Exploiting DOLCE, SUMO, and OpenCyc to Boost the Ontology Matching Process

Technical Report DISI-TR-08-08, Computer Science Department, DISI, Università degli Studi di Genova, Italy, 2008

Viviana Mascardi
DISI, Università degli Studi di Genova,
Via Dodecaneso 35, 16146, Genova, Italy,
E-mail: viviana.mascardi@unige.it

Angela Locoro
DIBE, Università degli Studi di Genova,
Via All'Opera Pia 11a, 16145 Genova, Italy,
E-mail: angela.locoro@unige.it

Paolo Rosso
Natural Language Engineering Lab., DSIC,
Universidad Politécnica de Valencia,
Camino de Vera s/n, 46022, Valencia Spain,
E-mail: prosso@dsic.upv.es

Abstract

This technical report describes a set of algorithms for exploiting upper ontologies (DOLCE, SUMO, and OpenCyc) as bridges in the ontology matching process, and discusses experimental results. The experiments demonstrate that when our "structural matching method via upper ontology" uses an upper ontology with a sufficient number of entities (OpenCyc, SUMO), the recall is significantly improved and the precision is kept (at least with OpenCyc). Instead, the "non structural matching method" via OpenCyc and via SUMO gives a precision greater than the one obtained with the "non structural direct alignment", and a comparable recall. The "mixed method" that combines the results of both structural direct alignment and structural alignment via upper ontologies, gives the best average recall and F-measure.

1 Introduction

"Ontology matching" is the process of finding correspondences between entities belonging to two or more ontologies. The output of this process, named "ontology alignment", may be computed by using different techniques and is used to cope with the semantic and syntactic heterogeneity of information represented by means of ontologies, in order to identify couples of entities with related meaning, belonging to two different ontologies.

An upper ontology is "*an attempt to create an ontology which describes very general concepts that are the same across all domains*" [16]. Few upper ontologies exist: BFO [8], Cyc [10], DOLCE [7], GFO [9], PROTON [3], Sowa's ontology [13], and SUMO [11]. They vary in dimension, ranging from the 30 classes of Sowa's ontology to the 300,000 of Cyc, in representation language (OWL [15], KIF [1], First Order Logic), in structure (monolithic vs. decomposed into modules), and in developed applications. Nevertheless, all of them describe general concepts (also named "classes") and share the aim to have a large number of ontologies accessible under them. Some of them, such as Cyc and SUMO, also provide domain-dependent sub-ontologies integrated with the upper one.

This technical report describes a set of algorithms for exploiting upper ontologies as bridges in the ontology matching process and discusses experimental results. We have chosen to run our experiments with DOLCE, SUMO, and OpenCyc (the open version of Cyc), because they are the largest and most used upper ontologies. The experiments demonstrate that when our "structural matching method via upper ontology" uses an upper ontology with a sufficient number of entities (OpenCyc, SUMO), the recall is significantly improved and the precision is kept (at least with OpenCyc). Instead, the "non structural matching method" via OpenCyc and via SUMO gives a precision greater than the one obtained with the "non structural direct alignment", and a comparable recall. The "mixed method" that combines the results of both structural direct alignment and structural alignment via upper ontologies, gives the best average recall and F-measure.

The report is organised in the following way: Section 2 describes the algorithms we have designed for implementing the ontology matching via upper ontologies, Section 3 describes the tests we have carried out and their results, and Section 4 concludes.

2 Our Algorithms for Ontology Matching via Upper Ontologies

In order to use upper ontologies as bridges in the ontology matching process, we implemented many algorithms.

Uo_match is based on three functions: $aggregate(a, a')$, $parallel\_match(o, o', res, par)$, and $merge(a, a')$, where o and o' are ontologies, a and a' are alignments, res are external resources, and par are parameters. All the three functions return an alignment.

$Aggregate(a, a')$ produces the alignment obtained by making the union of all the correspondences in a and a', and choosing the correspondence with highest confidence

measure, in case the same correspondence[1] belongs to both a and a'.

$Parallel\_match(o, o', res, par)$ computes an alignment between o and o' by applying substring, n-gram [2], SMOA [14], and language-based methods in parallel, as suggested in [5, Chap. 5.1], and aggregating them. The only external resource we use is WordNet ($res = \{WordNet\}$), which is given in input to the language-based method, and the only parameter is a configurable threshold in $[0, 1]$ used for discarding correspondences that are not relevant, since their confidence is lower than it ($par = \{th\}$). In order to simulate the parallelism of the aggregation process, $aggregate$ is initially called on the first two alignments obtained by running the first two matching algorithms; the output is then aggregated with the third alignment, and so on.

$Merge(a, a')$ computes the alignment a'' in such a way that a correspondence $\langle id, c, c', r, conf\rangle$ belongs to a'' iff $\exists$ c_u such that $\langle id_1, c, c_u, r, conf_1\rangle \in a, \langle id_2, c',$ $c_u, r, conf_2\rangle \in a'$, and $conf = conf_1 * conf_2$. In our algorithm the inputs of $merge$, a and a', are alignments between o and the upper ontology u, and o' and u, respectively. Thus, the second concept c_u in the correspondences belongs to u. If both $c \in o$ and $c' \in o'$ correspond to the same concept $c_u \in u$, then c and c' are related via r with confidence $conf_1 * conf_2$. The choice of $conf_1 * conf_2$ as confidence of the resulting alignment ensures that the confidence remains in $[0, 1]$, and that initial high confidences lead to a resulting confidence which is still high, whereas at least one low confidence leads to a low resulting confidence.
The $uo\_match(o, o', u, res, par)$ function just calls

$$
\begin{array}{l}
merge \\
\quad (parallel\_match(o, u, res, par), \\
\quad parallel\_match(o', u, res, par))
\end{array}
$$

with $res = \{WordNet\}$ and $par = \{th\}$. The th parameter has been set to 0.5 in our experiments, and u is an upper ontology.

Besides the $parallel\_match$ function, we have implemented the $structural\_parallel\_match$ function that adds to the output alignment those correspondences $\langle id, c, c', r,$ $conf\rangle$ such that

- either c is identical to one of c''s super-concepts (or vice versa, c' is identical to one of the super-concepts of c),
- or c and c' have two super-concepts, say $s \in o$ and $s' \in o'$ respectively, identical.

In a similar way, we have implemented a $structural\_merge$ function that merges two alignments considering correspondences involving super-concepts into account. We have prefixed the names of these functions with $structural$ because they look at the structure of the matched ontologies.

$Structural\_merge$ takes a decay factor $df \in [0, 1]$, used to measure the "confidence decay" as we consider relations that involve super-concepts. In our experiments, df has been set to 0.5. It also takes the upper ontology u used as the reference ontology for computing a and a' as input, since it must be navigated in order to find the super-concepts of a given concept.

[1] The "same" apart from the correspondence identifier, which usually will not be the same.

$Structural\_merge(a, a', u, df)$ computes the alignment a'' between o and o' via u in such a way that a correspondence $\langle id, c, c', r, conf\rangle$ belongs to a'' if either

- $\exists\ c_u$ such that $\langle id_1, c, c_u, r, conf_1\rangle \in a, \langle id_2, c', c_u, r, conf_2\rangle \in a'$, and $conf = conf_1 * conf_2$, or
- $\exists\ c_u, c'_u$ such that $\langle id_1, c, c_u, r, conf_1\rangle \in a, \langle id_2, c', c'_u, r, conf_2\rangle \in a'$, c'_u is a super-concept of c_u in u, and $conf = conf_1 * conf_2 * df$ (note the confidence decay multiplicative factor), and vice versa.
- $\exists\ c_u, c'_u$ such that $\langle id_1, c, c_u, r, conf_1\rangle \in a, \langle id_2, c', c'_u, r, conf_2\rangle \in a'$, c_u and c'_u have a common super-concept in u, and $conf = conf_1 * conf_2 * df^2$ (note the power factor applied to the confidence decay).

The $structural\_uo\_match$ function is defined as

$structural\_merge($
$\quad parallel\_match(o, u, WordNet, th),$
$\quad parallel\_match(o', u, WordNet, th),$
$\quad \{u\}, \{df\}).$

The $structural\_uo\_match$ uses the $parallel\_match$ function for computing the alignments, and not the $structural\_parallel\_match$ one. The exploitation of the structure is demanded to the merging stage. The reason why we have also defined a $structural\_parallel\_match$ function is that we want to compare the matching via upper ontology results and the direct match results in the case where both functions exploit structural information, and in the case where both do not. Thus, in our experiments, we have compared the results of uo_match with those of $parallel\_match$, and the results of $structural\_uo\_match$ with those of $structural\_parallel\_match$.

For gaining in efficiency by reusing the results obtained by running uo_match, $structural\_uo\_match$ has been implemented in the following way:

- it creates an alignment between o and o' via u by looking for mappings between
  - concepts belonging to o and super-concepts of concepts belonging o';
  - vice versa, concepts belonging to o' and super-concepts of concepts belonging o;
  - super-concepts of concepts in o and super-concepts of concepts in o'.
- it aggregates the alignment obtained with the uo_match, and the alignment obtained as above.

A $mixed\_match$ algorithm obtained by aggregating the alignment output by the $structural\_parallel\_match$ algorithm (direct matching exploiting structure), and the one output by the $structural\_uo\_match$ algorithm (matching via upper ontology, exploiting structure), has also been implemented.

Our algorithms accept input ontologies expressed in OWL. For their implementation we have extended the Align API version 3.1, delivered on February, the 5th, 2008,

and available form `http://alignapi.gforge.inria.fr/` under GNU Lesser General Public License. Among the methods offered by this API, we have used:

- **StringDistAlignment**, that provides the `subStringDistance`, `ngramDistance`, and `smoaDistance` string metric methods.
- **JWNLAlignment**, that computes a substring distance between the entity names of the first ontology and the entity names of the second ontology expanded with WordNet 2.0 synsets.

We have implemented a **SubSupClassAlignment** method that finds correspondences between $c \in o$ and $c' \in o'$ by looking at string equality between one of them, and one super-concept of the other one, or between super-concepts of both. We have also implemented the methods for alignment aggregation and merge.

We created reference alignments that only match concepts, and we discarded correspondences between individuals and between properties from the alignments computed by our algorithms. The reason of our choice is that finding correspondences between properties in an accurate way would require to take their domain and range into account. The Align API version 3.1 that we used for computing similarity measures cannot take domain, range, and class into account in the correct way. This also explains our choice of discarding property restrictions when exploring the sub- and super-classes, and considering only subClassOf relations with simple entities used as their property values.

In order to discard property restrictions when exploring the sub- and super-classes, we pre-processed the ontologies given in input to our algorithms, and eliminated property restrictions. Also, the easiest and most efficient way to discard correspondences involving properties and individuals from the alignments computed by our algorithms, was to pre-process the input ontologies and eliminate properties and individuals from them. We used the JENA OWL parser, `http://jena.sourceforge.net/`, for the pre-processing activities.

3 Experimentation

As indicators for measuring how good an alignment is, we have used precision, recall and F-measure adapted for ontology alignment evaluation [4].

The methodology that we have followed for carrying out our tests is aimed at ensuring the reproducibility of our experiments by reusing existing ontologies and existing APIs. It may be summarised in the following steps:

1. By exploiting the SWOOGLE Semantic Web Search Engine, `http://swoogle.umbc.edu/`, we have chosen 17 ontologies represented in OWL and available online. Only one of them has been reduced by hand to make it more tractable. The other 16 ones are exactly those that can be downloaded from the URLs listed in Table 1, and that we last accessed on February, 25th, 2008.
2. We have chosen 3 upper ontologies to use in our tests: SUMO, Cyc (in its open version, OpenCyc) and DOLCE. We have downloaded the OWL versions

of OpenCyc and DOLCE from `http://www.cyc.com/2004/06/04/cyc` and `http://www.loa-cnr.it/ontologies/DLP_397.owl` respectively. Our last access dates back to February, 25th, 2008. We have used an OWL translation of the SUO-KIF implementation of SUMO available from `http://sigmakee.cvs.sourceforge.net/sigmakee/KBs/`. The translation was performed using Sigma [12]. It contains SUMO, MILO, and all the domain ontologies but the terrorism and airport ones.

3. Using the JENA OWL parser, we have pre-processed all the 17 chosen ontologies and the 3 upper ontologies, and we have eliminated property restrictions, individuals, and properties.

4. We have designed the 10 tests to run, each one consisting of two ontologies to match and we have created the reference alignment for each test. All the reference alignments are available at `http://www.disi.unige.it/person/MascardiV/Software/OntologyMatchingViaUpperOntology.html`.

5. For each test, we have run the following algorithms:

   - direct alignment without exploiting structure;
   - direct alignment exploiting structure;
   - alignment via SUMO, OpenCyc, and DOLCE without exploiting structure;
   - alignment via SUMO, OpenCyc, and DOLCE exploiting structure;
   - mixed alignment obtained by aggregating the structural alignment via SUMO, OpenCyc, and DOLCE and the structural direct one.

6. For each test, and for each algorithm run within the test, we have computed:

   - number of correspondences found by the algorithm;
   - number of correct correspondences found by the algorithm;
   - precision;
   - recall;
   - F-measure.

   We exploited the library of evaluators provided by the Align API 3.1 for computing them.

7. We have aggregated the obtained results by identifying, for each test, the algorithms that allow us to get the best precision, the best recall, and the best F-measure, and by computing the average advantage of using upper ontologies.

Table 1 provides a summary of the ontologies used in our tests, including the upper ones. **C** stands for the total number of concepts, **O** stands for object properties, namely properties that link individuals to individuals, **D** stands for data properties, namely properties that link individuals to data values, and **I** stands for the number of individuals, or instances of the classes.

Table 2 describes the algorithms that allowed us to obtain the best precision, the best recall, and the best F-measure for each test. *Dir* stands for "direct alignment", *NS* means "no structure", *S* means "with structure", and *M* means "mixed". Sometimes, different algorithms led to the same result. In those cases we listed them all.

In seven tests out of ten, the best precision was obtained by methods that exploit non structural matching via upper ontologies. The best recall is always obtained by the mixed method. In two tests, the same best recall was obtained by the exploitation of structural matching via upper ontologies. The best F-measure is always obtained by methods that exploit upper ontologies. In only one test, the same best F-measure was obtained by direct matching methods too.

The exploitation of OpenCyc ensures a very good precision, while good recall and F-measures can be obtained by both OpenCyc and SUMO in a comparable number of tests.

Table 3 synthesises the average advantage in using upper ontologies vs. not using them. Each column refers to the comparison of the results obtained by using a given upper ontology (first element of the column's name) and a given method (second element of the column's name: again, **NS** means "no structure", **S** means "with structure", and **M** means "mixed"), and those obtained by performing the direct alignment with the same method (with or without structure respectively).

For example, cell ((**SUMO, NS**), **Precision**) reports the average difference in precision between aligning ontologies using SUMO without exploiting the ontology structure, and performing the direct alignment without exploiting the structure. If we identify the precision obtained using SUMO without structure in experiment i with $p(SUMO,\ NoStruct,\ i)$, and the precision obtained by performing the direct alignment without structure in experiment i with $p(Direct,\ NoStruct,\ i)$, then

$$(\mathbf{SUMO}, \mathbf{NS}, \mathbf{Prec.}) = \frac{\Sigma_{i=1}^{10}(p(SUMO, NoStruct, i) - p(Direct, NoStruct, i))}{10}$$

When the "mixed method" is used, we have compared its results with those of the direct alignment that obtains the best F-measure on that test.

4 Conclusions

In this paper we described a set of algorithms for exploiting upper ontologies as bridges in the ontology matching process, and we discussed the results of our experiments. These experiments have been carried out following a rigorous methodology that makes them easily reproducible: the three upper ontologies that we chose, SUMO, OpenCyc and DOLCE, are available on the web, as well as the seventeen ontologies used as inputs for our ten tests, and the ten reference alignment we built by hand; our alignment algorithms extend an existing free API; all the assumptions and simplifications we made are clearly stated and motivated.

The results of our tests may be summarised in the following way:

1. Very small upper ontologies like DOLCE do not possess enough information to properly act as bridges; their usage leads to very poor results.

2. OpenCyc and SUMO are large and detailed enough and give comparable results. OpenCyc performs better than SUMO, but algorithms that use it are less efficient than those that use SUMO, due to its dimensions.

3. If the usage scenario prevents the developer from performing a direct alignment, exploiting structural matching via upper ontologies is a valid alternative for improving the recall (with both SUMO and OpenCyc), keeping the same precision (with OpenCyc). Instead, if precision is more important than recall, the best matching method is the non structural one via OpenCyc and SUMO.

4. If the usage scenario allows the application developer to exploit both direct alignment and alignment via upper ontologies, the mixed method always gives the best recall, even if with a non negligible loss in precision. However, if we agree that F-measure synthesises the good features of the algorithm, mixed methods via SUMO and OpenCyc have the highest average F-measure, and thus should be considered the best ones.

Acknowledgements

We are grateful to A. Pease that provided us with the OWL translation of SUMO.
This work has been partly supported by the "Iniziativa Software" CINI-FinMeccanica project.

References

[1] American National Standard. KIF Knowledge Interchange Format – draft proposed American National Standard (dpANS) NCITS.T2/98-004, 1998.

[2] E. Brill, S. Dumais, and M. Banko. An analysis of the askmsr question-answering system. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2002, Proceedings*, 2002.

[3] N. Casellas, M. Blázquez, A. Kiryakov, P. Casanovas, M. Poblet, and R. Benjamins. OPJK into PROTON: Legal domain ontology integration into an upper-level ontology. In R. Meersman and et al., editors, *WORM 2005, 3rd International Workshop on Regulatory Ontologies, Proceedings*, volume 3762 of *Lecture Notes in Computer Science*, pages 846–855. Springer, 2005.

[4] H. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In A. B. Chaudhri, M. Jeckle, E. Rahm, and R. Unland, editors, *Web, Web-Services, and Database Systems, NODe 2002 Web and Database-Related Workshops, 2002, Revised Papers*, volume 2593 of *Lecture Notes in Computer Science*, pages 221–237. Springer, 2002.

[5] J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer, 2007.

[6] FIPA. *FIPA Ontology Service Specification*, 2001.

[7] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with DOLCE. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 2002.

[8] P. Grenon, B. Smith, and L. Goldberg. Biodynamic ontology: Applying BFO in the biomedical domain. In D. M. Pisanelli, editor, *Ontologies in Medicine*, volume 102 of *Studies in Health Technology and Informatics*, pages 20–38. IOS Press, 2004.

[9] H. Herre, B. Heller, P. Burek, R. Hoehndorf, F. Loebe, and H. Michalek. General formal ontology (GFO): A foundational ontology integrating objects and processes. part i: Basic principles. Technical report, Research Group Ontologies in Medicine (Onto-Med), University of Leipzig, 2006. Version 1.0, Onto-Med Report Nr. 8, 01.07.2006.

[10] D. Lenat and R. Guha. *Building large knowledge-based systems*. Addison Wesley, 1990.

[11] I. Niles and A. Pease. Towards a standard upper ontology. In C. Welty and B. Smith, editors, *FOIS 2001, 2nd International Conference on Formal Ontology in Information Systems, Proceedings*, pages 2–9. ACM Press, 2001.

[12] A. Pease. The Sigma ontology development environment. In F. Giunchiglia, A. Gomez-Perez, A. Pease, H. Stuckenschmidt, Y. Sure, and S. Willmott, editors, *Workshop on Ontology and Distributed Systems, ODS 2003, Proceedings*, volume 71 of *CEUR-WS*, 2003.

[13] J. F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing, 1999.

[14] G. Stoilos, G. B. Stamou, and S. D. Kollias. A string metric for ontology alignment. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *4th International Semantic Web Conference, ISWC 2005, Proceedings*, volume 3729 of *Lecture Notes in Computer Science*, pages 624–637. Springer, 2005.

[15] W3C. OWL Web Ontology Language Overview – W3C Recommendation 10 February 2004 , 2004.

[16] Wikipedia. Upper ontology – Wikipedia, the Free Encyclopedia, 2008. [Online; accessed 30-March-2008].

	URL	C	O	D	I
SUMO	SUO-KIF version available from http://sigmakee.cvs.sourceforge.net/sigmakee/KBs/; the OWL translation performed with Sigma is not available on the Web	4,393	757	2	4,412
OpenCyc	http://www.cyc.com/2004/06/04/cyc	26,965	4,854	1	62,469
DOLCE	http://www.loa-cnr.it/ontologies/DLP_397.owl	242	322	4	12
Agent	212.119.9.180/Ontologies/0.2/agent.owl	130	75	38	139
Bibtex	oaei.ontologymatching.org/2004/Contest/304/onto.rdf	15	0	40	2
Biosphere	sweet.jpl.nasa.gov/ontology/biosphere.owl	88	0	0	0
Ecology	wow.sfsu.edu/ontology/rich/EcologicalConcepts.owl	157	35	2	80
Food	silla.dongguk.ac.kr/jena-owl1/food	65	8	0	57
Geofile	www.daml.org/2001/02/geofile/geofile-ont.daml	89	21	4	131
HL7_RBAC	lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/ontologies/HL7_RBAC.owl	60	24	10	16
Ka	protege.cim3.net/file/pub/ontologies/ka/ka.owl	96	60	32	0
MPEG7	dmag.upf.es/ontologies/2003/03/MPEG7Genres.rdfs	349	1	0	0
Restaurant	guru-games.org/ontologies/restaurant.owl	164	27	29	32
Resume	statistic.gunadarma.ac.id/research/WorkGroupInformationSystem/DownLoad/onto_colection/resume.owl	167	27	35	46
Space	212.119.9.180/Ontologies/0.3/space.owl	165	17	4	0
Subject	www.library.yale.edu/ontologies/subject.owl	171	0	0	0
Top-bio	www.co-ode.org/ontologies/basic-bio/top-bio.owl	65	65	2	1
Travel	lsdis.cs.uga.edu/projects/meteor-s/downloads/Lumina/ontology/travelontology.owl	84	100	112	0
Vacation	www.guru-games.org/ontologies/vacation.owl	32	10	0	324
Vertebrate	*www.co-ode.org/ontologies/basic-bio/basic-vertebrate-gross-anatomy.owl* *	19	0	0	0

Table 1: Matched Ontologies; * Vertebrate has been reduced by hand

	o	o'	Best Precision	Best Recall	Best F-measure
TEST 1	Ka	Bibtex	OpenCyc, NS	SUMO, M	SUMO, S
TEST 2	Biosphere	Top-bio	Dir, NS; Dir, S	SUMO, M	SUMO, M
TEST 3	Space	Geofile	OpenCyc, NS	SUMO, M; OpenCyc, M	OpenCyc, M
TEST 4	Restaurant	Food	OpenCyc, NS	OpenCyc, S; OpenCyc, M	OpenCyc, S; OpenCyc, M
TEST 5	MPEG7	Subject	OpenCyc, NS	OpenCyc, M; DOLCE, M	SUMO, NS; OpenCyc, NS
TEST 6	Travel	Vacation	Dir, NS; Dir, S	SUMO, M	SUMO, M
TEST 7	Resume	Agent	SUMO, NS	OpenCyc, M	OpenCyc, M
TEST 8	Resume	HL7_RBAC	OpenCyc, NS	SUMO, M	SUMO, NS; OpenCyc, NS; OpenCyc, S
TEST 9	Ecology	Top-bio	Dir, NS; Dir, S	SUMO, M	Dir, NS; Dir, S; SUMO, M; OpenCyc, M
TEST 10	Vertebrate	Top-bio	OpenCyc, NS	OpenCyc, S; OpenCyc, M	OpenCyc, S

Table 2: Algorithm that gives the best precision, recall and F-measure in each test

	SUMO, NS	**OpenCyc, NS**	**DOLCE, NS**	**SUMO, S**	**OpenCyc, S**	**DOLCE, S**	**SUMO, M**	**OpenCyc, M**	**DOLCE, M**
Prec.	0.097	0.157	-0.145	-0.021	-0.004	-0.162	-0.089	-0.070	-0.158
Rec.	-0.013	-0.019	-0.027	0.028	0.030	-0.031	0.066	0.070	0.020
F-m.	-0.011	-0.018	-0.049	0.000	0.014	-0.053	0.016	0.024	-0.023

Table 3: Average advantage in using upper ontologies