Automatically Extending, Pruning and Trimming General Purpose Ontologies

Roberto Navigli
Dipartimento di Informatica
Università di Roma "La Sapienza"
Via Salaria, 113, 00198 Roma, Italia
navigli@dsi.uniroma1.it

Abstract **A domain ontology represents a shared understanding of a given sector of reality (for instance, mathematics, economics, tourism etc.). Many application fields like Information Retrieval, Information Extraction and so on, as long as the Semantic Web [2], the next Web generation, need this kind of structured domain knowledge in order to add the missing semantic layout.**
However, an accurate search through the Internet shows the lack of large domain ontologies available to the community. In fact, building such knowledge resources requires big efforts in terms of time, costs and work due to the difficulty in identifying and properly defining domain concepts and their inter-relationships. One primary problem in this process is to establish an appropriate *is-a* hierarchy for the ontology. To this end, general-purpose lexical resources like WordNet [3] can be of help because they code a massive, although non-specific, quantity of knowledge.
This paper shows an original solution to the problem of building an *is-a* hierarchy for a domain ontology. This is achieved through the automatic enrichment and reorganization of the WordNet hierarchy by properly adding domain knowledge structured in the form of concept trees.

I. INTRODUCTION

The Semantic Web [2] is the next Web generation, that is a knowledge-based web of documents in a machine-readable form. In this vision, the semantics underlying data are explicitly represented. To this end, documents refer to a set of available, structured knowledge resources called *ontologies*.
An ontology is a *shared understanding* of some domain of interest [14]. In other words, an ontology is an explicit, agreed specification about a shared conceptualization.
Ontologies may have different degrees of formality but they must necessarily include a vocabulary of terms with their meaning (i.e., definitions) and their relationships.
Building ontologies is a difficult process that involves specialists from several fields. Philosophical ontologists and Artificial Intelligence logicists are usually involved in the task of defining the basic kinds and structures of concepts (objects, properties, relations, and axioms) that are applicable in every possible domain. The issue of identifying these very few "basic" principles, referred to as the *Top Ontology* (TO), is not a purely philosophical one, since there is a clear practical need of a model which has as much generality as possible, to ensure reusability across different domains [13].
Domain modelers and knowledge engineers are involved in the task of identifying the key domain conceptualizations, and describing them according to the organizational *backbones* established by the Top Ontology. The result of this effort is referred to as the *Upper Domain Ontology* (UDO), which usually includes a few hundred application-domain concepts.
While many ontology projects eventually succeed in the task of defining an Upper Domain Ontology[1], populating the third level, that we call the *Specific Domain Ontology* (SDO), is the actual barrier that very few projects can overcome (e.g. Wordnet [3], Cyc [8] and EDR [16]) at the price of inconsistencies and limitations.

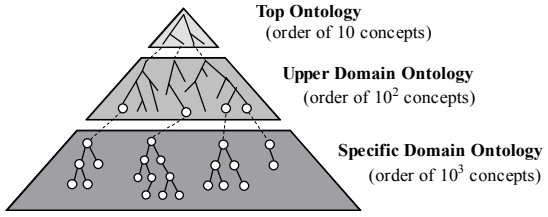

Figure I.1 The three levels of generality of a Domain Ontology.

It turns out that, although domain ontologies are recognized as crucial resources for the Semantic Web, in practice they are not available, and, when available, they are not used outside specific research environments[2].
In our recent work [12], as well as in [9] and [15], the third level, that is the Specific Domain Ontology, is learned in a semi-automatic manner. Figure I.2 shows the architecture of our system, *OntoLearn*, consisting of three main phases: first, a domain terminology is *extracted* from available texts in the application domain (specialized web sites and warehouses, or documents exchanged among members of a virtual community), and *filtered* using natural language processing [1] and statistical techniques. Second, terms are *semantically interpreted* using WordNet [3], a general-purpose lexical resource coding a massive quantity of non-specific knowledge, and Semcor [11], a corpus of semantically annotated sentences. Third, concepts are structured according to *taxonomic relations*, generating a *Domain Concept Forest* (hereafter DCF). The acquired DCF is therefore used to populate the SDO.
However, due to consensus problems among domain experts, an Upper Domain Ontology may also be missing. In this case, the WordNet hierarchy can be properly adjusted in order to fill the gap. Besides, the Specific

[1] In fact many ontologies are already available on the Internet including a few hundred more-or-less extensively defined concepts.
[2] For example, Wordnet is widely used in the Computational Linguistics research community, but large scale IT applications based on WordNet are not available.

Domain Ontology can be used to extend the hierarchy by adding the proper semantic connections (as shown in figure I.1) in a semi-automatic manner.

This paper shows an original method to create a domain hierarchy by making use of WordNet and a domain concept forest. This is achieved in three phases: WordNet extension through the attachment of domain trees (section II), pruning of dead branches (section III) and hierarchy trimming by deleting little informative concept nodes (section IV). Experimental results are discussed in section V. Finally, section VI presents some conclusions.

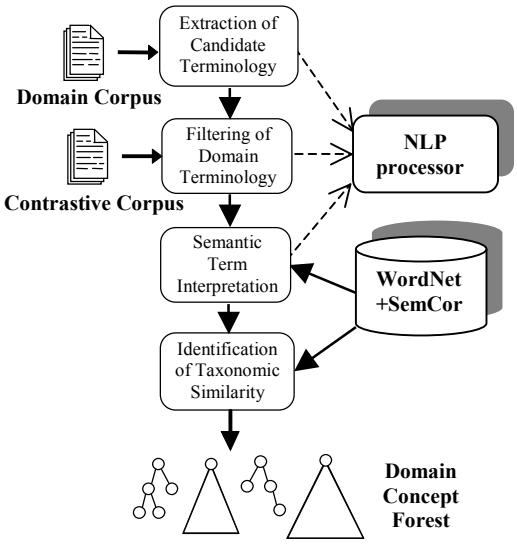

Figure I.2. Architecture of the OntoLearn system.

II. EXTENDING WORDNET

WordNet is a large lexical knowledge base whose popularity is recently growing even outside the computational linguistic community. In WordNet, a word sense is uniquely identified by a set of terms called *synset*, the equivalent of concepts in formal ontologies (e.g., for the sense #3 of *transport*: { *transportation#4*, *shipping#1*, *transport#3* }), and a textual definition called *gloss* (e.g. "*the commercial enterprise of transporting goods and materials*"). Synsets are taxonomically structured in a lattice, with a number of "root" concepts called *unique beginners* (e.g., { *entity#1*, *something#1* }). WordNet includes over 120,000 words (and over 170,000 synsets), but very few domain terms: for example, *transport* and *company* are individually included, but not *transport company* as a unique term.

Wordnet codes various semantic and lexical relations like *hyperonymy* (a car *isa* a vehicle), *hyponymy* (its inverse), *meronymy* (a room *has-a* a wall), *holonymy* (its inverse), *pertainymy* (dental *pertains-to* tooth), *attribute* (dry *value-of* wetness), *similarity* (beautiful *similar-to* pretty).

The WordNet hierarchy can be extended by carefully attaching the domain concept trees belonging to the SDO. These domain trees can be built in either a manual or an automatic way. Automatic methods are described in [12] and [15]. In [15], a domain terminology is extracted and then hierarchically organized in concepts by simple string inclusion (like in figure II.1). In [12], the automatic extraction of a domain terminology is followed by a step of semantic interpretation of terms. In both cases, a domain tree is an *isa* hierarchy of concepts rooted at a very basic domain concept. An example of domain concept tree is illustrated in figure II.2. As shown in the figure, it is reasonable to suppose that each domain concept is assigned at least one term[3] (i.e., one or more words, like *telephone number* or *travel agent*).

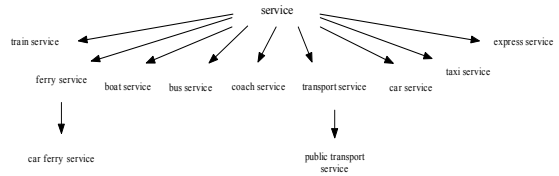

Figure II.1 A lexicalized tree in a Tourism Domain.

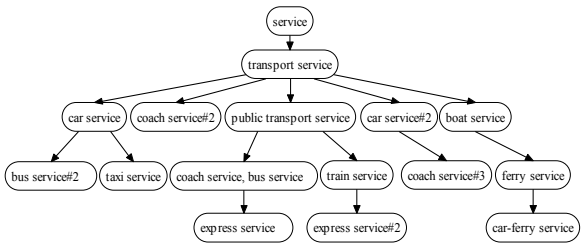

Figure II.2. An example of domain concept tree.

Here we present an automatic procedure for mapping each domain tree root to the right WordNet node (that is, to the right *synset* for that concept, as sketched in figure II.3). This is a very delicate matter because choosing the wrong sense, that is the wrong collocation for the root in the hierarchy, would also affect all its descendants. However, as long as the automatic procedure shows a good precision, domain experts can check the results in order to make the necessary adjustments.

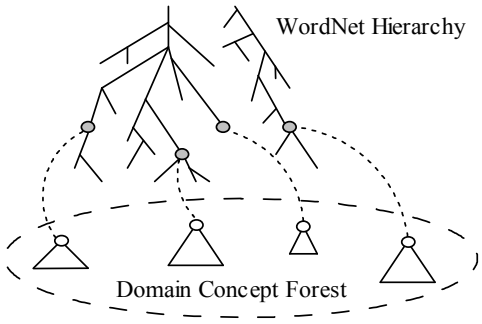

Figure II.3. A mapping among domain tree roots (in white) and concept nodes (in grey) chosen in the WordNet hierarchy.

[3] Of course this does not imply any string inclusion among the terms associated to a hyperonym and its hyponym (for example, *swimming pool* can be a hyponym of *hotel facility*). Also note that different concepts can be assigned the same term in case of polysemy within the domain.

Let F be the forest of our domain trees and let T be one of them. We make the following considerations:

- Properly attaching a domain tree T to the WordNet hierarchy is equivalent to disambiguating its root wrt WordNet;
- The *context* C_r of root $r \in T$ is given by the set of all other domain roots (although it can be extended with all descendants of r in the domain tree);
- For each term t in C_r, and for each sense S of t in WordNet[4], a semantic net can be built using the following relations: *hyperonymy*, *hyponymy*, *meronymy*, *holonymy*, *pertainymy*, *attribute*, *gloss* and *topic*[5]; to reduce the semantic net size, only concepts are a distance not greater than 3 are included (see figure II.4 for an example).
- The root r can be disambiguated by exploiting its context C_r.

This brings to the following procedure:

for each sense R of r **do**
  $\underline{score}_R := \underline{0}$ { initially, $\underline{score}_R$ is the null vector}
for each term $t \in C_r$ **do**
  for each sense S of t in WordNet[6] **do**
    for each sense R of r in WordNet **do**
      Calculate the score vector $\underline{v}$
      for $SN(S) \cap SN(R)$
      $\underline{score}_R := \underline{score}_R + \underline{v}$

The intersection between two semantic nets is assessed with a score vector, whose components are incremented whenever certain heuristics are matched (for instance, chains of hyperonymy/meronymy relations, parallelism etc.)[7].
At each inner step, the score calculated is summed to the total score vector for the sense of r involved in the intersection. At the end of the procedure, the maximum vector score (according to a lexicographic ordering) determines the sense chosen for r.
As WordNet is very fine grained ([7] and [5]) and because of the impossibility of reaching a large consensus on concept disambiguation [6], it is not uncommon that more than one choice be considered correct. This is taken into account in section V.

[4] If t is composed, the step is repeated for each subterm (usually one or two words) of t.
[5] The *gloss* and the *topic* relation are obtained parsing respectively the WordNet concept definitions and SemCor sentences including the sense in exam.
[6] For composed terms, the semantic nets associated to their leftmost subterms are considered first. If no result is found (that is, the intersection is empty) the other semantic nets are taken into account, moving from left to right.
[7] For a more detailed description of the method, refer to [12]. Some heuristics are inspired by the work presented in [4] and [10].

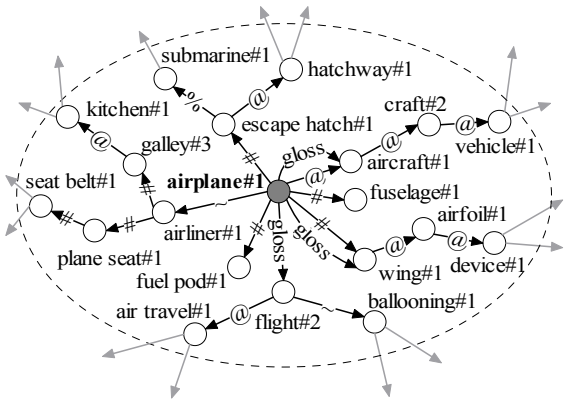

Figure II.4 The semantic net for sense 1 of *airplane* in WordNet.

As illustrated in figure II.5, associating a concept node in the WordNet hierarchy to each domain root concept creates a bipartition of WordNet nodes. In fact, those nodes not intersecting any path from a domain root to its WordNet unique beginner are isolated, suggesting that they can be pruned, as described in the next section.

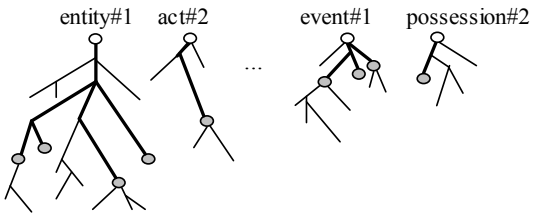

Figure II.5. Four of the nine WordNet noun hierarchies. Branches in bold highlight the paths from the domain root concepts (grey nodes) to the respective WordNet unique beginners (white nodes).

III. Pruning the Hierarchy

Pruning is achieved by erasing all the dead branches, that is all those nodes belonging to no path from a domain root node to its WordNet beginner. The effect of this operation is to delete nodes that are not related to the considered domain. Figure III.1 shows an example of hierarchy pruning.
Note that it is probable, although not mandatory, that brothers of domain roots in the WordNet hierarchy be also domain concepts. This helps find possible mistakes or omissions in the choice of domain root concepts. For instance, in the mathematical domain, the concepts of *integral#1* and *derivative#1* are both hyponyms of *computation#2* in WordNet, so, if one of them is not included in the domain forest, the procedure can warn the domain experts and help them enrich the ontology.
Furthermore, it may happen that significant descendants of root concepts in the WordNet hierarchy be not included in the respective domain tree. These considerations make it clear that WordNet can help fill the gap due to human arbitrariness or automatic mistakes during the creation of domain ontologies.

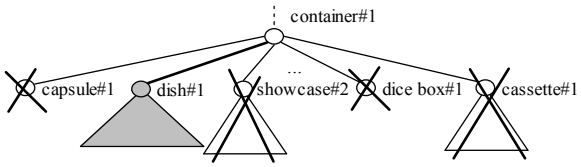

Figure III.1. An example of pruning in the catering domain. All non-domain brothers of *dish#1* in WordNet, as long as their descendants, can be deleted.

IV. TRIMMING THE HIERARCHY

After the simple pruning process, the WordNet hierarchy is trimmed, by deleting those nodes which are useless, redundant or too fine-grained.
Starting from each domain root concept *R*, all paths to its WordNet unique beginner are considered. The algorithm is the following:

{ the queue starts with the hyperonyms of R }
Q := *Hyperonyms*(*R*)
S := *R*
while *Q* <> ∅
  H := *Pop*(*Q*) { get a hyperonym }
  if (*H* has no brother **and** |*Hyponyms*(*H*)|=1 **and**
    H is not a domain concept **and** *H* ∉ *TopOntology*)
  then
    { delete *H* from the hyperonym set of *S* }
    Hyperonyms(*S*) :=
      Hyperonyms(*S*) \ { *H* } ∪ *Hyperonyms*(*H*)
    { cut *H* from the hyponym set of H's hyperonyms }
    for each hyperonym *H'* of*H*
      Hyponyms(*H'*) = *Hyponyms*(*H'*) \ { *H* } ∪ { *S* }
    S := *H* { move up through the hierarchy }
  Add(*Q*, *Hyperonyms*(*S*))

where *TopOntology* is the set of all nodes at a depth ≤ 2 in WordNet (that is, the unique beginners and their hyponyms), although this threshold can be extended.
The algorithm starts from *R* and moves up through the hierarchy by implementing a breadth first search (BFS). At each level, it chooses to delete each node for which the following four conditions hold together:

1. it has no brother;
2. it has one and only one hyponym;
3. it does not belong to the domain concept set;
4. it is not in the WordNet "top ontology" (this condition can be considered equivalent to: it has depth > 2).

Condition (1) prevents the algorithm from flattening the hierarchy (see figure IV.1). Condition (2) must hold because a node with more than one hyponym is surely valuable, as it collocates at least two nodes under the same concept; conversely, a node with only one hyponym gives no additional information and provides no further classification. Condition (3) is trivial: no domain node can be deleted. Condition (4) is also quite intuitive: nodes very high in the hierarchy represent the essential core of abstract concepts that cannot be deleted.
When a concept node *H* is deleted, all connections to the node are updated, that is:

- In the set of its hyponym's hyperonyms, *H* is replaced with its hyperonyms;
- In the set of its hyperonyms' hyponyms, *H* is replaced with its only hyponym, that is *S*.

An example of trimming is illustrated in figure IV.1.
One important consideration concerns the size of the domain ontology. In fact, the bigger the ontology is, the less the WordNet hierarchy is trimmed. This is due to the fact that a domain ontology containing many concepts fits very well in a part of the WordNet hierarchy, connecting to most of its branches.

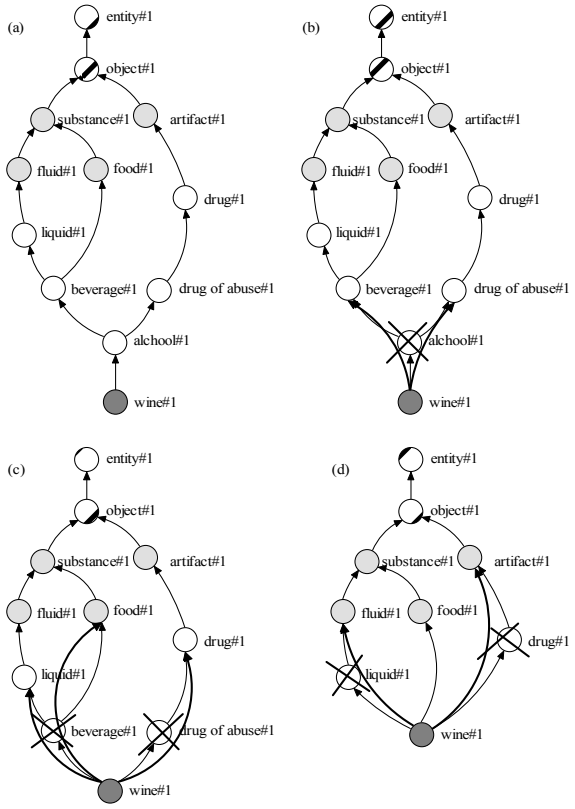

Figure IV.1. The four steps necessary for trimming the ancestors of the concept *wine#1* (in dark grey). If condition (1) missed, after step (d) it would be possible to delete the nodes in light grey thus allowing *wine#1* to be a direct hyponym of *object#1*. Nodes at a depth ≤2 in the WordNet hierarchy are shaded.

V. EXPERIMENTAL RESULTS

Starting from 2,156 concepts about the tourism domain belonging to 539 domain trees (a result of our work in [12]), we evaluated the precision of the automatic root disambiguation procedure presented in section II.
This was accomplished by manually attaching each of the 539 root concepts to a distinct WordNet synset. Although a certain factor of arbitrariness is unavoidable [6], the context gives the human taggers clear hints about the right senses for all of them (allowing to make multiple choices in case of uncertainty). Comparing the senses chosen by the procedure with the ones provided by the taggers led to a precision of 83.83%. A worse precision was achieved when

including all root descendants in the root context. The results give a clear evidence about the homogeneity of the root terms, as they expose strong interconnections within the domain, but makes it clear that subterms often refer to different senses of their domain ancestor (for example, *archaeological site* and *web site* refer to different senses of the *site* term; the same applies to *highway code* and *access code* etc.). The result is strongly dependent on the automatic method with which the domain concept trees were created. An accurate adjustment of root homonymy cases will be taken into account in our future work.

Finally we provide some data about the composition of the domain hierarchy after the various steps. Initially, WordNet 1.6 contains about 66,000 noun concepts. As a consequence of the pruning step, WordNet is reduced to overall 596 nodes (excluding the root domain nodes). Then, trimming the hierarchy results in the deletion of 116 nodes. So, the final hierarchy is composed of 480 non-specific nodes, 539 domain root nodes mapped to as many synsets and the remaining 1,617 domain nodes (figure V.1).

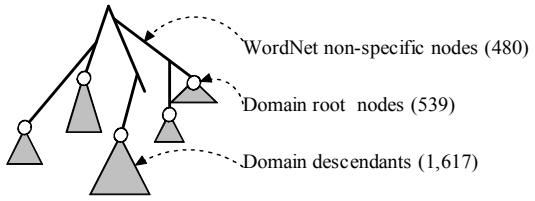

Figure V.1 The final hierarchy. WordNet nodes are represented as branches, domain root nodes as white nodes and their descendants as grey sub-trees.

VI. Conclusions

The original method presented in this paper builds a complete *isa* hierarchy for a domain ontology by fully exploiting WordNet and a domain concept forest. As in many other works, here again WordNet shows its usefulness in those tasks where a massive, structured and non-specific knowledge can help fill the gap left by the lack of domain ontologies.

Future directions of our work include the enrichment of the domain ontology with other semantic and thematic relations as long as its extensive use in fields like Information Retrieval, Information Extraction and Document Classification in order to show its valuable and vital contribution to the Semantic Web.

VII. Bibliography

[1] Basili R. Pazienza M.T and Velardi P., 1996. *An Empirical Symbolic Approach to Natural Language Processing*. Artificial Intelligence, n. 85.

[2] Berners-Lee, T., 1999. *Weaving the Web*. Harper, San Francisco.

[3] Fellbaum, C., 1995. *WordNet: an electronic lexical database*. Cambridge, MIT press.

[4] Harabagiu, S., and Moldovan D., 1999. *Enriching the WordNet Taxonomy with Contextual Knowledge Acquired from Text*. AAAI/MIT Press.

[5] Kilgariff, A., 1997. "*I don't believe in word senses*". Technical Report, ITRI-97-12. University of Brighton.

[6] Kilgariff, A., 1998. *Gold standard datasets for evaluating Word Sense Disambiguation programs*. Technical Report, ITRI-98-08. University of Brighton.

[7] Krovetz, R., 1997. *Homonymy and polysemy in Information Retrieval*. In Proceedings of ACL/EACL '97.

[8] Lenat, D.B., 1993. *CYC: a large scale investment in knowledge infrastructure*, in Communication of the ACM, vol. 3, N. 11.

[9] Maedche, A.., Volz., 2001. R. *The Ontology Extraction and Maintenance Framework Text-To-Onto* In Proceedings of ICDM'01: The 2001 IEEE International Conference on Data Mining, San Jose, California, USA.

[10] Milhalcea, R. and Moldovan. D., 2001. *eXtended WordNet: progress report*. NAACL 2001 Workshop on WordNet and Other Lexical Resources, Pittsbourgh, June 2001.

[11] Miller, G.A., Leacock, C., Tengi, R., and Bunker R. T., 1993. *A Semantic Concordance*. Proceedings of the ARPA WorkShop on Human Language Technology. San Francisco, Morgan Kaufman.

[12] Missikoff, M., Navigli, R., Velardi, P., 2002. *The Usable Ontology: An Environment for Building and Assessing a Domain Ontology*. In Proceedings of The International Semantic Web Conference, Sardinia, Italy, June 2002.

[13] Smith, B. and Welty, C., 2001. *Ontology: towards a new synthesis*, Formal Ontology in Information Systems, ACM Press, (2001).

[14] Uschold M., M. Gruninger, 1996. *Ontologies: Principles, Methods and Applications*. The Knowledge Engineering Review, vol. 11, n. 2.

[15] Vossen, P., 2001. *Extending, trimming, fusing WordNet for technical documents*. In Proceedings of the NAACL workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations. Pittsburg, USA.

[16] Yokoi T., 1993. *The EDR electronic dictionary*, Communications of the ACM, vol. 38, N. 11.