therefore, the main contributions of this work are:

1. Our algorithm integrates the use of an Artificial Neural Network (ANN) to assign weights to various kinds of similarity measures so that they do not have to be given by a human (Djeddi and Khadir, 2013). The similarity measures involved take into account string, linguistic and structural information of ontology.
2. Moreover, our learning technique is carried out based on the ontology schema information alone, which distinguishes it from most other learning-based algorithms (Chortaras et al., 2005; Mao et al., 2010), relying in most cases on ontology instance. In order to avoid the problem of missing instance data (either in quality or in quantity), which is common for real-world ontologies, our weight learning technique is carried out at the schema level instead of the instance level.
3. The prospect of obtaining precise goal driven results, with the important parameter number will surely optimize ontology alignment (Shvaiko and

Euzenat, 2013). So we provide a comprehensive evaluation involving different matching tasks to provided empirical evidence about the flexibility of using an ANN.
4. We provide results following a standard benchmark to enable the comparison with other approaches.

The rest of this paper is organized as follows. Section 2 describes the related works for ontology alignment. Section 3 gives an overview of our approach, while Section 4 discusses the strategy in applying machine learning techniques in our algorithm. Section 5 reports on the experiments conducted and Section 6 analyzes the results. Finally, Section 7 concludes on the results.

2. Related work

Ontology matching (Euzenat and Shvaiko, 2007) is used for creating mappings between ontologies, where ontologies alignment enables the knowledge and data expressed in the matched ontologies to be interoperated. A major insight of Ontology Alignment Evaluation Initiative (OAEI) (Euzenat and Shvaiko, 2007), is that there is no definite method or system for all existing matching problems. During the past years, a lot of research has been devoted to developing highly sophisticated tools for performing ontology matching automatically (Euzenat et al., 2010). Those tools are able to produce high-quality mappings between ontologies, given that their parameters (such as weights and thresholds used to compute the mappings) are well tuned. Such tuning, however, is often complicated, since it involves setting of many parameters and requires a lot of detailed knowledge about the underlying algorithms and implementations.

Tuning framework provides an effective mechanism to automatically determine in what cases, a single strategy method should be used, and in what cases a combination method should be used. Adding to that it allows an expert knowledge engineer to predict what entity alignment strategy is most successful for a given pair of ontologies. Bellahcene and Duchateau (Bellahsene et al., 2011) provide an overview of recent approaches including tuning frameworks. Most previous approaches for automatic tuning apply supervised machine learning methods. They use previously solved match tasks in training to find effective choices for matcher selection and parameter settings such as similarity thresholds and weights to aggregate similarity values, see, e.g., Duchateau et al. (2009). CIDER (Gracia et al., 2011) uses two different neural networks for computing similarities between classes and properties, respectively. It extracts the ontological context of the compared terms and enriches it by applying lightweight inference rules. CIDER uses Jaro–Winkler measure to compute similarity between concept names after enriching each concept name with its WordNet synonyms. The PRIOR+ system (Mao et al., 2010) proposes a new weight assignment method to adaptively aggregate different similarities and then adopts a neural network based constraint satisfaction model to improve overall mapping performance from aggregated results. In Chortaras et al. (2005)

an automatic ontology alignment method based on a recursive neural network model that uses ontology instances to learn similarities between ontology concepts is proposed. Li et al. (2006) developed a system using Bayesian decision theory in order to generate an alignment between ontologies and additionally accepting user input to improve the mappings (RiMOM). GLUE (Doan et al., 2003) employs machine learning techniques that analyze the taxonomy and the information within concept instances of ontologies. The main problem of the aforementioned systems is that most of them use weights initially set up by a human operator. Instead, the proposed approach computes the weights in an automatic way, so the process can be more flexible in real scenarios.

3. Overview of our approach

3.1. Details of XMap++

XMap++ (eXtensible Mapping) is a system for ontology alignment that performs semantic similarity computations among terms of two given ontologies. In XMap++'s previous version (Djeddi and Khadir, 2011), we proposed a strategy selection method to automatically combine the matching strategies based on the weight of the linguistic affinity W_{LA}. This weight is calculated as given by

$$W_{LA} = \frac{linguistic\ similarity}{max(linguistic\ similarity, structural\ similarity)}. \quad (1)$$

In the current version, we integrate machine learning techniques, such that the weights W_{LA} can be learned from training examples, instead of being calculated from (1). We build a 3-dimensional vector for each concept, and each dimension records one semantic aspect, which represents a combination of different categories of similarity measures, such as string, linguistic and structural methods.

3.2. Feature selection

String-based, linguistic and structural methods are three different categories of measuring similarities in ontology alignment. Each method returns a similarity value in the range of [0, 1] for a given entity pair from two ontologies. These methods are briefly introduced in the following subsections.

3.2.1. String similarity

The string similarity methods compare the concepts textual descriptions associated with the nodes (labels, names, identity, etc.) of each ontology.

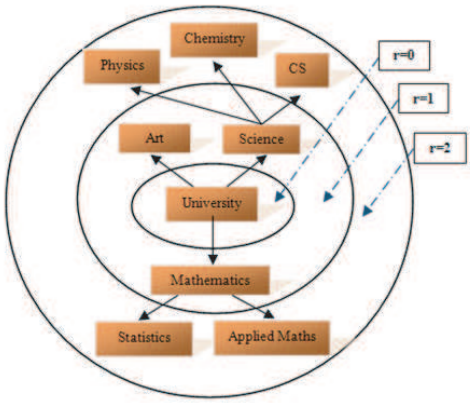


Figure 1. Sketch of ontology and the scope of the concept of University at different radiuses r

3.2.2. Linguistic similarity

WordNet (Fellbaum, 1998) is currently the most popular semantic resource in the computational linguistics community. A known problem of WordNet is that it is too fine-grained in its sense definitions (many classes in WordNet are very generic). For instance, it does not distinguish between homographs (words that have the same spelling and different meanings) and polysemes (words that have related meanings) (Jiamjitvanich and Yatskevich, 2009). Often the same word placed in different textual contexts assumes completely different meanings. In order to deal with lexical ambiguity, this approach introduces the notion of "scope" of a concept which represents the context where the concept is placed. In our approach, the similarity between two entities of different ontologies is evaluated not only by investigating the semantics of the entities names, but also taking into account the local context, through which the effective meaning is described. The context is the set of information (partly) characterizing the situation of some entity (Dey 2001). The notion of context is not universal but relative to some situation, task or application (Dourish, 2001; Chalmers, 2004). In particular, the neighborhood of a term (immediate parent and children in the "is a" hierarchy) may be especially important.

Fig. 1 sketches the idea. The scope defines an area composed of all concepts that are connected directly or indirectly to the central node. This area represents the context. Increasing the radius means enlarging the scope (i.e. this area) and, consequently, the set of neighbour concepts that intervene in the description of the context. The value of linguistic methods is added to the linguistic matcher or structure matcher in order to enhance the semantic ambiguity during the comparison process of entity names.

3.2.3. Structural similarity

Ontology alignment solely based on string and linguistic similarities may provide incorrect match candidates. Structural matching is used to correct such match candidates based on their structural context. The structural approach matches the nodes based on their adjacency relationships. The superclass-subclass relationships (subsumption relationships) that are frequently used in ontologies serve as the foundation of structural matching. XMap++ algorithm values the semantic relation between two concepts while taking in consideration the types of cardinality constraints and values between their properties (Djeddi and Khadir, 2011). OWL makes the distinction between two types of object proprieties, permitting to link instances to each other and type data properties permitting to link individuals to data values. An object propriety is an instance of the owl:ObjectProperty and a data type property is an instance of the owl:DatatypeProperty. Both classes are sub-classes of the RDF class: rdf:Property. The XMap++ algorithm values the semantic relation between two concepts while taking in consideration the types of cardinality constraints (e.g. OWLAllValuesFrom, OWLSomeValuesFrom , OWLMinCardinality, OWL-Cardinality, OWLMaxCardinality, Same_as or Kind_of) and values between their properties (e.g. OWLMaxCardinality >=1).

4. Machine learning method

Supervised machine learning methods are utilized to extract the optimal model of compound metrics, where the training algorithm is given a training set of inputs and the ideal output for each input. The feedforward neural network, or Multi-Layered Perceptron (MLP), is a type of neural network first described by Warren McCulloch and Walter Pitts in the 1940s. The feedforward neural network, with variants, is the most widely used form of neural network. It is often trained with the back propagation training technique, given the celebrated Multi Layered Perceptron (MLP). More advanced training techniques may be used, such as resilient propagation. The Resilient Propagation Training (RPROP) (Reidmiller and al., 1993) algorithm is usually the most efficient training algorithm provided by the framework Encog (Heaton, 2011) for supervised feedforward neural networks. One particular advantage of the RPROP algorithm is that it requires no parameter setting before usage. There are no learning rates, momentum values or update of constants that need to be determined. This is obviously a major advantage as it can be difficult to determine the exact learning rate that might be optimal.

4.1. Concept similarity matrix

To construct the similarity matrix, similarity measures (Section 3.2) are applied to a pair of ontologies selected from the data sets. Similarity matrix is a table

with M rows and N columns, where M is the number of given entity pairs and N is the number of applied features (similarity measures). Having provided the similarity matrix and target values, the problem would be reduced to a supervised learning task comprised of training and testing phases.

Definition 1. *Let α and β be respectively the concepts of the ontology O_1 and O_2. Let s_1, s_2 and s_3 the respectively the metrics of the string, linguistic and structural similarity. Let w_1, w_2 and w_3 the respectively the weights of the string, linguistic and structural similarity. After s_1, s_2 and s_3 between two concepts, α and β, are calculated, the similarity value s is obtained as the weighted sum of s_1, s_2 and s_3:*

$$s = \sum_{i=1}^{3} (w_i s_i) \tag{2}$$

Where $\sum_{i=1}^{3} w_i = 1$. Notice that w_i are randomly initialized and will be adjusted through a learning process (see Section 4.2 below).

For two ontologies being matched, O_1 and O_2, we calculate the similarity values for pairwise concepts. Then we build an $n_1 \times n_2$ matrix M_1 to record all values calculated, where n_i is the number of concepts in O_i.

4.2. MLP network design

We regard the hypothesis space in this learning problem as a 3-dimensional space consisting of w_1, w_2, and w_3, i.e., a set of weight vectors. Our objective is to find the weights that best fit the training examples. The neural network used corresponds to an MLP, which consists of multiple layers of nodes in a directed graph, each layer fully connected to the next one. Each connection (synapse) has an associated weight. In our particular situation, the network is composed of three layers: input, hidden, and output layer (with three, four, and one neuron respectively; additionally, two bias neurons are used in the input and hidden layers respectively). Sigmoid activation function scales the output from one layer before it reaches the next layer. We adopt a three-layer 3×1 network in XMap++, as shown in Fig. 2 .

The network input is a vector, which consists of s_1, s_2, and s_3, representing string, linguistic, and structual similarities respectively, for a given pair of concepts, where the network output is s, the similarity value between these two concepts as given by (2). To train the neural network, we must construct an object that contains the inputs and the expected outputs. To construct this object, we must create two arrays. The first array will hold the input values for the neural network. The second array will hold the ideal outputs for each corresponding input values. These will correspond to the possible values for XMap++. Now that the training set has been created, the neural network can

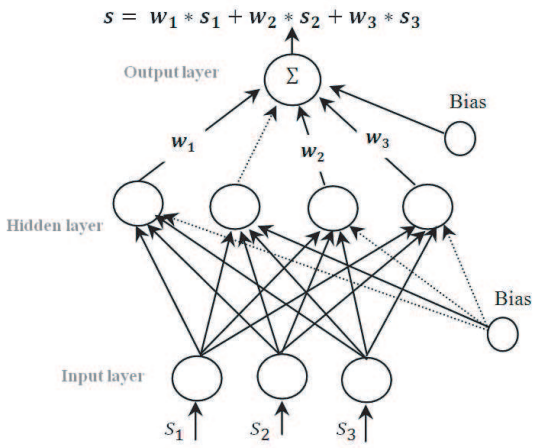


Figure 2. Neural network structure

be trained. Training will continue for many iterations, or epochs, until the error rate of the network is below an acceptable level (e.g. error = 0.1). For this example we are going to use Resilient Propagation (RPROP). Once the neural network has been trained, it is ready for use.

5. Experiments and evaluation criteria

Given the experimental nature of the implementation, it is feasible to set up a series of hypotheses which, in turn, will assist in the derivation of strategies for network training and testing. The proposed hypotheses are summarized below:

1. Cross validation will help inhibit the neural networks overfitting during training.
2. As cross-validation can monitor the error variation during the training process it can reduce the risk of overfitting by stopping training.

These hypotheses have led to the derivation of testing strategies, which should determine the most suitable network variables (i.e. weight and biases or thresholds). Network performance is analysed in terms of Mean-Squared Error (MSE), the average difference between actual output and desirable output. For each item of training data, some change to the weight matrix and thresholds will be calculated and applied in batches. Batch learning with 10 hidden units carried out as a strategy throughout the rest of the experiment. Batch learning is the process of calculating the weight change (δw) with respect to the error, for the entire epoch. The new weights are calculated at the end of the training epoch.

5.1. Implementation and setting

XMap++ is implemented as a Protege[1] PlugIn and is tested under Protege 3.4. Moreover, the parameters taken by the approach (i.e. weights, thresholds and the radius value) were tuned and set depending on the type of information contained in the ontologies to be mapped. The thesaurus WordNet[2] (version 2.1) is optionally used to calculate the lexical similarities between each pair of concepts and properties, in order to derive semantic similarity measures. Finally, to create and manipulate neural networks we use the framework Encog[3].

5.2. Data sets and evaluation criteria

To evaluate our approach we have applied the benchmark tests from OAEI ontology matching campaign 2010. This test set consists of one reference ontology O_R (33 classes, 59 properties, 56 individuals and 20 anonymous individuals), for a bibliographic domain, to be compared with other test ontologies O_T. Some introduced changes include, for example, the extension, or shrinkage of the ontology hierarchy, the use of synonyms, foreign names, removal of class properties and many more. The benchmark tests can be divided into five groups as shown in Table 1.

Table 1. Overview of OAEI benchmark tests.

Tests	Description
♯ 101-104	O_R and O_T have the same representation and conceptualisation
♯ 201-210	O_R and $O_T$have the same structure but different linguistic
♯ 221-247	O_R and $O_T$have the same linguistics but different structure
♯ 248-266	Both structure and linguistics are different between O_R and O_T
♯ 301-304	O_T are real-life bibliographic ontologies

We adopt the evaluation criteria used by the campaign. That is, standard evaluation measures precision, recall and f-measure will be computed against the reference alignments. These measures are defined as equations (3), (4) and (5):

$$Precision = \frac{|alignment\ given\ \cap correct\ alignment|}{|alignment\ given|} \tag{3}$$

$$Recall = \frac{|alignment\ given\ \cap correct\ alignment|}{|correct\ given|} \tag{4}$$

[1] http://protege.stanford.edu/
[2] http://wordnet.princeton.edu/
[3] http://www.heatonresearch.com/encog

$$F - measure = \frac{2 \times Precision \times Recall}{(Precision + Recall)} \tag{5}$$

5.3. N-fold cross validation

In this approach, we randomly partition the training data into N sets of equal size and run the learning algorithm N times. Each time, a different one of the N sets is deemed the test set, and the model is trained on the remaining $N-1$ sets. The cross-validation process is then repeated N times (the folds), with each of the N sets used exactly once as the validation data. The $N\text{-}fold$ cross-validation gives an indication of how well the learner will do when it is asked to make new predictions for data it has not already seen. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. The result will be the average of the result of each fold (McLachlan et al., 2004). When compared with the actual error, the error given by cross-fold validation is greater than that given by the network (see Fig. 3). This is possibly due to the data being skewed, though this would have to be proven with an accurate plot of all elements in the test set. It shows that the lowest error is given by 100 folds. A 100 fold cross validation scheme is the most commonly used technique for producing the most reliable validation error. The training data is randomly split into 100 separate parts. The average of the 100 validation errors is a good overall error estimate.

5.4. Overfitting

The algorithm is trained with 4 ontologies: (♯101 against ♯223), (♯101 against ♯302), (♯101 against ♯240) and (♯101 against ♯260). Figs. 4a-d show the decreas-

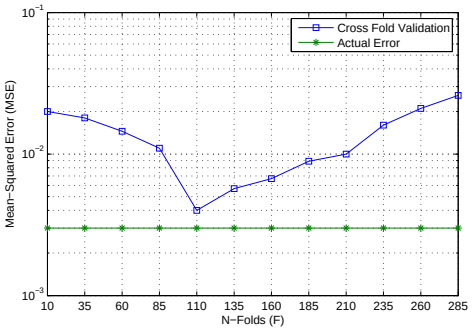
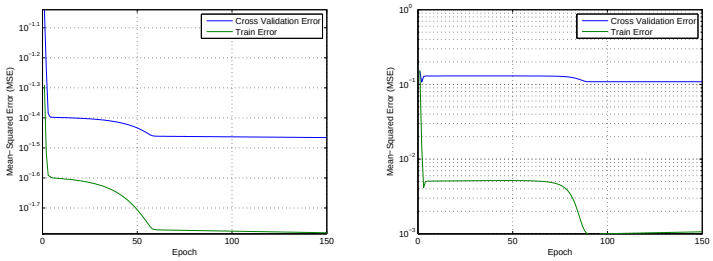
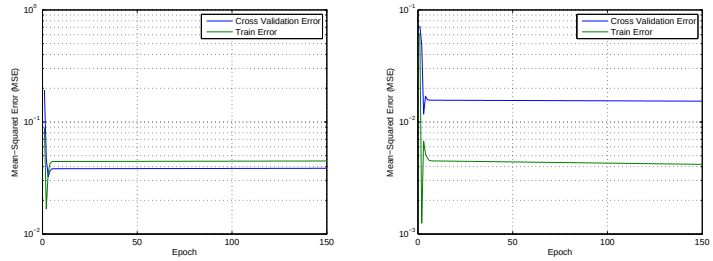


Figure 3. 10-fold to 285-fold cross validation.



(a) Error plot for tests ♯101 and ♯223 (b) Error plot for tests ♯101 and ♯302



(c) Error plot for tests ♯101 and ♯240 (d) Error plot for tests ♯101 and ♯260

Figure 4. The four ontologies used to train the ANN

ing MSE during training. Both errors keep decreasing, indicating that a state of overfitting has not been reached yet. However, convergence slowly drops to a rate in which performance improvement is negligible. In the latest epochs the MSE for each training test decreases with $-4.3411e-009$. Good performance on the test set indicates a representative training set. In this case, training can be stopped when the training error has reached a sufficiently small value or when changes in the training error remain small. There is no chance that it will reach lower levels at a later stage if training were continued.

Having conducted an exhaustive series of tests throughout this experiment, it is possible to compare the obtained results with the initial hypotheses set in Section 5. Initially, it was correct to assume that the process of trade-offs between accuracy (in error reduction), complexity and computational time was an interesting tuning problem to tackle. We believe that the strategy which we executed was almost optimal for this particular dataset. However, if the network is tested on a completely different dataset, it is likely that the training approach may have to be altered.

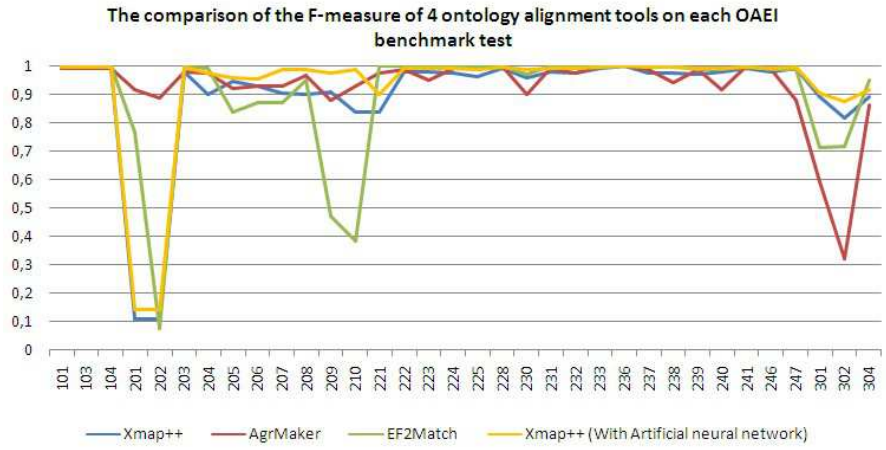


Figure 5. The comparison of the F-measure of four ontology alignment tools on each of the OAEI benchmark test

6. Experimental design and results

6.1. The improvement induced by artificial neural network in XMap++

Here, two experiments have been conducted. First experiment addresses an aspect which has its impact on the training model and the second focuses on testing the training dataset.

1. First experiment: The first experiment has simply chosen the optimum model based on string, linguistic and structural similarity measures, from Section 3.2. The value of the weights w_1, w_2, w_3 and the learning rate η are initialised randomly by the RPROP method. The obtained similarity matrix is then aggregated via classification (string, linguistic and structural matcher). After the adjustment of classifiers' parameters (threshold value, radius value), the training model was obtained.
2. Second experiment: This experiment explores the effect of training samples quantity on the final trained model quality. In this experiment, the number of entity pairs is increased by using other ontologies such as tests ♯102 and ♯103, i.e. entity pairs extracted from (♯101, ♯102) and (♯101, ♯103). The diversity of instances in training phase is, therefore widened. To avoid training the model with similar input samples, those samples from ♯102 and ♯103 ontologies which represent the highest variances are selected.

As shown in Fig. 5, tests ♯103 and ♯104, reach a full recall, because XMap++ is tested with simple and similar names. For ♯201 and ♯202, XMap++ is not good at precision and recall, both down to 0.11 and 0.12 for XMap++ (with

Table 2. The improvement induced by neural networks on XMap++ results

Alignment Tools	Precision	Recall	F-Measure
XMap++	0,9	0.89	0.89
N-XMap++	0.93	0.92	0.92
AgrMaker	0.96	0.91	0.92
EF2Match	0.98	0.86	0.89

artificial neural networks termed from now on N-XMap++), because the names of classes/properties have been "removed".

The ♯204 test is a naming conventions test, and the result is encouraging for XMap++ (or N-XMap++); the proposed technique gives good precision, the low recall is due to using some shortcuts which are not included in the dictionary such as 'MScthesis' with 'MasterThesis'. Test ♯205 is a synonyms test, the two implemented systems shows good results based on the retrieved synonyms from WordNet, in which, for example, 'frequency' and 'periodicity' are matched. In fact, concerning the test case ♯205, the high scale of the recall is explained through the searching for WordNet synonyms based not only on the full labels of entities but also based on the context where the entities are placed which reduce the incorrect correspondences generated by the system. Tests (♯221, ♯222, ♯223, ♯224, ♯225 and ♯228) achieve full recall and precision. These tests use same string properties with different structure representation and eliminations. ♯230 is an expansion of class components and string properties test, and the result gives good precision and good recall. The proposed technique resolves some matching difficulties such as matching 'Institution' with 'institutionName', 'Organization' with 'organizationName' and 'Journal' with 'JournalName'. Series (♯301-304) represent real-life ontologies modeled by different institutions but for the same domain of bibliographic metadata. In these tests, precision ranges from 0.85 to 0.9 and recall stays between 0.75 and 0.9. XMap++ just can find equivalent alignment relations. However, the inclusion alignments cannot be generated. For ♯301 and ♯302, XMap++ finds most correct alignments, but it also returns some wrong results. The alignment results for ♯303 are far from satisfactory. The reason might be that test ♯303 has no individuals and a shallow class hierarchy, and there are no direct connections between the classes and properties. Moreover, it is clear that the two systems are efficiently processing tests ♯301, ♯302 and ♯304. Thus, this proves that the semantically context based-approach developed in XMap++ is appropriate for real alignment. As observed from the Table 2, the proposed system has stability characteristics with the different types of tests (except those that have no labels).

As shown in Table 2, a direct comparison between the XMap++ without ANN, and XMap++ with ANN, shows that the addition of ANN does not affect negatively the algorithm performances but, on the contrary, leads to slightly

better results. Such results indicate also that the new approach leads to a better recall, at the cost of precision. This translates in a percentage improvement, of 3%, for precision, recall, and f-measure.

6.2. Comparison between N-XMap++ and top ranked systems

We have chosen the alignments generated by the two best matchers that have participated in the 2010 OAEI conference track (Euzenat et al., 2010): AgrMaker and Eff2Match. Table 2 shows that the overall f-measure of XMap++ with ANN (0.92) is competitive when compared to the performances of AgrMaker (0.92) and EF2Match (0.89). Yet, precision and recall of AgrMaker and EF2Match are slightly superior to our algorithm, because they perform perfectly on tests ♯201 and ♯202. Comparing N-XMap++, EF2Match and AgrMaker in terms of performance, as it is shown in Fig. 5 and when considering tests ♯203 to ♯301, we found that our results are most of the time superior or equal to EF2Match and AgrMaker results.

7. Conclusions and future work

In this paper we proposed a novel neural network based approach to automatically find the best manner of aggregating different similarity measures into a single similarity metric. The proposed approach is then exploited to learn the MLP weights for different semantic ontologies scenarios (i.e., string, linguistic and structural aspect). This, in turn, increases the discrimination ability of the model and enhances the overall accuracy of the system. Therefore, we tackle the difficult problem of carrying out machine learning without help from instance data as well as skipping over the use of human heuristics and/or domain knowledge to predefine the weights of different categories. We explain and analyze our algorithm in detail, and our experimental results on OAEI (2010) benchmark tests show that the approach is promising and constitutes a starting point for future explorations with the use of ANN for computing similarities. XMap++ adopts vectors to record semantic aspects, so it is not difficult to handle if more relationships are to be taken into consideration. What needs to be done is for us to expand the current vectors into more dimensions to hold more semantic aspects like adding the similarity measure taking into account the instance information of ontology. An ANN with multiple layers might be necessary in this case. We are also interested in extending our algorithm using Genetic Algorithms (GAs) to obtain the optimal topology for our ANN.

References

Bernstein, P.A. and Melnik, S. (2007) Model management 2.0: Manipulating richer mappings. In: *SIGNMOD'07. Proc. of the 2007 ACM SIGMOD International Conference on Management of Data.* ACM Press,

New York, 1-12.

Bellahsene, Z., Bonifati, A. and Rahm, E., eds. (2011) *Schema Matching and Mapping. Springer ++Data–Centric Systems and Applications Series.*

Chalmers, M. (2004) A Historical View of Context. *Computer supported cooperative work* **13**(3), 223–247.

Chortaras, A., Stamouet, G. B., and Stafylopatis, A. (2005) Learning Ontology Alignments Using Recursive Neural Networks. *In: Proceedings of the 15th international conference on ANN: formal models and their applications*, Part II. Springer, 811–816 .

Dey, A., D., Salber and Abowd, G. (2001) A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *J. Human-Computer Interaction (HCI)* **16**(2-4), 97–166.

Djeddi, W. and Khadir, M. T. (2011) A Dynamic Multistrategy Ontology Alignment Framework Based on Semantic Relationships using WordNet.In: *Proceedings of the 3rd International Conference on Computer Science and its Applications (CIIA'11)*, December 13-15, Saida, Algeria. CEUR–WS.org, 149–154.

Djeddi, W. and Khadir, M. T. (2012) Introducing Artificial Neural Network in Ontologies Alignement Process. *New Trends in Databases and Information Systems Advances in Intelligent Systems and Computing* **185**, 175–186.

Doan, A., Madhaven, J., Dhamankar R., Domingos P. and Halevy, A.Y. (2003) Learning to match ontologies on the semantic web. *The International Journal on Very Large Data Bases* **12** (4), 303–319.

Dourish, P. (2001) Seeking a foundation for context-aware computing. *J. Human-Computer Interaction (HCI)* **16**(2-3).

Duchateau, F., Coletta, R., Bellahsene, Z. and Miller, R. J. (2009) (Not) yet another matcher. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009.* ACM Press, 1537–1540.

Euzenat, J., Bach, T. et al. (2004) State of the art on ontology alignment. *Knowledge web NoE.* Technical Report, deliverable 2.2.3. Statistical Research Division, Bureau of the Census, Washington.

Euzenat, J., Ferrara, A., Meilicke et al. (2010) Results of the Ontology Alignment Evaluation Initiative 2010. In: *Proceedings of the Fifth International Workshop on Ontology Matching (OM-2010).* CEUR-WS **689**.

Euzenat, J. and Shvaiko, P. (2007) *Ontology Matching.* Springer, Berlin, Heidelberg, New York.

Falconer, S.M. and Storey, M. (2007) Cognitive support for human-guided mapping systems. Technical Report DCS-318-IR, University of Victoria, Victoria, BC, Canada.

Falconer, S.M., Noy, N.F. and Storey, M. (2006) Towards understand-

ing the needs of cognitive support for ontology mapping. In: *International Workshop on Ontology Matching*, ISWC 2006. CEUR-WS **225**.

Fellbaum, C. (1998) WordNet: An Electronic Lexical Database. MIT Press, Cambridge MA.

Gracia, J., Bernad, J. and Mena, E. (2011) Ontology Matching with CIDER: Evaluation Report for OAEI 2011. In: *Proceedings of 6th Ontology Matching Workshop (OM'11), at 10th International Semantic Web Conference (ISWC'11)*, Bonn (Germany). CEUR-WS **814**.

Gruber, T. (1993) A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* **5**(2), 199–220.

Heaton, J. (2011) Programming Neural Networks with Encog3 in Java. 2nd ed. Heaton Research, Chesterfield.

Kittler, J., Hatef, M., Duin, R.P.W. and Matas, J. (1998) On Combining Classifiers. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, **20**, 226–239.

Jiamjitvanich, K. and Yatskevich, M. (2009) Reducing polysemy in WordNet. In: *Proceedings of the 4th International Workshop on Ontology Matching(OM-2009).* CEUR–WS.org, 260–261.

Kalfoglou, Y. and Schorlemmer, M. (2003) Ontology mapping: the state of the art. *Knowledge Engineering Review* **18**(1), 1–31.

Li, Y., Li, J.Z., Zhang, D. and Tang, J. (2006) Result of Ontology Alignment with RiMOM at OAEI'06. *Ontology Matching. CEUR Workshop proceedings*, **225**. CEUR–WS.org

Mao, M., Peng, Y. and Spring, M. (2010) An Adaptive Ontology Mapping Approach with Neural Network based Constraint Satisfaction. *Journal of Web Semantics* **8**(1), 14–25.

McLachlan, G. J., Do, K. A. and Ambroise, C. (2004) Analyzing microarray gene expression data. In: *Wiley Series in Probability and Statistics.* Wiley-Interscience, New Jersey.

Melnik, S., Garcia-Molina, H. and Rahm, E. (2001) Similarity flooding: A versatile graph matching algorithm and its application to schema matching. *Proceedings of ICDE*, IEEE Computer Society, Washington, DC, 117–128.

Noy, N. and Musen, M.A. (2000) PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. *In: Proceedings of the 7th National Conference on Artificial Intelligence.* AAAI Press, 450–455.

Noy, N. and Musen, M.A. (2001) Anchor PROMPT: Using Non-Local Context for Semantic Mapping. In: B. Nebel, ed., *Proc. of the 17th International Joint Conference on AI, IJCAI 2001.* Morgan Kauffmann, Seattle, Washington, 63–70.

Reidmiller, M. et al. (1993) A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm. *In: Proceedings of the IEEE International Conference on Neural Networks (ICNN).* IEEE Press, 586–591.

Shraiko, P. and Euzenat, J. (2013) Ontology Matching: State of the art and future challenges. *IEEE Trans.on Knowledge and Data Engineering*, **25** (1), 158–176.

Tumer, K. and Ghosh, J. (1996) Classifier Combining: Analytical Results and Implications. *In: Proceedings of the 13th National Conference on Artificial Intelligence*. AAAI Press.