MULTISENSOR

Mining and Understanding of multilinguaL contenT for Intelligent Sentiment Enriched coNtext and Social Oriented inteRpretation

FP7-610411

D4.2

Mapping discovery and validation

Dissemination level:	Public
Contractual date of delivery:	Month 20, 30/6/2015
Actual date of delivery:	Month 20, 30/6/2015
Workpackage:	WP4 Multidimensional Content Integration and Retrieval System
Task:	T4.2 Mapping discovery and validation T4.3 Content alignment and integration
Type:	Report
Approval Status:	Draft
Version:	1.0
Number of pages:	42
Filename:	D4.2_MappingDiscoveryValidation_2015-06-30_v1.0.pdf

Abstract

This Deliverable reports the developed methodology of the mapping discovery between ontologies and on the approaches used for the validation of these mappings. A novel late fusion algorithm for combining the different ontology matching algorithms is presented and an ontology matching algorithm based on visual features is presented, while also presenting the methods for validating the consistency of mappings. The Deliverable gives an overview of the developed software tool and user interface for ontology matching and reports on the initial study for content alignment and integration of the MULTISENSOR knowledge base and

on the development of a baseline methodology for content alignment.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	04/05/2015	Initial draft and table of contents	A. Moumtzidou (CERTH)
0.2	01/06/2015	First version with draft content	A. Moumtzidou, D. Liparas (CERTH)
0.3	16/06/2015	Second iteration with Section 4 completed	V. Alexiev (ONTO)
0.4	20/06/2015	Review ready version	A. Moumtzidou, S. Vrochidis (CERTH)
0.5	25/06/2015	Internal review	B. Simeonov (ONTO)
1.0	30/06/2015	Final version	A. Moumtzidou (CERTH)

Author list

Organization	Name	Contact Information
CERTH	Stefanos Vrochidis	stefanos@iti.gr
CERTH	Dimitrios Liparas	dliparas@iti.gr
ONTO	Vladimir Alexiev	vladimir.alexiev@ontotext.com
CERTH	Anastasia Moumtzidou	moumtzid@iti.gr

Executive Summary

In Deliverable 4.2 the progress of Tasks 4.2 and 4.3 of WP4 for the development of ontology and content alignment techniques respectively, are being presented. The position of the ontology and content alignment modules in the processing pipeline of MULTISENSOR is given and the interactions with other modules are presented. Section 3 gives an overview of the state of the art in ontology alignment and presents a novel approach for ontology alignment using visual features, in cases where ontological concepts can be represented using visual content, e.g. images. Section 3 continues with a presentation of an alignment fusion approach for combining separate results of ontology matching algorithms in order to produce a single alignment list, and the approach followed for the semantic validation of matching results is presented. Section 4 presents the progress regarding content alignment where the module for validating and integrating RDF content in the MULTISENSOR knowledge base and the baseline module for discovering related content are described. Finally, Section 5 concludes the deliverable.

Abbreviations and Acronyms

CAP	Content Alignment Pipeline
CEP	Content Extraction Pipeline
CIDOC-CRM	CIDOC Conceptual Reference Model
EDM	Europeana Data Model
IEEE	Institute of Electrical and Electronics Engineers
IST	Information Society Technologies
JPEG	Joint Photographic Experts Group
MPEG	Moving Picture Experts Group
OAEI	Ontology Alignment Evaluation Initiative
OWL	Ontology Web Language
RDF	Resource Definition Framework
W3C	World Wide Web Consortium
XML	eXtensible Markup Language
SPARQL	SPARQL Query Language
SWRL	Semantic Web Rule Language

Table of Contents

1 INTRODUCTION

This Deliverable targets two of the research challenges of MULTISENSOR, which are included in WP4. These challenges correspond to Tasks 4.2 “Mapping discovery and validation” and 4.3 “Content alignment and integration”.

For T4.2, MULTISENSOR focuses on the development of new ontology alignment algorithms for concept matching, in order to support the definition of the ontology framework in WP5. This task facilitates the extensibility of the semantic representation framework by supporting the semi-automatic integration of additional ontological resources. Ontology alignment is the process where two ontologies that usually describe the same knowledge domain are ‘aligned’, i.e. the correspondences between their classes and relations in terms of meaning are identified. The alignment between the ontologies is realized by using appropriate matching algorithms that rely on different features such as string similarity or lexical and structural similarity for producing matching pairs. In the proposed alignment approach we exploit features that haven’t been considered so far in the ontology matching research area, i.e. visual features, and the development of a novel fusion matching algorithm for combining matching scores of different ontology alignment algorithms. In MULTISENSOR, the final ontology mapping lists are selected after a semantic verification of an initial matching list.

For T4.3, the work carried out involves two directions. The first direction, involves the integration of information so the work that has been developed regarding the validation and cleaning of the RDF data that come from the Content Extraction Pipeline (CEP) of MULTISENSOR is presented. The second direction involves the development of a baseline version of content alignment for the discovery of relevant content from user-initiated queries. The integrated module in the Content Alignment Pipeline (CAP) is presented. CEP and CAP are both explained in detail in D7.2 and D7.4.

The document is structured as follows: In Section 2 a short overview of MULTISENSOR’s architecture is realized with a focus on how the aforementioned modules fit in it. In Section 3 the content alignment and validation modules are presented. Section 4 presents the work regarding content alignment and integration while Section 5 concludes the Deliverable.

2 ARCHITECTURE

Section 2 gives an overview of the modules regarding their position in the MULTISENSOR processing pipeline and of a high level architecture of the modules that have been developed for T4.2 and T4.3 respectively.

2.1 Description

Ontology alignment (Task 4.2) is run as a separate processing module, which is independent from the Content Extraction Pipeline (CEP) that has been described in Deliverable 7.2. Ontology alignment operates on the knowledge base of MULTISENSOR at the ontology structure level in order to identify equality relations (in classes and properties) between the different ontologies that are integrated in the knowledge base. The usage of the ontology alignment module is not continuous and shouldn't be run as a service. The alignment between ontologies should be performed at the initial stage of the knowledge base deployment when all ontologies are being integrated and after that, in cases where new ontologies are added to the knowledge base, it should be performed for identifying any relations between the new and the existing ontologies. The identified relations are added in the knowledge base, e.g. as new equivalence statements in case of equivalence relations, so that the reasoning engine can make use of them and link content. Since ontology alignment is not part of the CEP, it is implemented as a separate and standalone software module. The validation of the identified mappings has to do with ensuring that the identified mappings do not cause inconsistencies in the knowledge base, i.e. not asserting statements that cause conflicts between entities and content.

Content alignment and integration (Task 4.3) involves the validation of RDF content being stored in the MULTISENSOR knowledge base and the development of the Content Alignment Pipeline (CAP), which has been described in Deliverable 7.2. The validation of RDF content performs a syntactical sanitization action on the RDF content that is generated at the end of the CEP for each content item. It assures that the content that is generated meets the RDF requirements to be stored in the knowledge base and not cause exceptions. The CAP involves the discovery of links and inconsistencies between content items that is stored in the knowledge base. In this Deliverable, the baseline module for content alignment, which involves a query-based methodology to discover relevant content based on user input, is described. The final version of the CAP will adhere to the description that was presented in D7.2 and will be able to be executed periodically and discover content relations and inconsistencies within the whole knowledge base without user interaction.

2.2 WP4 modules and pipelines

The module and the pipeline that have been described in D7.2 and are the subject of this deliverable are the "Mapping discovery and validation" module, and the "Content alignment pipeline".

The "Mapping discovery and validation" module is independent of the CEP and runs separately as a standalone application. It takes as input two ontologies, where usually these ontologies describe the same or similar domains, and produces a set of proposed mappings between the ontologies classes and properties. The process of identifying these mappings is automatic so the user interaction is limited to parameters configuration. This mapping set can be accepted as is or it can be verified by the user, who is typically a domain expert. For the development of the module, the following research and technical activities have been carried out: a) Research and development of a novel ontology alignment algorithm which makes use of visual similarity features, b) research and development of a novel late fusion algorithm for combining separate ontology matchers results using adaptive weighting, c) development of sematic mapping validation techniques, d) development of a user interface for ontology alignment. The process of ontology alignment is displayed in Figure 1.

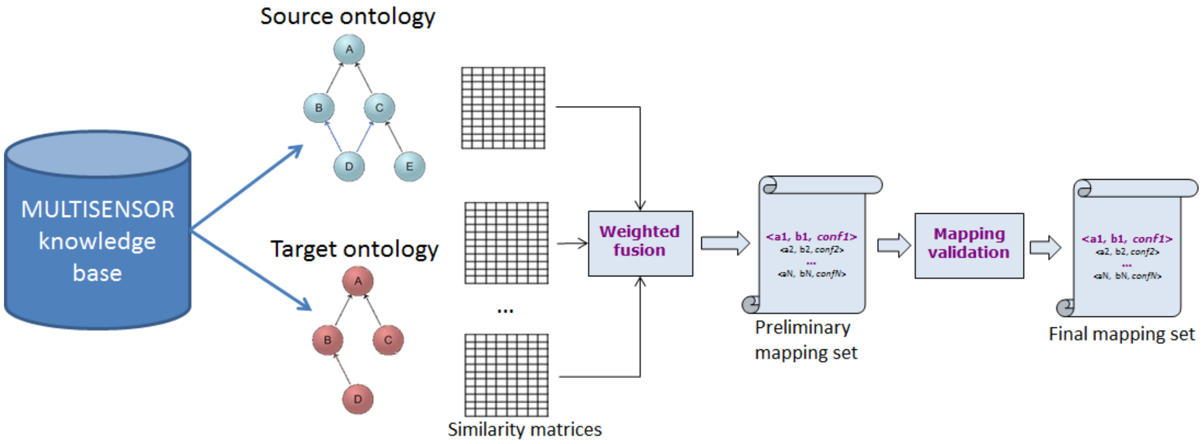


Figure 1: Basic workflow for aligning a source to a target ontology. The ontologies are taken from the MULTISENSOR knowledge base

For the "Content alignment pipeline" two different modules have been developed, a) the "RDF validation and cleaning" module for validating the correctness of the RDF content that is generated during the CEP processing. This module is integrated as validation stage in CEP and performs tests for the integrity of the data before they are stored in the MULTISENSOR RDF repository (Figure 2) and, b) the actual "Content alignment pipeline" (CAP) that performs on the knowledge base in order to discover links, relations, contradictions and inconsistencies in the content itself. Related content items could for example be defined based on spatiotemporal features, e.g. events that take place at the same or a nearby location, or even based on common named entities, e.g. news articles that mention the same company. Inconsistencies and contradictions can be identified based on e.g. articles describing the same event but having opposite sentimentality. In the current deliverable, the baseline module for content alignment has been implemented as at the time of writing the knowledge base wasn't populated with content in order to perform a study of the RDF content and develop the processing methods. The baseline module discovers relevant content according to user input using a SPRQL query-based approach and it is run when the user performs search. The next version of content alignment will be developed in accordance to what has been described in D7.2 (Figure 3).

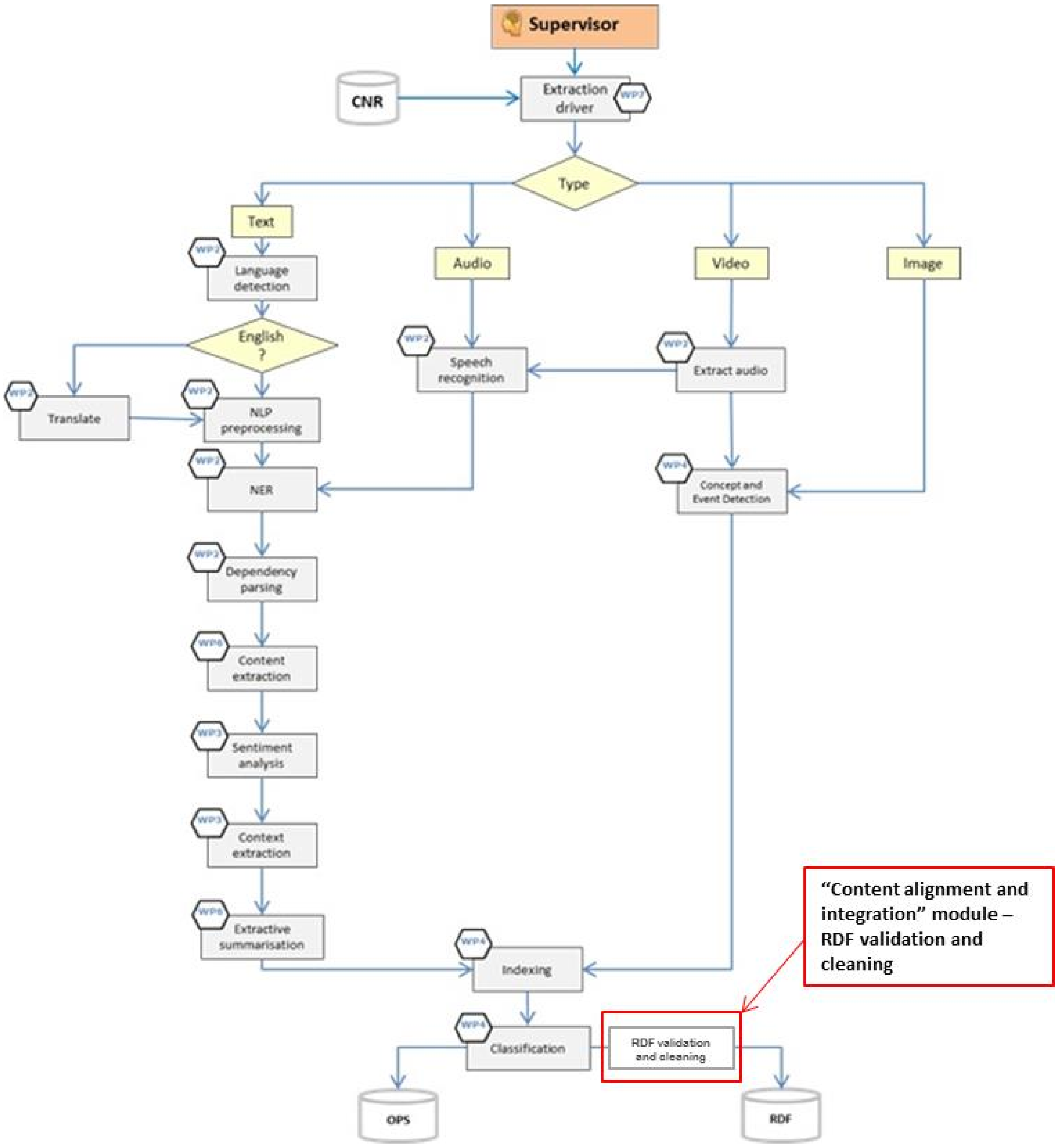


Figure 2: Position of the "RDF cleaning and validation" module in the CEP

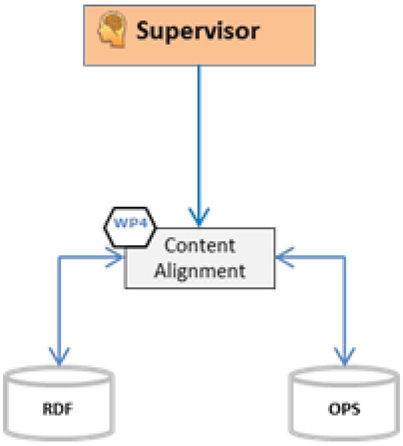


Figure 3: The "Content alignment pipeline" as it has been documented in D7.2

3 MAPPING DISCOVERY AND VALIDATION

The mapping discovery and validation module works directly on the MULTISENSOR knowledge base in order to assist domain expert users to identify relations between concepts (classes and properties) of the ontologies that have been integrated in the knowledge base. Ontology mapping in MULTISENSOR will help identify content that is semantically annotated using resources from different ontologies, such as DBpedia, PROTON, or other domain specific ontologies to be aligned. For example, if an entity has been defined as being of type "dbpedia-owl:Company" in DBpedia then through the mappings between DBpedia and PROTON, the entity should also be automatically defined as being of type "pext:Company" using the statement:

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/Company>
PREFIX pext: <http://www.ontotext.com/protonext>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

dbpedia-owl:Company owl:equivalentClass pext:Company
```

where the classes "dbpedia-owl:Company" and "pext:Company" are declared as equivalent classes. A reasoner can then deduce that instances of one class are also instances of the other class thus being able for example for queries to return instances of both classes.

In MULTISENSOR, since the number of ontologies that are integrated is large, the process of defining mappings should be aided by an automated method for mapping discovery. At the end though, all discovered mappings will have to be verified by a domain expert since a wrong mapping between classes can result in content inconsistencies.

For the reasons above, in MULTISENSOR for the mapping discovery and validation process the following research and development activities have taken place:

a) A new algorithm for visual-based ontology matching has been developed. The algorithm makes use of imaging sources such as ImageNet[1] in order to acquire appropriate images and construct a visual representation of an ontology's resources, i.e. its classes and properties. Low level visual features are extracted and these are used to compare the ontological resources and compute their similarity. The new algorithm has been tested in ontology alignment benchmark datasets and has shown promising results, especially if it is combined with other similarity measures such as lexical-based measures. In addition, a new fusion algorithm has been developed which makes use of an adaptive weighting scheme in order to adjust the different weights that are assigned to the similarity measures scores. Finally a methodology to validate the identified ontology mappings based on semantic verification has been developed.

b) In order to assist domain experts for defining the ontology mappings an intuitive user interface has been implemented. Through the user interface the user has the ability to combine the matching algorithms, have an overview of the matching results, perform edit operations (insert new mappings, delete mappings) and save them in different formats.

[1] ImageNet, http://www.image-net.org

All the above are described in the following sections.

3.1 State of the art in ontology matching

Semantic Web is providing shared ontologies and vocabularies in different domains that can be openly accessed and used for tasks such as semantic annotation of information, reasoning, querying, etc. The Linked Open Data (LOD) paradigm shows how the different exposed datasets can be linked in order to provide a deeper understanding of information. As each ontology is being engineered to describe a particular domain for usage in specific tasks, it is common for ontologies to express equivalent domains using different terms or structures. These equivalences have to be identified and taken into account in order to enable seamless knowledge integration. Moreover, as an ontology can contain hundreds or thousands of concepts, there is a need to automate this process. An example of the above comes from the cultural heritage domain where two ontologies are being used as standards, one is the CIDOC-CRM[2], used for semantically annotating museum content, and the other is the Europeana Data Model[3], which is used to semantically index and interconnect cultural heritage objects. While these two ontologies have been developed for different purposes, they are used in the cultural heritage domain and correspondences between their concepts should exist and be identified.

In ontology alignment the goal is to automatically or semi-automatically discover correspondences between the entities (classes, properties or instances) of ontologies. An 'alignment' is a set of mappings that define the similar entities between two ontologies. These mappings can be expressed e.g. using the *owl:equivalentClass* or *owl:equivalentProperty* properties so that a reasoner can automatically access both ontologies during a query.

While the proposed methodologies in literature have proven quite effective, either alone or combined, in dealing with the alignment of ontologies, there has been little progress in defining new similarity metrics that take advantage of features that haven't been considered so far. In addition existing benchmarks for evaluating the performance of ontology alignments systems, such as the Ontology Alignment Evaluation Initiative[4] (OAEI) have shown that there is still room for improvement in ontology alignment algorithms.

3.1.1 Algorithms for ontology matching

In order to accomplish the automatic discovery of mappings, numerous approaches have been proposed in the literature, that rely on various features. Of the most common are methods that compare the similarity of two strings, e.g. comparing *hasAuthor* with *isAuthoredBy*, are the most used and fastest to compute as they operate on raw strings. Existing string similarity metrics are being employed, such as Levenshtein distance, Edit distance, Jaro-Winkler similarity, etc, while string similarity algorithms such as (Stoilos et al., 2005) have been developed especially for ontology matching. Other mapping discovery methods rely on lexical processing in order to find synonyms, hypernyms or hyponyms

[2] CIDOC-CRM, http://www.cidoc-crm.org

[3] Europeana Data Model, http://labs.europeana.eu

[4] OAEI, http://oaei.ontologymatching.org

between concepts, e.g. comparing *Author* and *Writer*, where Wordnet is most commonly used. In (Lin and Sandkuhl, 2008) a survey on methods that use Wordnet (Miller, 1995) for ontology alignment, is carried out. Approaches for exploiting other external knowledge sources have been presented (Sabou et al., 2006; Faria et al., 2013; Perquita et al., 2014; Chen at al., 2014). Other similarity measures rely on the structure of the ontologies, such as the Similarity Flooding (Melnik et al., 2002) algorithm that stems from the relational databases world but has been successfully used for ontology alignment, while others exploit both schema and ontology semantics for mapping discovery. A comprehensive study of such methods can be found at (Shvaiko and Euzenat, 2005).

3.1.2 Matching systems and benchmarks

In terms of matching systems, numerous approaches that combine matching algorithms or include external resources for the generation of a valid mapping list between ontologies have been proposed. Most available systems have been evaluated in the OAEI benchmarks that are held annually. In (Jean et al., 2009) the authors use a weighted approach to combine several matchers in order to produce a final matching score between the ontological concepts. In (Ngo and Bellahsene, 2012) the authors go a step further and propose a novel approach to combine elementary matching algorithms using a machine learning approach with decision trees. The system is trained from prior ground truth alignments in order to find the best combination of matchers for each pair of concepts. Other systems, such as AML (Faria et al., 2013) and (Kirsten et al., 2011), make use of external knowledge resources or lexicons to obtain ground truth structure and concept relations. This is especially used when matching ontologies in specialized domains such as in biomedicine.

The abundance of ontology matching systems and algorithms coupled with the importance of the ontology matching in the Linked Data world has led to the establishment of the Ontology Alignment Evaluation Initiative (OAEI) and the publishing of benchmarks in order to objectively measure the performance of the multitude of ontology matching systems. The OAEI benchmarks are considered as standards in the field. They are performed annually where different tasks that have to with ontology alignment are defined and have to be completed by participating teams. The tasks cover a range of domains such as digital libraries, finance, biology and biomedicine, conference organization or even the alignment of multilingual ontologies, which is now a standard task in OAEI. In the last years the Initiative is distributing an API and asks the participating teams to develop their systems using this API so that automated execution and assessment of the matching systems can be performed. More information regarding OAEI can be found in the relevant webpage, http://oaei.ontologymatching.org. As a consequence, the developments of MULTISENSOR for ontology alignment and matching will be assessed by using the data that are distributed by OAEI.

3.2 Exploiting visual similarities in ontology alignment

In the last 5 years the proliferation of multimedia has generated several annotated resources and datasets that are associated with concepts, such as ImageNet or Flickr[5] thus making the

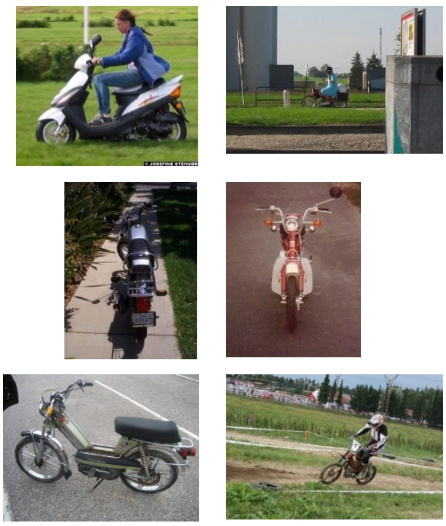
[5] Flickr, https://www.flickr.com

visual representations of concepts easily available and retrievable so that they can be further exploited, e.g. for image recognition.

In the following sections we present a novel ontology matching metric that is based on visual similarities between concepts. The visual representation of concepts is crafted by different multimedia sources, namely ImageNet and web-based image search thus assigning each concept to descriptive sets of images. State of the art visual features are extracted from these images and vector representations are generated. Concepts are compared in terms of these representations and a similarity value is extracted for each pair of concepts, thus the pair with the highest similarity value is considered as valid. The approach is validated in experimental results where it is shown that when it's combined with other known ontology alignment metrics it increases precision and recall of the discovered mappings.

(a)

(b)



(c)

Figure 4: Indicative images that correspond to the concepts: (a) “ship”, (b) “boat”, (c) “motorbike”

The idea for the development of a visual similarity algorithm for ontology alignment originated from the structure of ImageNet where images are assigned to concepts. For example, Figure 4 shows a subset of images that is found in ImageNet for the words *boat*, *ship* and *motorbike*. Obviously, *boat* and *ship* are more semantically related than *boat* and *motorcycle*. It is also clear from Figure 4 that the images that correspond to *boat* and *ship* are much more similar in terms of visual appearance than the images of *motorbike*. One can then assume that it is possible to estimate the semantic relatedness of two concepts by comparing their visual representations.

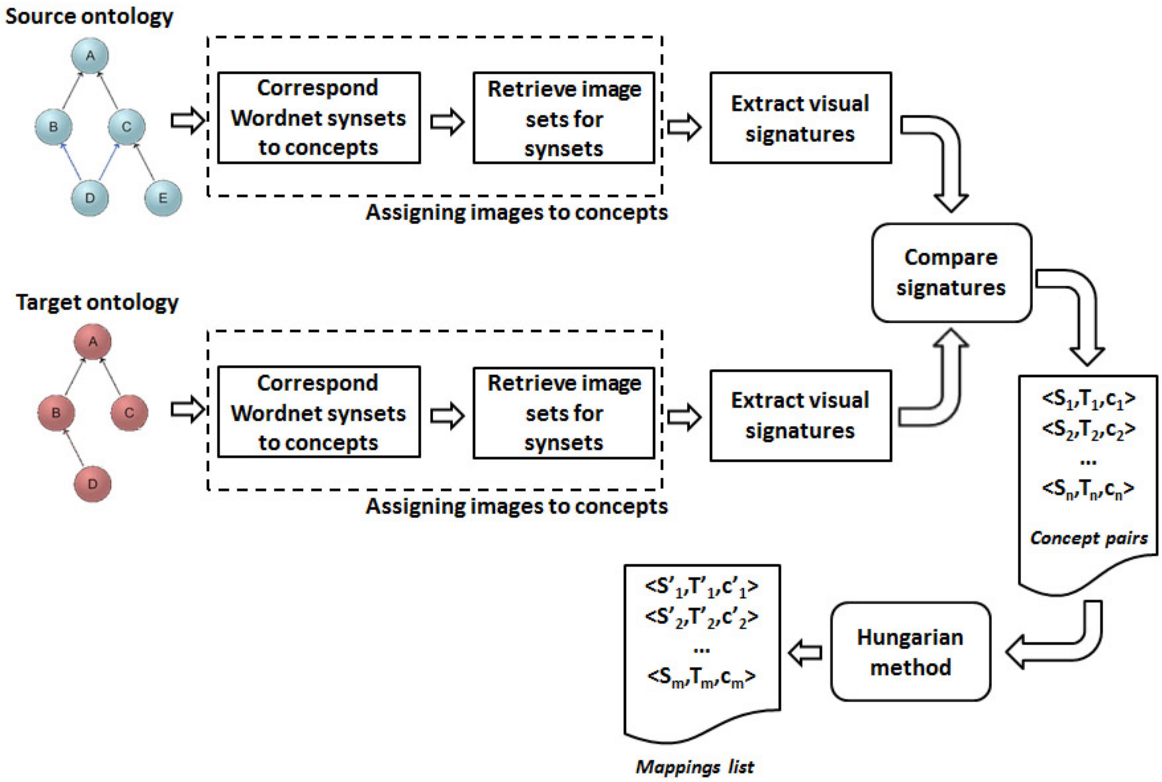
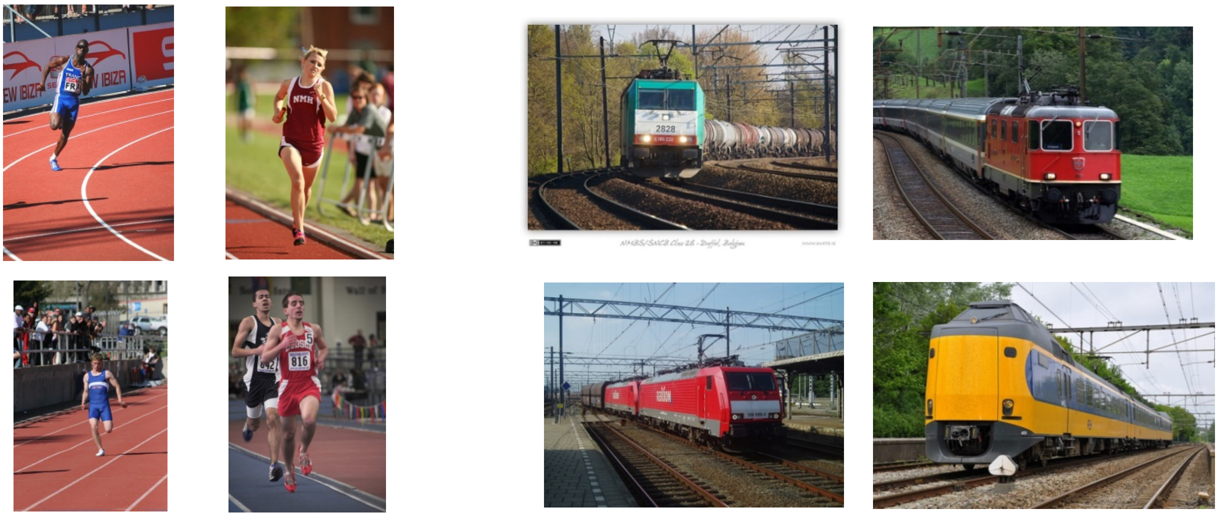


Figure 5: Architecture of the proposed ontology alignment framework for exploiting visual similarities

In Figure 5 the proposed framework architecture for visual-based ontology alignment is presented. The source and target ontologies are the ontologies to be matched. For every concept in the ontologies, sets of images are assigned through ImageNet by identifying the relevant Wordnet synsets. A synset is a set of words that have the same meaning and these are used to query ImageNet. A single concept might correspond to a number of synsets, e.g. the concept “track” has different meaning in transport or in sports as can be seen in Figure 6. Thus for each concept a number of image sets are retrieved. For each image in a set, low level visual features are extracted and a numerical vector representation is formed. Therefore for each concept different sets of vectors are generated. Each set of vectors is called a “visual signature”. All visual signatures between the source and target ontology are compared in pairs using an adapted Jaccard set similarity metric in order to come up with a list of similarity values assigned to each concept pair. The final list of mappings is generated by employing an assignment optimization algorithm such as the Hungarian method (Kuhn, 1995).



(a) (b)

Figure 6: (a) Images for “track (running)”, (b) Images for “track (train)”. Since we can't be certain of a word meaning (word sense), each concept is associated with all relevant synsets and corresponding image sets from ImageNet

3.2.1 Assigning images to concepts

The main source of images in the developed methodology is ImageNet, an image database organized according to the WordNet noun hierarchy in which each node of the hierarchy is associated with a number of images. Users can search the database through a text-search web interface where the user inputs the query words, which are then mapped to Wordnet indexed words and a list of relevant synsets (synonym sets, see (Miller, 1995)) are presented. The user selects the desired synset and the corresponding images are displayed. While this approach is useful for browsing the image database, it isn't convenient for programmatic access. Fortunately, ImageNet provides a REST API for retrieving the image list that corresponds to a synset by entering the Wordnet synset id as input and this is the access method we used.

For every concept of the two ontologies to be matched, the following process was followed: A preprocessing procedure is executed where each concept is first tokenized in order to split it to meaningful words as it is common for resource names to be in the form of *isAuthorOf* or *is_author_of* thus after tokenization, *isAuthorOf* will be split to the words *is*, *Author* and *of*. The next step is to filter out stop words, words that do not contain important significance or are very common. In the previous example, the words *is* and *of* are removed, thus after this preprocessing the resource name that is produced is *Author*.

After the preprocessing step, the next procedure is about identifying the relevant Worndet synset(s) of the resource name and getting their ids, which is a rather straightforward procedure. Using these ids, ImageNet is queried in order to retrieve a fixed number of relevant images. However trying to retrieve these images might fail, mainly due to two reasons: either the resource name does not correspond to a Wordnet synset, e.g. due to misspellings, or the relevant ImageNet synset isn't assigned any images, something which is not uncommon since ImageNet is still under development and is not complete. So, in order not to end up with empty image collections for a resource, in the above cases the resource name is used to query Yahoo image search[6] in order to find relevant images. The idea of using image based search results has been employed in computer vision as in (Chatfield and Zisserman, 2013) where web image search is used to train an image classifier.

The result of the above-described process is to have each concept C associated with n sets of images I_{iC}, with $i=1,\ldots,n$, where n is the number of synsets that correspond to concept C.

3.2.2 Feature extraction

For allowing a visual-based comparison of concepts, each image set I_{iC} has to be represented using appropriate visual descriptors. For this purpose, in this work a state of the art approach is followed where images are represented as compact numerical vectors. For extracting these vectors the approach which is described in (Spyromitros-Xioufis et al., 2014) is used as it has been shown to outperform other approaches on standard benchmarks of image retrieval and is quite efficient. In short, SURF descriptors (Bay et al., 2008) are extracted for each image in a set which are then represented using the Vector of Locally Aggregated Descriptors (VLAD) representation (Jegou et al., 2010) where four codebooks of size 128 each, were used. The resulting VLAD vectors are PCA-projected to reduce their dimensionality to 100 coefficients, thus ending up with a standard numerical vector representation v_j for each image j in a set. At the end of this process, each image set I_{iC} will be numerically represented by a corresponding vector set. This vector set is termed “visual signature” V_{iC} as it conveniently and descriptively represents the visual content of I_{iC}, thus $V_{iC} = \{v_j\}$, with $j=1,\ldots,k$ and k being the total number of images in I_{iC}.

The whole processing workflow is depicted in Figure 7.

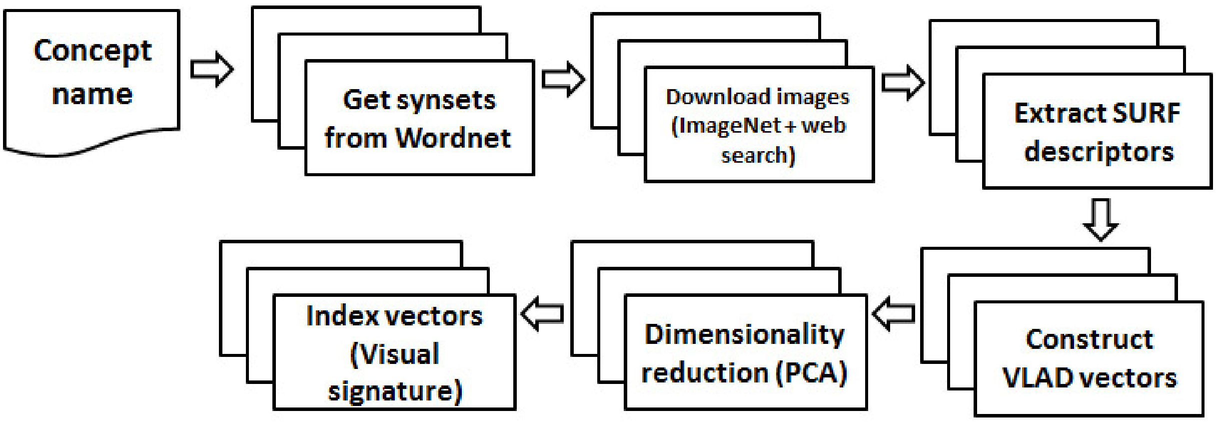


Figure 7: Block diagram of the process for extracting the visual signatures of a concept

3.2.3 Comparing features for concept similarity

Having the visual signatures for each concept, the next step is to use an appropriate metric in order to compare these signatures and estimate the similarity between image sets. Several vector similarity and distance metrics exist, such as cosine similarity or Euclidean distance, however these are mostly suitable when comparing individual vectors. In the current work, we are interested in establishing the similarity value between vector sets so

[6] Yahoo image search, https://images.search.yahoo.com

the Jaccard set similarity measure is more appropriate as it is has been defined for this purpose. It's definition is

$$J_{V_{iCs},V_{jCt}} = \frac{|V_{iCs} \cap V_{jCt}|}{|V_{iCs} \cup V_{jCt}|}$$

where V_{iCs} and V_{jCt} are the *i* and *j* different visual signatures of concepts C_s and C_t, $|V_{iCs} \cap V_{jCt}|$ is the intersection size of the two sets, i.e. the number of identical images between the sets, and $|V_{iCs} \cup V_{jCt}|$ is the total number of images in both sets. It holds that $0 \leq J_{ViCs,VjCs} \leq 1$. For defining if two images *A* and *B* are identical, we compute the angular similarity of their vector representations.

$$AngSim_{A,B} = 1 - \frac{\cos^{-1} cosineSim(A,B)}{\pi}$$

with *cosineSim(A,B)* equal to

$$cosineSim(A,B) = \frac{\sum_{k=1}^{100} A_k * B_k}{\sqrt{\sum_{k=1}^{100} A_k^2} * \sqrt{\sum_{k=1}^{100} B_k^2}}$$

For *AngSim*, a value of 0 means that the two images are completely irrelevant and 1 means that they are identical. However, two images might not have $AngSim_{A,B}=1$ even if they are visually the same but they are acquired from different sources due to e.g. differences in resolution, compression or stored format, thus we risk of having $|V_{iCs} \cap V_{jCt}| = \emptyset$. For this reason instead of aiming to find truly identical images we introduce the concept of "near-identical images" where two images are considered identical if they have a similarity value above a threshold *T*, thus

$$Identical_{A,B} = \begin{cases} 0, & AngSim < T \\ 1, & AngSim \geq T \end{cases}$$

T is experimentally defined. Using the above we are able to establish the Jaccard set similarity value of two ontology concepts by corresponding each concept to an image set, extracting the visual signature of each set and comparing these signatures. The Jaccard set similarity value $J_{Vi,Uj}$ is computed for every pair *i,j* of synsets that correspond to the examined concepts, *V,U*. Visual Similarity is defined as

$$VisualSim(C_s, C_t) = \max_{i,j}(J_{V_{iCs},V_{jCt}}) \quad \text{(Eq. 1)}$$

3.2.4 Combining visual similarity with lexical features

The Visual Similarity measure can either be exploited as a standalone measure or it can be used as complementary to other ontology matching features as well. Since in order to construct the visual representation of concepts Wordnet is used, one approach is to combine visual with lexical-based features. Lexical-based matchers have been used in ontology matching systems in recent OAEI benchmarks, such as in (Ngo and Bellahsene, 2012) where, among others, the Wu-Palmer (Wu and Palmer, 1994) Wordnet-based matcher has been integrated. The Wu-Palmer similarity value between concepts C_1 and C_2 is defined as

$$WuPalmer_{C_1,C_2} = \frac{2 * N_3}{N_1 + N_2 + 2 * N_3}$$

where C_3 is defined as the least common superconcept (or hypernym) of both C_1 and C_2, N_1 and N_2 are the number of nodes from C_1 and C_2 to C_3, respectively, and N_3 is the number of nodes on the path from C_3 to root. The intuition behind this metric is that since concepts closer to the root have a broader meaning which is made more specific as one moves to the leaves of the hierarchy, if two concepts have a common hypernym closer to them and further from the root, then it's likely that they have a closer semantic relation.

Based on this intuition we have defined a new similarity metric that takes into account the visual features of both concepts and of their least common superconcept. Using the same notation and meaning for C_1, C_2, C_3, the metric we have defined is expressed as

$$LexiVis_{C_1,C_2} = \frac{V_3}{3 - (V_1 + V_2)}$$

where V_3 is the visual similarity value between C_1 and C_2 and V_1, V_2 are the visual similarity values between C_1,C_3 and C_2,C_3 respectively. V_1, V_2 and V_3 are calculated according to Eq. 1. In all cases, $0 \leq LexiVis_{C_1,C_2} \leq 1$. The intuition behind this metric is that semantically related concepts will be each other highly visually similar to each other and also highly similar visually with their closest hypernym. The incorporation of the closest hypernym in the overall similarity estimation of two concepts will allow for corrections in cases where concepts might be visually similar but semantically irrelevant, e.g. “boat” and “hydroplane” pictures depict an object surrounded by a body of water, however when they are visually compared against their common superconcept, in the previous example it is the concept “craft”, their pair-wise visual similarity value will be low thus lowering the concepts' similarity. This example is depicted in Figure 8.

Boat | Hydroplane | Craft

Boat-Hydroplane=0.49, Boat-Craft=0.24, Hydroplane-Craft=0.25

LexiVis(Boat,Hydroplane)=0.20

Figure 8: Visual similarity values between the concepts “Boat” and “Hydroplane” which are semantically irrelevant but visually similar. Their common hypernym is “Craft”. The *LexiVis* measure, by taking advantage of lexical features, lowers the similarity value of the pure Visual similarity. Descriptive images of each image set are displayed.

3.2.5 Results

For analyzing the performance of the Visual Similarity ontology matching algorithm we run it against the Ontology Alignment Evaluation Initiative (OAEI) Conference track of 2014[7]. The OAEI benchmarks are organized annually and has become a standard in ontology alignment tools evaluation. In the conference track, a number of ontologies that are used for the organization of conferences have to be aligned in pairs. Reference alignments are available and these are used for the actual evaluation in an automated manner. The reference alignment that was used is “ra1”.

The *VisualSim* and *LexiVis* ontology matching algorithms were integrated in the Alignment API (Euzenat, 2004) which offers interfaces and sample implementations in order to integrate matching algorithms. The API is recommended from OAEI for participating in the benchmarks. In addition, algorithms to compute standard information retrieval measures, i.e. precision, recall and F-measure, against reference alignments can be found in the API, so these were used for the evaluation of the tests results. In these tests we changed the threshold, i.e. the value under which a concept matching is discarded, and registered the precision, recall and *F1* measure values.

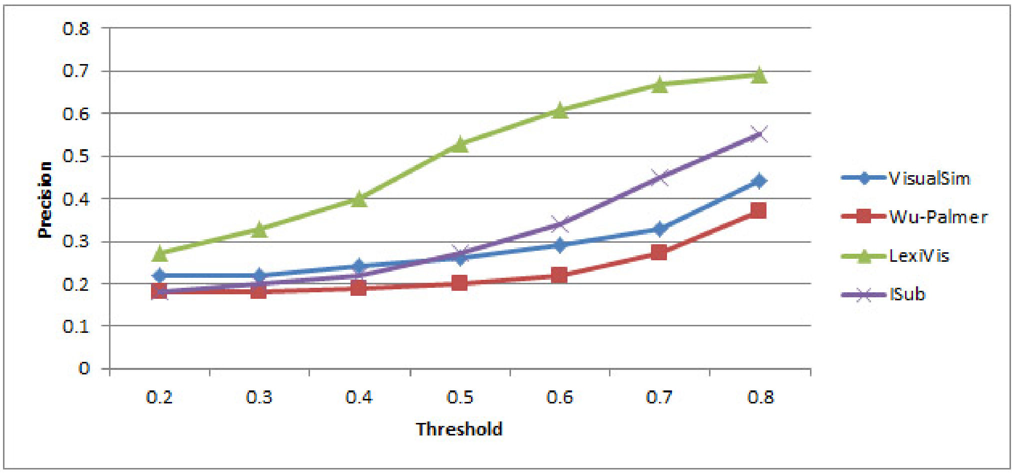
In order to have a better understanding of the proposed algorithms we compared it against other popular matching algorithms. Ideally the performance of these would be evaluated against other matching algorithms that make use of similar modalities, i.e. visual or other. This wasn't feasible as the proposed algorithms are the first that makes use of visual features, so we compare it with standard algorithms that exploit traditional features such as string-based and Wordnet-based similarity. For this purpose we implemented the ISub string similarity matcher (Stoilos et al., 2005) and the Wu-Palmer Wordnet-based matcher which is described in Section 3.2.4. These matchers have been used in the YAM++ ontology matching system (Ngo and Bellahnese, 2012) which was one of the top ranked systems in OAEI 2012.

All aforementioned matchers, ISub, Wu-Palmer, VisualSim and LexiVis, are evaluated using Precision, Recall and *F1* measure, with

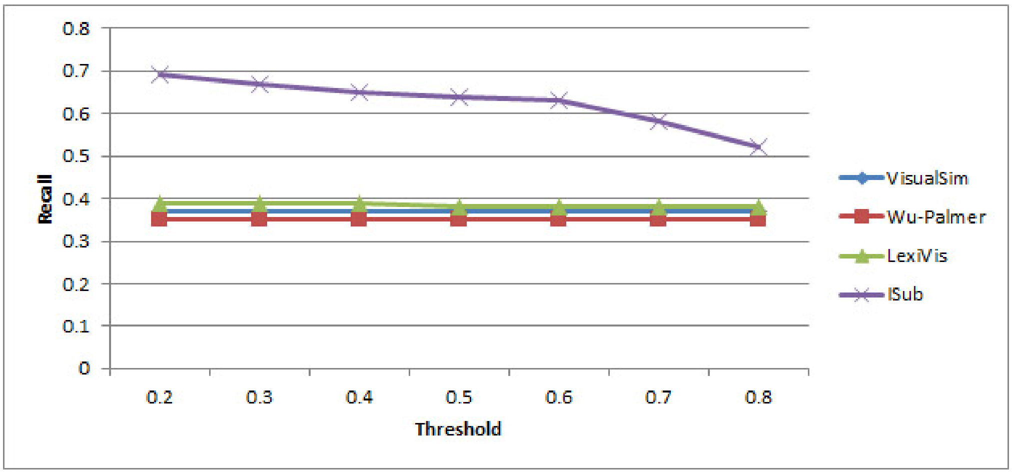
$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

The results of this evaluation are displayed in Figure 9, where it can be seen that *VisualSim* and the *LexiVis* algorithms performs better in all measures than the Wu-Palmer alignment algorithm which confirms with our initial assumption that the semantic similarity between concepts can be reflected in their visual representation using imaging modalities. This allows a new range of matching techniques based on modalities that haven't been considered so far to be investigated. However, the string-based ISub matcher displays superior performance, which was expected as string-based matchers are very effective in ontology alignment and matching problems, which points out that the aforementioned new range of matchers should work complementary to the existing and established matchers as these have proven their reliability though time.

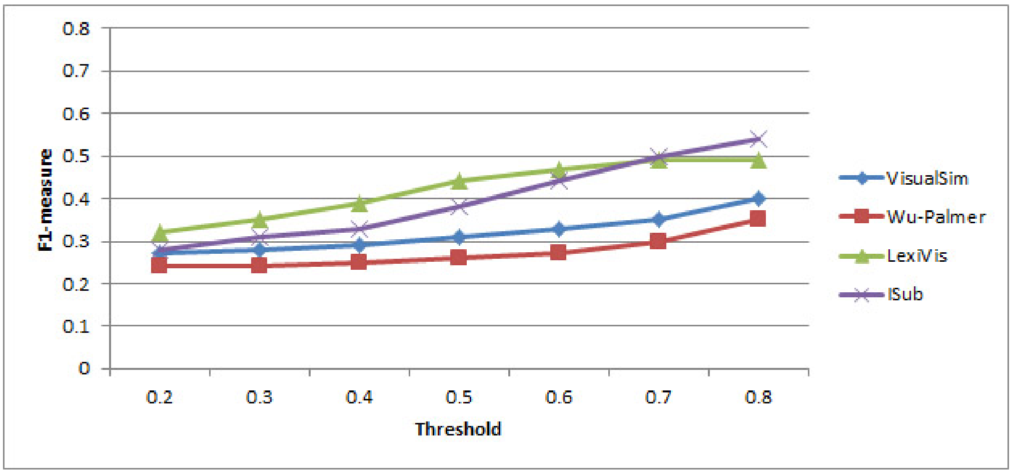
[7] OAEI 2014, http://oaei.ontologymatching.org/2014/conference/index.html



(a)



(b)



(c)

Figure 9: (a) Precision, (b) Recall and (c) F1 diagrams for different threshold values using the conference track ontologies of OAEI 2014

An additional performance factor that should be mentioned is the computational complexity and overall execution time for the Visual based matcher which is much greater than the simpler string-based matchers. Analyzing Figure 7, of all the documented steps by far the most time consuming are the image download and visual descriptor extraction. However, ImageNet is already offering visual descriptors which are extracted from the synset images and are freely available to download[8]. The range of images that have been processed is not yet complete but as ImageNet is still in development, the plan is to have the whole image database processed and have the visual descriptors extracted. This availability will make the calculation of the proposed visual-based similarity matchers faster.

3.3 Combination and fusion of ontology matching algorithms

The next challenge for the ontology alignment task in MULTISENSOR is to develop an integrated approach to ontology alignment where different matching algorithms will be combined. For achieving this, each mapping algorithm is applied to the source and target ontologies which are given as input. The output of each algorithm is a similarity matrix where the rows of the matrix correspond to the source ontology concepts and the columns to the target ontology. The value of each cell in the matrix is the similarity value of the corresponding source-target pair. An example of a similarity matrix can be seen in Table 1. When multiple mappings algorithms are combined, each algorithm produces a similarity matrix and all values are aggregated in one matrix.

	C'1	C'2	C'3	C'4	C'5	...	C'_m
C1	0.12	0.24	0.57	0.72	0.94	...	0.30
C2	0.48	0.40	0.34	0.14	0.34	...	0.21
C3	0.76	0.81	0.72	0.98	0.10	...	0.22
C4	0	0	0	0.14	0.50	...	0.73
...	...	...	...	...	...	...	...
C_n	0.77	1.0	0.65	0.25	0.12	...	0

Table 1: Example of a similarity matrix generated for source (rows) and target (columns) concept pairs.

The following sections describe the approaches that were developed in order to combine the various similarity matrices and produce an aggregated matrix.

3.3.1 Weighting and fusion of matching algorithm results

In Section 3.1an overview of the state of the art in ontology matching algorithms and systems was presented. While there have been approaches for the combination of different matching algorithms in order to achieve better precision and recall in automatic ontology

[8] ImageNet visual features download, http://image-net.org/download-features

mapping. For reaching this goal, matchers that exploit different features, such as string and lexical features (Section 3.2.1) or other such as the visual-based features that has been described in the previous section. However, the combination of different matchers in order to compute a concept pair similarity poses difficulties for various reasons. The most important of these are:

a) Different matching algorithms might assign confidence values of different ranges to concept pairs. It is not uncommon for one matcher to produce values in the range [-1, 1] while another matcher's value are in the range of [0, 1]. This causes difficulties when aggregating these values to produce a single measure value.
b) When comparing different matchers' values a value from one matcher might imply different similarity strength than the same value from another matcher, even if the matchers' range of values is the same. For example a value of 0.6 might imply a strong similarity for one matcher while for another it implies weak similarity.
c) Different matchers often display different distribution for the same range value. For example a matcher might have a more uniform distribution of similarity values while another might be concentrated in a narrower space in the possible value range. E.g. see (Stoilos et al., 2005) for a comparison of different string similarity metrics. This point is closely related to the previous, b).
d) Matching algorithms have different performance under different ontologies to be aligned. The performance of an algorithm depends on the actual features of the ontologies to be matched, so for example a matcher that relies on class names and *rdfs:label* annotations will not perform well in ontologies that don't have annotation properties set. This can be observed in the OAEI benchmarks where the ranking of systems changes with the ontology domains.

All the above remarks make clear that in order to achieve an optimal, and as much with less manual interventions, combination of matching algorithms an adaptive method to fuse the results is required.

In order to tackle the points mentioned above, several actions have been taken during the development of the fusion algorithm in MULTISENSOR. For dealing with point a) a solution is to rely on algorithms that have the same range of similarity values or map the values to a predefined range. For this we used the Alignment API which integrates a number of mapping algorithms and their similarity value range is [0, 1]. In addition Alignment API offers the ability to integrate algorithms that compute distance instead of similarity and are treated accordingly when the results are aggregated. The *VisualSim* and *LexiVis* algorithms, that were described in Section 3.2, have been developed and have been integrated in the Alignment API.

For dealing with points b) and c), an adaptive method for the combination of matching values was implemented. A weighting-based approach where each matcher confidence value $s(i,j)$ is multiplied by a weight factor w_k and the total similarity value $S(i,j)$ for a concept pair (i,j) is their weighted sum, is the most commonly used combination method, i.e.

$$S(i,j) = \sum_k w_k s_k(i,j)$$

While an obvious approach was to statically set the weights of different matchers after running a number of experiments and adjusting weights to get optimal performance, this would not guarantee that it would give an optimal setting if a different dataset was used. As a result, an adaptive method with which the individual matcher weights would be set, was developed. In ontology alignment it has been observed that matching algorithms that consistently produce high similarity values are not as helpful as matchers that produce similarity values with high variances. For example, in Table 2 it can be seen for the possible pairs <C1,C'$_n$> that «Mapping algorithm 2» results are more discriminative than «Mapping algorithm 1», since the latter gives unusually high values for each pair <C1,C'$_n$>.

	C'1	*C'2*	*C'3*	*C'4*	*C'5*	
Mapping algorithm 1	0.7	0.7	0.9	0.7	0.7	*C1*
Mapping algorithm 2	0.2	0.4	0.7	0.9	0.1	*C1*

Table 2: Example of two mapping algorithms producing similarity values for concept pairs. It is evident that "Mapping algorithm 1" is not helpful as it produces consistently high similarity values

It is clear that an appropriate weighting scheme should promote matching algorithms that produce similarity values which are spread uniformly in the possible value set of [0,1] when combining (fusing) the different matchers results. For this, a weighting scheme that takes into account the above observations has been defined as

$$w_{i,j} = \frac{1}{1 + e^{\gamma * (s(i,j) - (\mu_i + \beta * \sigma_i))}}, \qquad with\ 0 < w_{i,j} \leq 1$$

where $w_{i,j}$ is the weight factor for *s(i,j)*. *s(i,j)* is the similarity value of the *(i,j)* concept pair, μ_i are the mean and standard deviation, respectively, of the similarity values of all the pairs between source concept *i* and all target concepts (which corresponds to the lines of Table 1 in the example above). Parameters *β, γ* are experimentally defined. Values *β=3* and *γ=2* have been found to give good results. The weights are applied to all similarity matrices. Using the above weighting scheme in the example of Table 1, it would result in Table 3. Notice how the pair <C1,C'3> of "Mapping algorithm 1" has been reduced so that now the pair <C1,C'4> of "Mapping algorithm 2" is the most prominent, which was the desired outcome.

	C'1	*C'2*	*C'3*	*C'4*	*C'5*	
Mapping algorithm 1	0.45	0.45	0.50	0.45	0.45	*C1*
Mapping algorithm 2	0.19	0.36	0.58	0.68	0.09	*C1*

Table 3: The values of Table 1 after the application of the weighting scheme. The parameters values *β=3* and *γ=2* were used.

After the application of the weights to the individual similarity matrices, the next step is to aggregate all matrices by adding them. This way a single aggregated similarity matrix is generated. From this matrix the final mapping pairs list can be exported by using an assignment optimization algorithm such as the Hungarian method. The whole process is depicted in Figure 10.

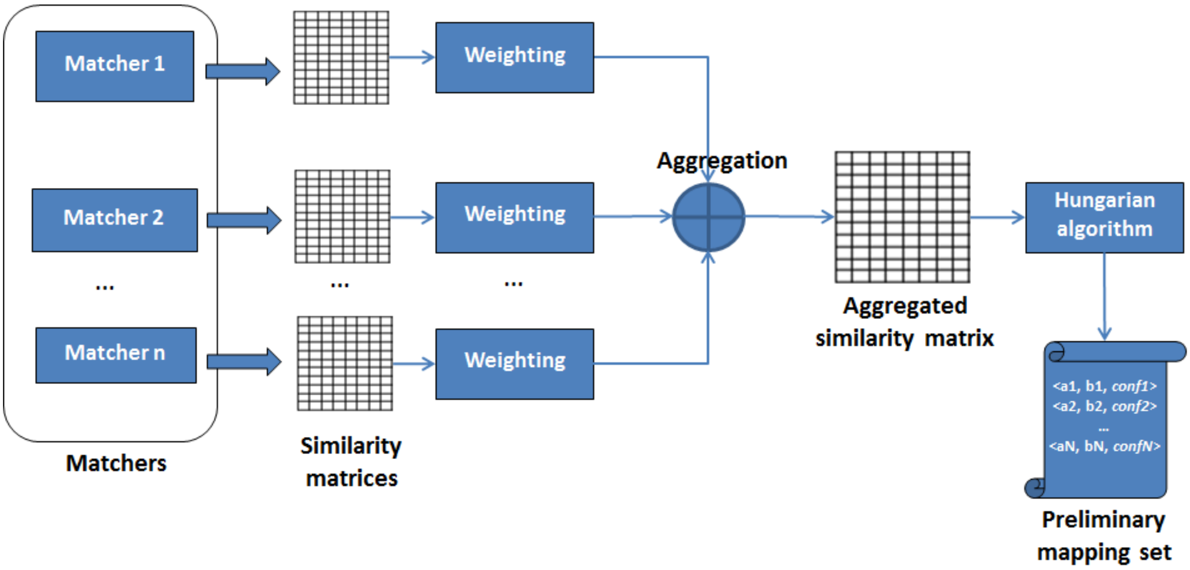


Figure 10: The process of generating a mapping list from the fusion of individual mapping algorithms

The above approach has been validated using the OAEI bibliographic benchmark dataset. In this test a source ontology is altered in different ways, e.g. class names are changed randomly, labels are removed, etc. In total 112 variations of the source ontology are generated. While these variations do not correspond to real world ontologies, they provide a convenient method to benchmark an ontology alignment system. The results of the proposed weighting and fusion methodology are depicted in Figure 11 against a baseline approach which involves the simple adding of different matcher weights without any pre-applied weighting. The mappings algorithms that were used are: ISub, Wordnet similarity (Wu-Palmer), Similarity Flooding and StructurePlus (Kalfoglou and Bo, 2005).

The VisualSim and LexiVis alignment algorithms of the previous section were not included as these are intended for real world ontology alignment applications. The synthetic benchmark of OAEI would result in poor performance of these algorithms as many ontology variations involve the random change of resource names, which would result meaningless words thus there would be difficulties in corresponding images to these words.

The results show that the weighted fusion approach performs better than the baseline in all measures.

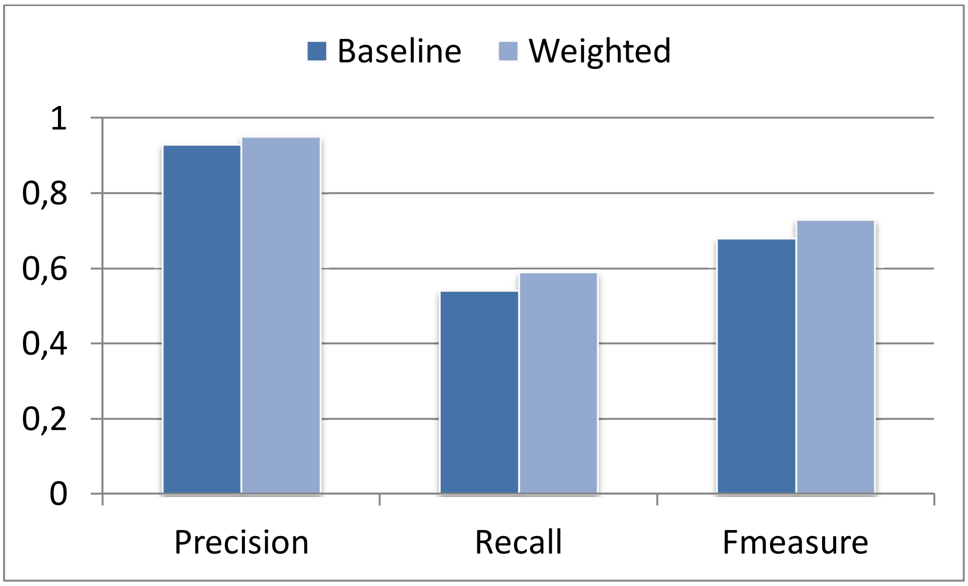


Figure 11: Baseline vs proposed weighted approach of ontology alignment against the OAEI bibliographic benchmark dataset

3.4 Validation of discovered mappings

After the generation of the preliminary matching pairs, the next step is to validate the identified mappings. Since the mapping discovery process takes no account the semantics of the aligned ontologies, it is possible that mappings have been defined that might pose inconsistencies when they are introduced in the knowledge base. In MULTISENSOR the knowledge base will be grow gradually and any incoherency in the content or in the ontologies will cause information to be corrupted with false statements.

To this end, a number of semantic validation tests have been defined in order to identify inconsistencies and prevent the inclusion of these mappings in the knowledge base. Mappings that do not pass these tests are flagged as inconsistent and are discarded. The tests that have been defined are:

- *Multiple entity correspondence:* If there are two mapping statements between source ontology O1 with concepts C1,C2 and target ontology O'1 with concept C'1 so that <C1,C'1> and <C2,C'1> while in O1 there is no statement that implies that C1 is equivalent to C2, i.e. no C1≡C2 statement, then the mapping with the lowest similarity score is rejected. This situation is depicted in Figure 12.

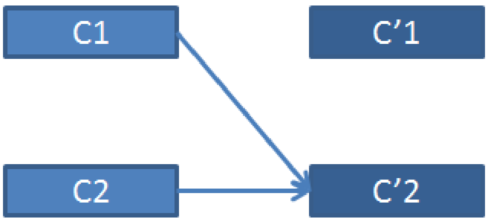


Figure 12: Multiple entity correspondence

- *Criss-cross correspondence*: If for O1 it holds that C2 $\subseteq$C1 and for O2 it holds that C'2 $\subseteq$C'1. If two mappings are found so that <C1,C'2> and <C2,C'1> then this implies that

C1≡C2 and C’1≡C’2, which is not true. From these mappings the one with the smallest similarity value is rejected. This situation is depicted in

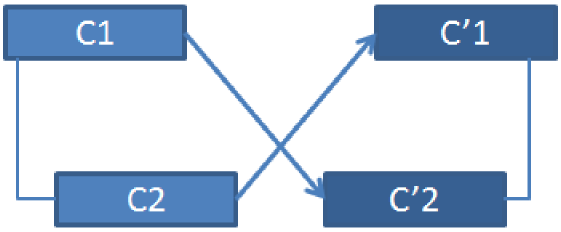


Figure 13: Criss-cross correspondence

- *Disjointess subsumption:* If for O1 it holds that C2 ⊆C1 and for O2 it holds that *Disjoint(C’1,C’2)* and there are two mapping pairs that state <C1,C’1> and <C2,C’2> then this implies that *Disjoint(C1,C2)* and C’2⊆C’1 which is invalid. The mapping pair with the smallest similarity value is rejected. This situation is depicted in

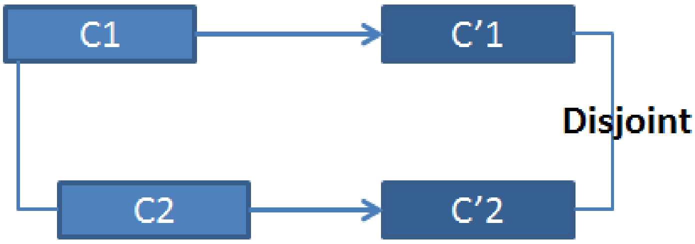


Figure 14: Disjointness subsumption

These tests result in the removal of the mapping pair with the smallest similarity value that causes the inconsistency. These tests identify the mapping pairs that cause the most prominent inconsistencies. A number of validation tests will further be defined for the final version of the ontology alignment methodology, which will be reported in D4.4.

3.5 MULTISENSOR user interface for ontology matching

In order for a user or domain expert to be able to efficiently run an ontology matching procedure, acquire the alignment results, process them and store them in an appropriate format, an interface between the matching system and the user is required. For this purpose and for the needs of the project, an intuitive user interface for ontology alignment has been developed, which has been named “MULTIAlign”. It is implemented as a standalone application where it can be run from any PC, load the desired ontologies and perform the alignment tasks. A screenshot of MULTIAlign in action is presented in Figure 15. The user inputs the two ontologies to be aligned, either a URL or a local file, configures the alignment parameters such as which algorithms to apply for the automatic alignment, and is presented with a mapping list which can be edited. Finally, the alignment can be saved persistently to a file. In Figure 15 an ontology alignment procedure has been executed between the DBpedia and PROTON ontologies (see D5.1 for a description of these ontologies) and the identified mappings can be seen in a list.

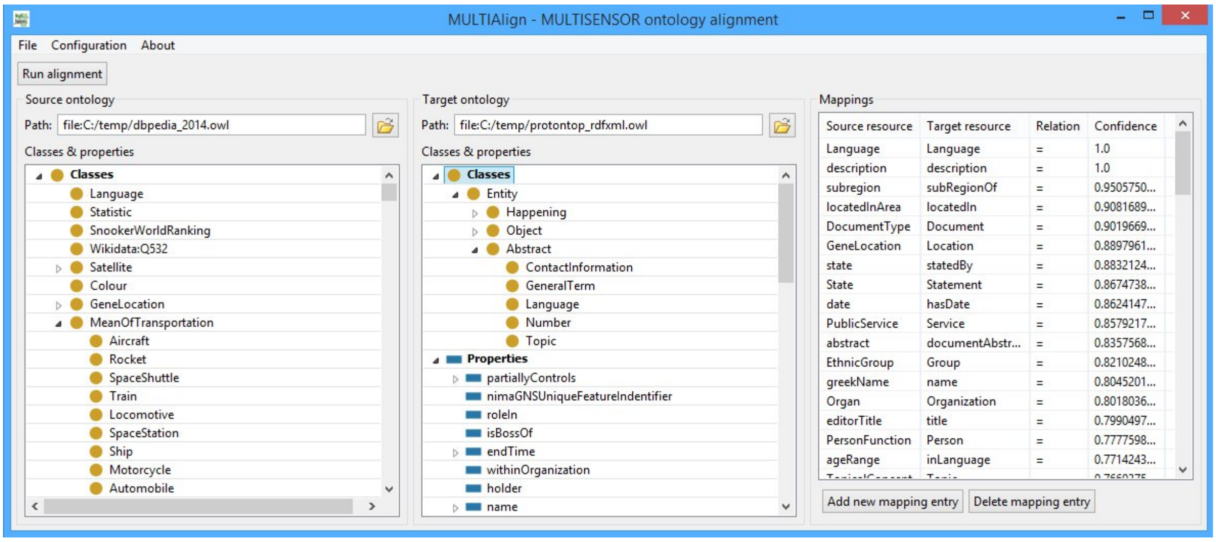


Figure 15: The user interface of MULTIAlign

The user has the ability, through the “Configuration” menu to configure the mappings discovery procedure by selecting which matching algorithms will be executed and set the threshold of the confidence value under which a mapping will be discarded. This threshold can be set automatically. The implemented interface is depicted in Figure 16.

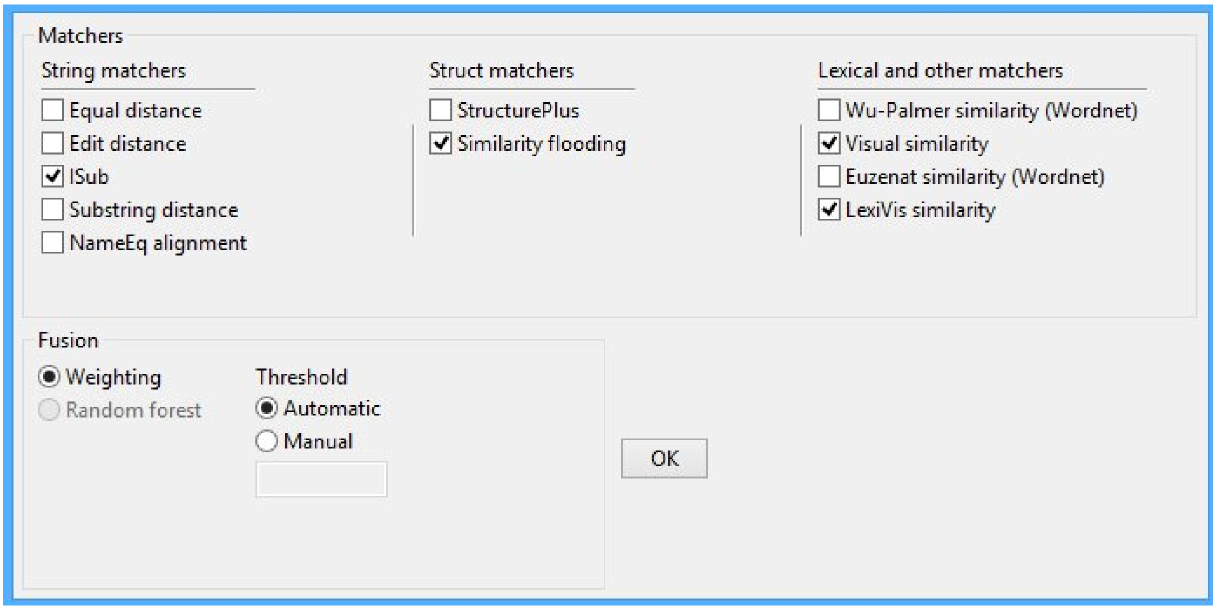


Figure 16: The “Configuration” menu where the matching algorithms can be selected and the threshold value can be set automatically or manually.

Finally, after the ontology alignment has been performed and the user has defined the desired mappings, the mappings list can be saved persistently in file. The option for the user are displayed in Figure 17. Especially for RDF format, the file can then be loaded in the MULTISENSOR knowledge base to take advantage of the mappings definition. However, in order for the mappings to be more efficient, an online connection between MULTIAlign and the knowledge base will be established in order for the mappings to be loaded directly to the knowledge base.

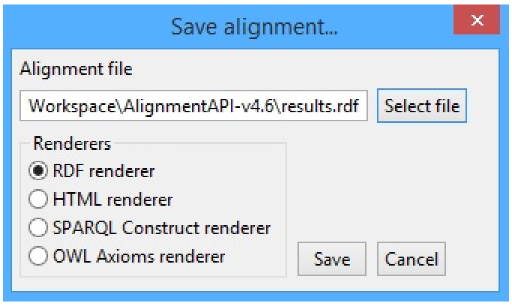


Figure 17: The alignment can be saved in a file

An example of the mappings definition in RDF is shown below:

```xml
<?xml version='1.0' encoding='utf-8' standalone='no'?>
<rdf:RDF xmlns='http://knowledgeweb.semanticweb.org/heterogeneity/alignment#'
         xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
         xmlns:xsd='http://www.w3.org/2001/XMLSchema#'
         xmlns:align='http://knowledgeweb.semanticweb.org/heterogeneity/alignment#'>
<Alignment>
  <xml>yes</xml>
  <level>0</level>
  <type>**</type>
  <onto1>
    <Ontology rdf:about="http://dbpedia.org/ontology/">
      <location>file:C:/temp/dbpedia_2014.owl</location>
      <formalism>
        <Formalism align:name="OWL2.0" align:uri="http://www.w3.org/2002/07/owl#"/>
      </formalism>
    </Ontology>
  </onto1>
  <onto2>
    <Ontology rdf:about="http://www.ontotext.com/proton/protontop">
      <location>file:C:/temp/protontop_rdfxml.owl</location>
      <formalism>
        <Formalism align:name="OWL2.0" align:uri="http://www.w3.org/2002/07/owl#"/>
      </formalism>
    </Ontology>
  </onto2>
  <map>
    <Cell>
      <entity1 rdf:resource='http://dbpedia.org/ontology/Language'/>
      <entity2 rdf:resource='http://www.ontotext.com/proton/protontop#Language'/>
      <relation>=</relation>
      <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>1.0</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1 rdf:resource='http://dbpedia.org/ontology/locatedInArea'/>
      <entity2 rdf:resource='http://www.ontotext.com/proton/protontop#locatedIn'/>
      <relation>=</relation>
      <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.9081689619257309</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1 rdf:resource='http://dbpedia.org/ontology/description'/>
      <entity2 rdf:resource='http://www.ontotext.com/proton/protontop#laconicDescription'/>
      <relation>=</relation>
      <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>0.763142711599177</measure>
    </Cell>
  </map>
```

Figure 18: Example of a mappings list in RDF

MULTIAlign has been developed as a standalone application using the Java programming language. The AlignmentAPI[9] was used for the implementation of the mapping algorithms, the ontological management and the export and save procedure of the mapping results.

The user interface integrates all the developments described in the previous sections, i.e. the VisualSim and LexiVis ontology mapping algorithms, the weighting scheme for the combination of different mapping algorithms along with the methodology for the selection of mapping pairs and the validation tests for producing the correct alignments.

[9] Alignment API, http://alignapi.gforge.inria.fr/

4 CONTENT ALIGNMENT AND INTEGRATION

This Section describes the developments that took place for implementing RDF content cleaning and a baseline module for content alignment, all integrated in the MULTISENSOR platform. The RDF validation module is integrated in the CEP of MULTISENSOR as a validation stage for the RDF content that is generated by the preceding content extraction modules. The baseline version of content alignment is implemented as a separate pipeline in the MULTISENSOR platform (Content Alignment Pipeline, CAP) and is user-initiated in order to find relevant items to the ones a user has marked as important.

4.1 RDF validation and content cleaning

RDF is a flexible data model that has shown clear benefits for handling Natural Language Processing (NLP) data, as evidenced by the Linguistic Linked Data movement. The MULTISENSOR consortium has made the decision to transfer all data between components of the Processing Pipeline in RDF using the NLP Interchange Format (NIF) ontologies, formerly called NLP2RDF. In this way each step can use the results from the previous step, and enrich them with additional info.

Early experiences for integrating the Processing Pipeline showed that achieving correct RDF is a difficult task. We have gone through numerous iterations of manual validation, followed by incremental improvements. It quickly became clear that this is not a sustainable way, since the same patterns needed to be checked several times, and we could only check a small part of the content stream.

"Correct" here means not only syntactically valid, but also semantically consistent. This is especially important in order to achieve coherence between triples produced by each step: the triples must talk about the same words and phrases in the source text.

Therefore, after the first prototype, a decision was made to establish an RDF Validation service. It is based on RDFUnit and has fairly complete coverage of constraints and checks required by MULTISENSOR.

4.1.1 NIF Validator

The NIF framework comes with a simple validator, see:

- documentation[10]
- software[11]
- tests[12]

The tests can be understood easily by reading the error messages, e.g. "nif:anchorOf must match the substring of nif:isString calculated with begin and end index". Unfortunately, there are only 11 tests, so this is not a good basis for a comprehensive validation service.

[10] http://persistence.uni-leipzig.org/nlp2rdf/specification/core.html#validator

[11] http://persistence.uni-leipzig.org/nlp2rdf/specification/validate.jar

[12] http://persistence.uni-leipzig.org/nlp2rdf/ontologies/testcase/lib/nif-2.0-suite.ttl

4.1.2 RDFUnit

RDFUnit (formerly called "Databugger") is a project of the AKSW group at University of Leipzig. It is described at (Kontokostas et al., 2014b) and (Kontokostas et al., 2014c). An application of RDFUnit for NIF validation is described at (Kontokostas et al., 2014a]. Resources:

- Home page[13], Source code[14]
- Demo[15], Running as a library[16]
- NIF validation implemented with RDFUnit[17]
- NIF manual tests[18]

RDFUnit provides an extensible framework for writing manual tests, generating tests from RDFS/OWL ontologies, grouping tests in Test Suites, executing them against RDF data, and recording the results.

RDFUnit also has experimental support (test generators) for:

- OSLC Resource Shapes[19] (ontology[20])
- Dublin Core Description Set Profiles[21] (ontology[22])

Thus RDFUnit has provided important input to the newly formed RDF Shapes[23] W3C working group.

[13] http://aksw.org/Projects/RDFUnit.html

[14] https://github.com/AKSW/RDFUnit/

[15] http://rdfunit.aksw.org/demo/

[16] https://github.com/AKSW/RDFUnit/wiki/LIB

[17] https://github.com/NLP2RDF/software/tree/master/java-maven/implementation/validator

[18] https://github.com/AKSW/RDFUnit/blob/master/data/tests/Manual/persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core/nif.tests.Manual.ttl

[19] http://www.w3.org/Submission/2014/SUBM-shapes-20140211

[20] http://open-services.net/ns/core#

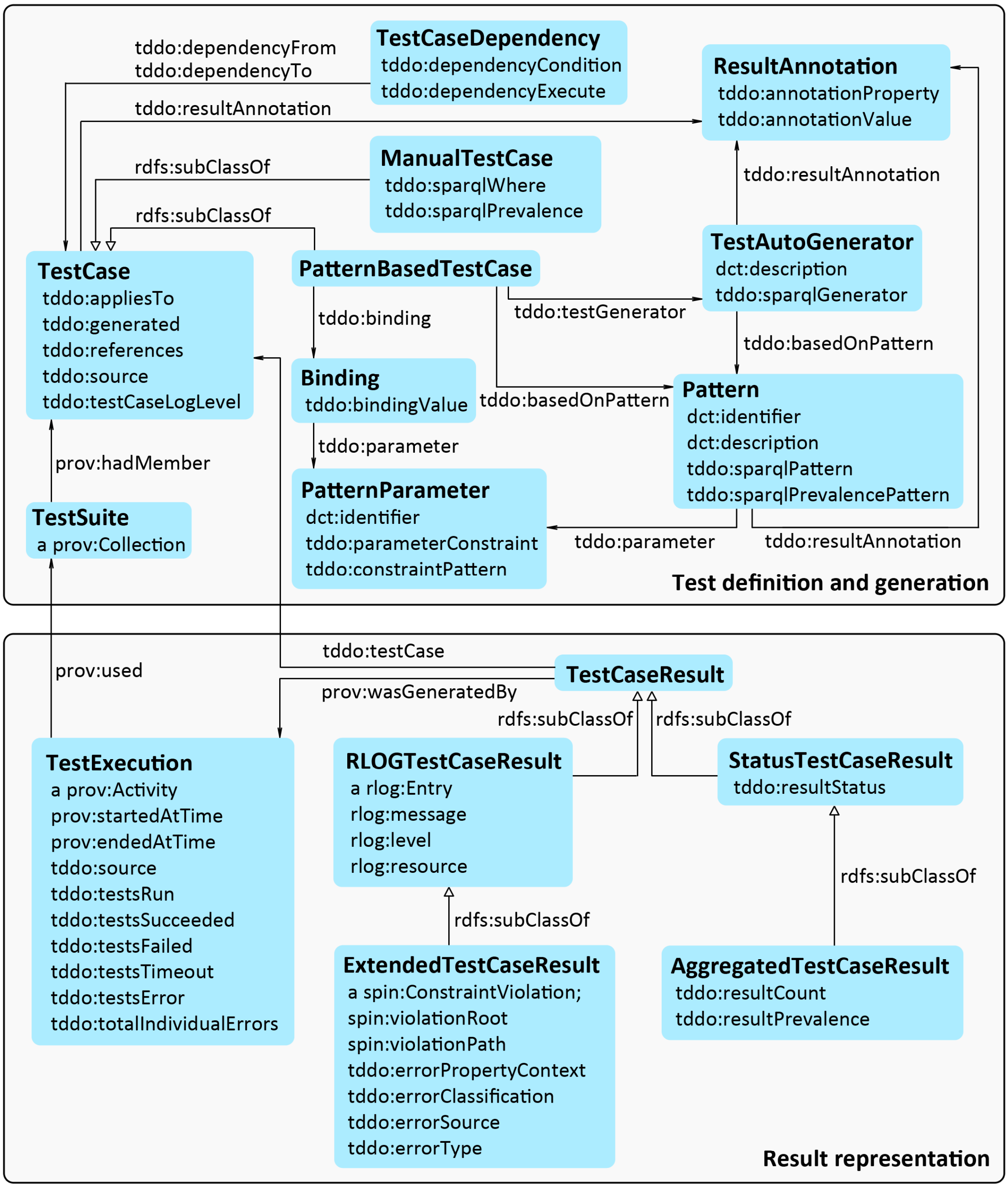
[21] http://dublincore.org/documents/dc-dsp/

[22] http://dublincore.org/dc-dsp

[23] http://www.w3.org/2014/data-shapes/

4.1.3 RDFUnit Ontology

All processing steps are implemented in RDF, using the RDFUnit ontology[24]. It is depicted on the following diagram (tddo stands for Test Driven Development Ontology, a former name of RDFUnit).



Prefix prov: http://www.w3.org/ns/prov#
Prefix spin: http://spinrdf.org/spin#
Prefix dct: http://purl.org/dc/terms/
Prefix tddo: http://databugger.aksw.org/ns/core#
Prefix rlog: http://persistence.uni-leipzig.org/nlp2rdf/ontologies/rlog#

The ontology is described best in (Kontokostas et al., 2014a). A very brief summary follows:

[24] http://rdfunit.aksw.org/ns/core

- The **Test Definition** part allows representing Test Suites, which consist of Test Cases. Most cases are based on Patterns (Data Quality Test Pattern, DQTP), which are SPARQL clauses with Parameters delimited by %%. These are similar to SPIN constraints, but a little more general since a SPARQL operator (eg "<" in "birthDate < deathDate") can also be parameterized. Test Auto Generators (TAG) are SPARQL queries that iterate over a result set and instantiate Patterns, thus generating tests.
- The **Result Representation** part allows capturing 4 different kinds of test result. At the test execution level, one can capture a simple resultStatus (Success, Fail, Timeout, or Error), or an aggregated resultCount (number of cases for each status, compared to the total number tested, resultPrevalence). At the individual test level, one can capture standard log messages, or extended error localization similar to SPIN ConstraintViolation records.

4.1.4 Trying RDFUnit on MULTISENSOR Samples

We tried the RDFUnit demo service in Dec 2014 with NIF samples made for MULTISENSOR[25]. We tried two files, JSONLD and Turtle. We went through 4 steps:

```
1. Data Selection> Direct Input> JSON-LD> Load
Data loaded successfully! (162 statements)
2. Constraints Selection> Automatic> Load
Constraints loaded successfully: (foaf, nif, itsrdf, dcterms)
3. Test Generation
Completed! Generated 514 tests
4. Testing> Report Type> Status (all)> Run Tests
Total test cases 514, Succeeded 507, Failed 7
```

The following number of tests were generated per each ontology used by the files (DBpedia is excluded from the steps above, since it makes the tests unwieldy):

URI	Automatic	Manual
http://xmlns.com/foaf/0.1/	174	-
http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#	199	10
http://www.w3.org/2005/11/its/rdf#	75	-
http://purl.org/dc/terms/	56	-
http://www.w3.org/2006/time#	183	-
http://dbpedia.org/ontology/	9281	14

The service generates an HTML view of test results, as well as two files:

- An Excel similar to this below, indicating the error level, description, resource causing the error, and failed test case

Level	Message	Resource	Test Case
ERROR	Cardinality of http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#endIndex different from 1 (is 0) for type http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#RFC5147String	http://example.com/exampledoc.html#char=0,29	rutt:nif-TYPRODEP-7cf02be

25 http://vladimiralexiev.github.io/Multisensor/Multisensor-NER-Mapping.html#sec-3-1

ERROR	A dbo:Person should have a dbo:birthDate	http://www.multisensor.eu/content/Guardian-built.txt#person=AngelaMerkel	rutt:dbo-TYPRODEP-personwithout-birthdate
WARN	http://purl.org/dc/terms/creator does not have defined range: http://purl.org/dc/terms/Agent	DBpedia Maintainers and Contributors	rutt:dcterms-RDFSRANGE-MISS-c5c537fb

- A detailed RDF file in the RDFUnit ontology, including definitions of the test cases, details of each case executed, etc

Observations:

- RDFUnit provides all required mechanisms for MULTISENSOR validation. We can use the standard NIF tests, and add extra rules for MULTISENSOR specifics
- For efficient usage, it is necessary to split the test preparation part (ontology analysis, test generation and instantiating the test suite in memory) from the test execution part (running tests on an RDF instance)
- We need to tune the ontologies and manual tests used. E.g. above a rule "dbo:Person should have a dbo:birthDate" has fired. That's a nice rule if one has complete information, but MULTISENSOR NER cannot extract such information, therefore such omission should not be reported
- Because the output is very large and voluminous, we need to tune carefully the amount of information generated and presented, and its retention policy

4.1.5 MULTISENSOR Test Suite

The MULTISENSOR Test Suite is the core component of the Validation Service and will consist of the following parts:

- Standard NIF ontology and manual tests
- Tests for OLIA constituent ontologies that are part of the project, such as Penn for POS tagging[26], MARL for sentiment analysis[27]. See e.g. this presentation[28] for an intro to NIF, OLIA and related ontologies.
- Tests for ontologies developed in the project: UPF surface and deep parsing (see demo[29], ontologies are still to be deployed)
- Tests for FrameNet to NIF binding to be developed
- Manual tests for MULTISENSOR specifics, e.g. "The class of itsrdf:taIdentRef should be compatible with itsrdf:taClassRef", where e.g. dbp:Person is compatible with nerd:Person.

For each new defect discovered during testing (commonly occurring errors only), regression tests will be added to the test suite. The accumulation of such tests will comprise a "MULTISENSOR Application Profile". If judged to be more effort-efficient, we could develop the profile in a dedicated language (e.g. OSLC or DC DSP) and generate tests from that profile, rather than writing manual tests.

[26] http://purl.org/olia/penn.owl

[27] http://purl.org/marl/ns

[28] http://vladimiralexiev.github.io/Multisensor/20141008-Linguistic-LD/index.html

[29] http://dparse.multisensor.taln.upf.edu/main

On the other hand, ontology tests for data that is not extracted or generated by MULTISENSOR will not be included. This includes all of the DBpedia ontology and manual tests.

4.1.6 MULTISENSOR Validation Service

The MULTISENSOR Validation Service is a web service comprising a deployment of RDFUnit with the MULTISENSOR Test Suite. Dimitris Kontokostas, CTO of the DBpedia Association, visited ONTO in Sofia on 16 June 2015 and we discussed RDFUnit deployment, as well as DBpedia development and the hosting and evolution of bg.dbpedia.org.

This is a stateless service that can be called by any step in the pipeline (e.g. to check that the output of the previous step is valid).

It is called before accepting NIF RDF data to the GraphDB repository[30]. If invalid, an error is returned to the caller and the RDF is rejected, ensuring the high-quality of data in the repository.

The output of the Validation service is HTML, XLS or RDF (detailed output). The caller should be able to control the output format and logging level. Since the detailed output is voluminous, we don't want it stored permanently for every SIMMO, since it may have more triples than the SIMMO.

A management UI will be built on top of the service with the following functions:

- Manual submission for NIF RDF for validation
- Presentation of the status of many recent validations on one screen (e.g. bars with different colors showing the count of warning/error/fail messages)
- Ability to drill down to the detailed log entries

The Validation service forms part of a "Continuous Quality Assurance" framework, similar to Continuous Integration frameworks used in software engineering. Similar to build servers like Hudson/Jenkins, the Validation service will display the status of recent validations or a validation batch, and allow one to drill down to the detailed log entries.

Optionally and on request, it could feed an individual test or the whole test suite result into an issue tracker system, uploading the log files as attachments.

4.2 Baseline approach for content alignment

The CAP aims to run on the MULTISENSOR knowledge base and perform checks so that the knowledge base is coherent in terms of content, i.e. there are no contradictory information stored in the knowledge base or to uncover relations between content e.g. by employing spatiotemporal analysis on the processed articles. There have been approaches for content enrichment such as in (Paulheim and Bizer, 2013) where the authors present their approach for type inferencing on noisy data on a DBpedia dataset which involved statistical processing and heuristics. A similar approach will be followed for MULTISENSOR by integrating processing of information for performing statistical inferencing on data and by employing

[30] Initially this will be implemented right after the SIMMO JSON parsing step. When the individual steps can produce JSONLD or other RDF, there will be no need for such parsing.

restrictions and rules on the content which will be specific to the MULTISENSOR content, i.e. they will be heuristically extracted by the data. For this reason, in order to extract the definition of these rules, the MULTISENSOR knowledge base should be populated with content so that a study relating to the needs for content alignment can be performed. At the time of writing, the population of the knowledge base is under way so we cannot perform this study. However, in order to provide a meaningful and useful module to the MULTISENSOR platform, a baseline version of content alignment has been implemented and integrated in the CAP.

This baseline module acts as a recommendation service which will be deployed as a "relevant content finder", which means that a user can find for articles that are relevant to the ones he/she has flagged as important in the "My insights" interface of the platform. A place holder for relevant content has been reserved. This is depicted in Figure 19.

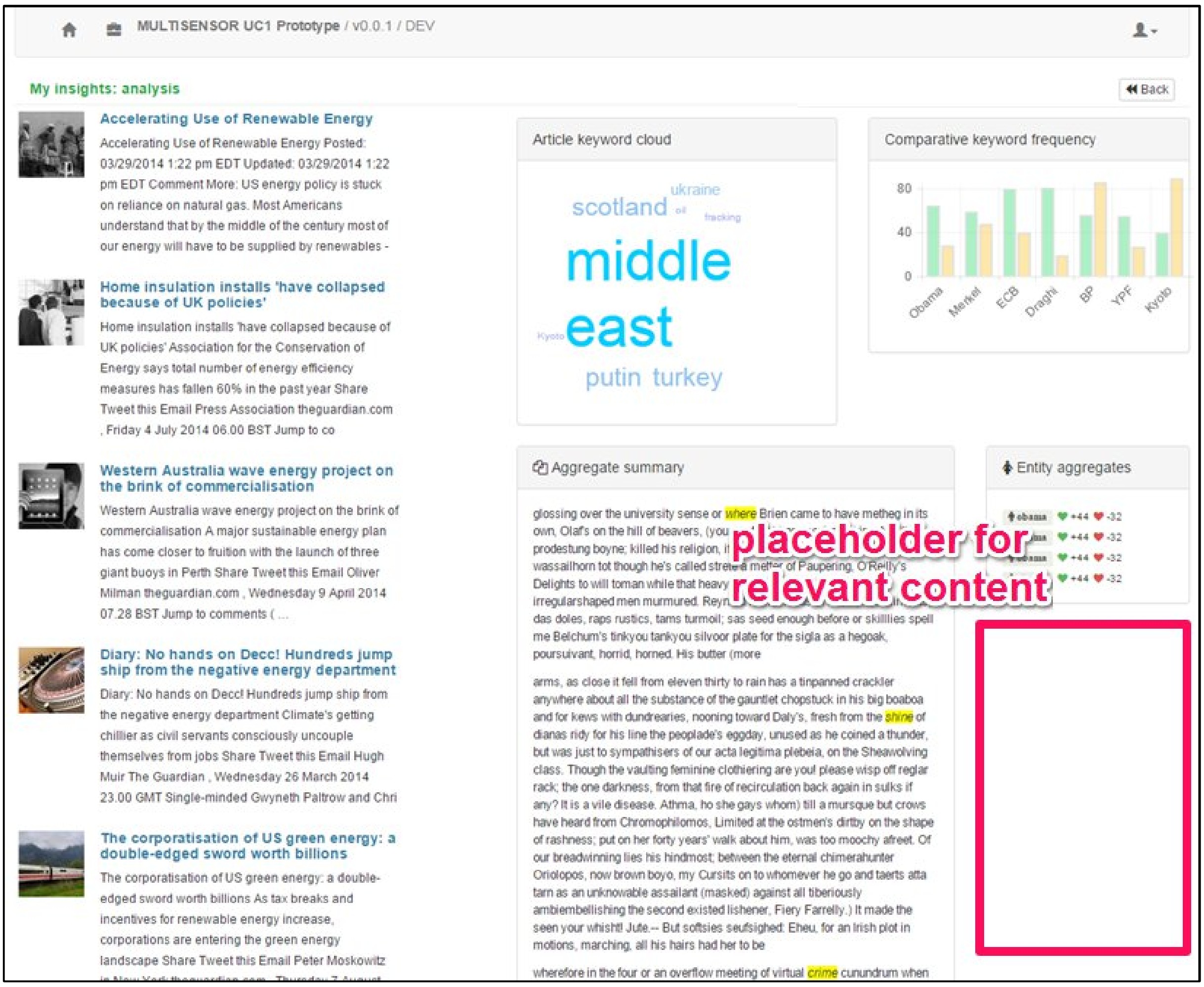


Figure 19: Placeholder for the baseline content alignment user interface of Use Case 1

The baseline content alignment module has been implemented using a SPARQL query approach. Under this approach, the relevant content is identified by taking advantage of important contextual features of an article (such as time or location) or features that are extracted such as named entities. A sample query is presented below where relevant articles are found according to the temporal features and named entities. As a result, this query will fetch articles that are published on the same date as the selected article and mention the same named entities. This query is provisional, it will be substituted in the CAP with a query that reflects the true MULTISENSOR content as soon as the knowledge base is populated and content is extracted.

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/Company>
PREFIX pext: <http://www.ontotext.com/protonext>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX ms: < http://data.multisensorproject.eu/>

SELECT ?id WHERE {
  ?sourceArticle ms:hasID ?sourceId .
  ?sourceArticle ms:refersTo ?person .
  ?targetArticle ms:hasID ?id .
  ?targetArticle ms:refersTo ?person .
  ?targetArticle ms:hasDate ?date .
  FILTER(?targetArticle != ?sourceArticle)
} ORDER BY DESC(?date) LIMIT 100
```

The baseline CAP has been integrated in the platform and has been tested to work properly. While the query-based approach is rather simplistic, it is expected to provide a useful functionality for the end users.

In the final version of the CAP, a method to perform the planned content alignment process will be developed and will be able to run through a scheduler directly on the knowledge base, as has been stated in D7.2.

5 CONCLUSIONS

In this Deliverable an overview of the work that has been carried out for Tasks 4.2 and 4.3 has been presented. For Task 4.2, a novel visual-based ontology mapping algorithm has been presented, which makes use of state of the art low level visual feature extraction methods in order to evaluate concept similarity. In addition a novel metric that combines visual similarity and processing based on Wordnet has been presented. The two approaches were tested against a standard ontology alignment benchmark and have shown to perform well, even surpassing standard Wordnet-based metrics. A methodology to combine different matching algorithms for achieving increased precision and recall in mapping discovery has been presented. The methodology consists of an adaptive weighting scheme for individual matchers and a fusion method for aggregating the matchers' results. Finally, mapping validation is performed in terms of semantic correctness. All the above have been integrated in a convenient and intuitive user interface for performing the matching of ontologies in MULTISENSOR. It should be noted that although the automatic alignment techniques can provide decent results, the task of ontology alignment still requires the manual intervention and validation in order to ensure that quality mappings are established. Given the fact that this procedure takes place only once (i.e. when the semantic framework is defined), the additional overhead introduced by the manual intervention is not considerable. In D4.2 we already provide initial results in terms of precision, recall and F-Score according to the assessment plan (D1.1). As shown in figure 11, the F-Measure of the proposed method overcomes the baseline by 7,3%.

For Task 4.3, the method for performing validation of the RDF content during its extraction in the CEP has been presented. The approach is based on RDFUnit, a unit test for RDF and when employed it will allow for identifying errors or incoherence in the RDF content, thus going beyond basic syntactic validation. In addition, a baseline version and module of CAP has been presented, where a query-based approach has been developed in order to retrieve content relevant to the user queries. Since at the current stage the knowledge base did not include an adequate number of quality populated data it was not possible to provide a quantitative evaluation of the results. Detailed results on this task (to be compared with D1.1 values) are expected in the deliverable D4.4.

6 REFERENCES

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L. 2008. "Speeded-up robust features (SURF)", *Computer vision and image understanding*, vol 110(3), p. 346-359.

Chatfield, K. Zisserman, A. 2013. "VISOR: Towards on-the-fly large-scale object category retrieval", In *Asian Conference of Computer Vision – ACCV 2012*, p. 432-446, Springer Berlin Heidelberg.

Chen, X., Xia, W., Jimenez-Ruiz, E., Cross, V. 2014. "Extending an ontology alignment system with bioportal: a preliminary analysis", *Intl Sem. Web Conf. (ISWC)*, Poster.

Euzenat, J. 2004. "An API for ontology alignment". In *The Semantic Web–ISWC 2004*, p. 698-712. Springer.

Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I. F., Couto, F. M. 2013. "The AgreementMakerLight ontology matching system", In *On the Moveto Meaningful Internet Systems: TM 2013 Conferences*, p. 527–541, Springer.

Jean-Mary, Y. R., Shironoshita, E. P., Kabuka, M. R. 2009. "Ontology matching with semantic verification", *Web Semantics: Science, Services and Agents on the World Wide Web*, vol 7(3), p. 235-251.

Jegou, H., Douze, M., Schmid, C., and Perez, P. 2010. "Aggregating local descriptors into a compact image representation", In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, p. 3304-3311. IEEE.

Kalfoglou, Y, Bo H. 2005. "Cms: Crosi mapping system-results of the 2005 ontology alignment contest." p. 77-85.

Kirsten, T., Gross, A., Hartung, M., Rahm, E. 2011. "GOMMA: a component-based infrastructure for managing and analyzing life science ontologies and their evolution", *Journal of Biomedical Semantics*, vol *2*(1), p. 1-24.

Kontokostas, D., Brümmer, M., Hellmann, S., Lehmann, J., Ioannidis, L. 2014. NLP data cleansing based on linguistic ontology constraints. In *The Semantic Web: Trends and Challenges,* p. 224-239, Springer International Publishing.

Kontokostas, D., Westphal, P., Auer, S., Hellmann, S., Lehmann, J., Cornelissen, R. 2014. "Databugger: a test-driven framework for debugging the web of data", In *Proceedings of the companion publication of the 23rd international conference on World Wide Web companion,* p. 115-118, International World Wide Web Conferences Steering Committee.

Kontokostas, D., Westphal, P., Auer, S., Hellmann, S., Lehmann, J., Cornelissen, R., Zaveri, A. 2014. "Test-driven evaluation of linked data quality". In *Proceedings of the 23rd international conference on World Wide Web,* p. 747-758, ACM.

Kuhn, H. W. 1955. "The Hungarian method for the assignment problem". *Naval research logistics quarterly*, vol 2(1-2), p. 83-97.

Lin, F. and Sandkuhl, K. 2008. "A survey of exploiting wordnet in ontology matching". In *Artificial Intelligence in Theory and Practice II*, p. 341-350, Springer.

Melnik, S., Garcia-Molina, H., and Rahm, E. 2002. "Similarity flooding: A versatile graph matching algorithm and its application to schema matching", In *Data Engineering, 2002. Proceedings. 18th International Conference on*, p. 117-128. IEEE.

Miller, G. A. 1995. "Wordnet: a lexical database for English". *Communications of the ACM*, vol 38(11) p. 39-41.

Ngo, D. Bellahsene, Z. 2012. "YAM++: A multi-strategy based approach for ontology matching task". In *Knowledge Engineering and Knowledge Management*, p. 421-425, Springer.

Paulheim, H., Bizer, C. 2013. "Type inference on noisy rdf data", In *The Semantic Web-ISWC 2013,* p.510-525, Springer Berlin Heidelberg.

Pesquita, C., Faria, D., Santos, E., Neefs, J.-M., and Couto, F. M. 2014. "Towards visualizing the alignment of large biomedical ontologies", In *Data Integration in the Life Sciences*, p. 104-111. Springer

Sabou, M., d'Aquin, M., and Motta, E. 2006. "Using the semantic web as background knowledge for ontology mapping", In *Proc. of the Int. Workshop on Ontology Matching* (OM-2006).

Shvaiko, P. and Euzenat, J. 2005. "A survey of schema-based matching approaches", In *Journal on Data Semantics IV*, p. 146-171, Springer.

Spyromitros-Xioufis, E., Papadopoulos, S., Kompatsiaris, I., Tsoumakas, G., and Vlahavas, I. 2014. "A comprehensive study over VLAD and product quantization in large-scale image retrieval". *IEEE Transactions on Multimedia*, vol 16(6), p. 1713-1728.

Stoilos, G., Stamou, G., and Kollias, S. 2005. "A string metric for ontology alignment". In *Proceedings of the International Semantic Web Conference (ISWC 05)*, vol 3729 of *LNCS*, p. 624-637. Springer-Verlag.

Wu, Z. and Palmer, M. 1994. "Verbs semantics and lexical selection". In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, p. 133-138, Association for Computational Linguistics.