Commonsense Ontologies and the Use of Words in Natural Language

Ruth Janning

FernUniversität in Hagen
Fakultät für Mathematik und Informatik
ruth.janning@fernuni-hagen.de

Abstract: Since the appearance of the 'Semantic Web' and the development of 'RDF' and 'OWL', ontologies gained new importance in computer science. Ontological structures can be used to make knowledge available to artificial intelligent systems. But such systems need commonsense knowledge to simulate human reasoning beyond the boundaries given by specific domains. For this purpose commonsense ontologies are employed. However, existing commonsense ontologies, e.g. Cyc [Cy10], were constructed over a lengthy period of time. An interesting proposal to reach this in shorter time with less effort is to expose the structure of commonsense knowledge by analyzing the use of words in natural language. Based on this, a method to automatically gain commonsense ontologies with less effort will be presented. The main point of this method is the automatization. For this purpose data mining techniques are applied and an algorithm to generate the resulting ontology out of the gained data is introduced.

1 Exposing the structure of commonsense knowledge

To avert the immense effort[1] for the construction of commonsense ontologies, there is need for new methods to gain these ontological structures. Saba [Sa07] proposes to expose – and not newly develop – the structure of commonsense knowledge by analyzing the use of words in natural language. According to this, natural language should guide the process of gaining a commonsense ontology.

2 Four steps toward an ontological structure

Asking a child, if it makes sense to say 'a dog barks' or 'a book barks', one gets an explicit positive or negative answer. On this simple base – the view of a child – the analysis of words is executed.

[1] Knowledge was fed over 20 years into the knowledge base of Cyc [Cy10] until the system was able to learn by itself [Wi05].

Accordingly, quantitative or philosophic assessments, which an adult would maybe have in mind (e.g. there are dogs that do not bark or such dogs who bark louder or lower), are not regarded. So it is a binary decision if it in general makes sense to say 'a dog barks' or not.

Introducing a predicate *App(p,c)* [Sa07], one can describe a process which guides within four steps toward an ontological structure (see Figure 1). In this expression p denotes a property (adjective) or action (verb) and c a concept (noun)[2]. *App(p,c)* receives the value *true* if it makes sense to speak of the property or action p of c.[3]

1. Assume a set of concepts (nouns) $C = \{c_1, \ldots, c_m\}$ and a set of properties (adjectives) or actions (verbs) $P = \{p_1, \ldots, p_n\}$ to be already known.

2. Furthermore a predicate *App(p,c)*, $c \in C$ and $p \in P$, is given. *App(p,c)* becomes *true* if the property or action p is reasonably applicable to objects of type c (i.e. if it makes sense to speak of the property or action p of c).

3. For every property or action $p \in P$ a set $C_p = \{c \mid App(p,c)\}$, which includes all concepts c for which *App(p,c)* is *true*, is generated.

4. As a result, the desired hierarchy is gained through an analysis of the subset relationship between the sets generated in step 3.

Figure 1: Four steps toward an ontological structure.

3 Realizing and automatizing the four steps

This process of four steps proposed by Saba [Sa07] is an interesting beginning and should be further developed. Now a suggestion for realizing and especially for automatizing the four steps will follow.

3.1 Step 1 and 2: Choice of the sets and evaluation of the combinations

To automatize step 1 and 2, i.e. the choice of the sets of nouns and adjectives or verbs and the decision, whether a specific combination makes sense, data mining techniques should be used. That is, a computer searches in given texts (e.g. from the numerous sources of the Internet like Wikipedia [Wi10a]) for nouns with a preceding adjective or following verb.[4] To detect the nouns, adjectives and verbs, the computer uses a computerized dictionary (e.g. WordNet [Pr10] or Wiktionary [Wi10b]).

[2] Nouns are supposed to be plain base concepts.
[3] If a child answered 'yes' to the question whether this combination makes sense.
[4] Auxiliary verbs should be excluded. For words with the same character string, which can be nouns on the one hand and adjectives on the other hand (e.g. 'human'), the meaning of the first discovered word is used. The other meaning, respectively combinations with it, will be ignored in the following.

The discovered words (e.g. as given in Figure 2) are inserted in alphabetical order into a table (see Figure 3). The nouns occur in the first column (down from the second row) and the adjectives and verbs in the first row (except the first cell of the first column). The combinations of nouns and adjectives or verbs are explored with data mining by checking how often some combinations occur (checking the *support* of *associations* [BKI06]). Often occurring combinations (with higher support than other combinations)[5] get a '+' in the corresponding table cell (that means *App(p,c)* is *true* for the combination of *p* and *c*). Otherwise – if the support has a value below the determined threshold – the table cell gets the symbol '-' (see Figure 3).

C	=	{bird, book, dog, machine, man, woman}
P	=	{bark, exist, defective, fly, live, pregnant, read}

Figure 2: Example sets.

	bark	**exist**	**defective**	**fly**	**live**	**pregnant**	**read**
bird	-	+	-	+	+	-	-
book	-	+	-	-	-	-	-
dog	+	+	-	-	+	-	-
machine	-	+	+	-	-	-	-
man	-	+	-	-	+	-	+
woman	-	+	-	-	+	+	+

Figure 3: Example table.

3.2 Step 3 and 4: Obtaining concept sets and generating a concept hierarchy

The sets C_p from step 3 can be obtained by picking out the table column corresponding to *p*. A set C_p contains all nouns of which the relevant table cell holds the symbol '+' (see Figure 4). Step 4 can be realized by an algorithm (see Figure 5). The root node of the desired hierarchy is the set which contains all considered concepts (nouns). The algorithm first chooses from the sets C_p the largest proper subset of the root node set and inserts this subset as left son of the root into the hierarchy. The right son node is the complementary set of the left son. Subsequently the algorithm chooses the largest subsets of both gained son nodes and their son nodes and so on until there are no more subsets other than the empty set. The reason for choosing the largest subset is that the sets, which represent nodes, should become smaller downward. That is, they should become more and more specific and represent more specific concepts. The result is an ontological structure like the one shown in Figure 6. In such a hierarchy all son nodes and descendants are sub concepts of their father node.

The obtained hierarchy causes relations between the different concepts and allows to derive rules, e.g. in first-order logic[6], like

$$\forall c(App(bark,c) \Rightarrow \neg App(read,c))$$

[5] Of course a reasonable threshold has to be chosen.
[6] In this paper no specific ontology representation language is mentioned since the method is yet a conception.

which means that for objects, which can bark, it does not make sense to say they read (in short: dogs cannot read).

C_{exist}	=	{bird, book, dog, machine, man, woman}	(complementary set: {})
C_{live}	=	{bird, dog, man, woman}	(compl. set: {book, machine})
C_{read}	=	{man, woman}	(compl. set: {bird, dog})
$C_{defective}$	=	{machine}	(compl. set: {book})
$C_{pregnant}$	=	{woman}	(compl. set: {man})
C_{bark}	=	{dog}	(compl. set: {bird})
C_{fly}	=	{bird}	(compl. set: {dog})

Figure 4: Sets C_p.

Method: generateHierarchy()
Given: Set which contains all concepts.
Output: Concept hierarchy.
Algorithm:

1. Initialization of the root node of the concept hierarchy with the set which contains all concepts.
2. Call of the recursive method getSubSets(<*set*>) with the set of the root node as its argument:
   getSubSets(*root node set*)
3. **return** concept hierarchy

Method: getSubSets(<*set*>)
Given: Discovered sets C_p of concepts (see above step 3).
Input: Set of which the largest subset and its complement are to be determined and inserted as left and right son.
Algorithm:

1. Find all proper subsets of <*set*>.
2. Determine the largest set of these subsets (if there are more than one, choose those, of which the related property or action is alphabetically smaller than all the others).
3. *LeftSonNode* = determined largest set.
4. *RightSonNode* = <*set*> \ *LeftSonNode*.
5. **if** *LeftSonNode* $\neq \emptyset$
   then
   - Insert into the hierarchy *LeftSonNode* as left son of the node which is represented by <*set*>.
   - Insert into the hierarchy *RightSonNode* as right son of the node which is represented by <*set*>.
   - Determine the subsets of the left son: getSubSets(*LeftSonNode*).
   - Determine the subsets of the right son: getSubSets(*RightSonNode*).

Figure 5: Algorithm for the concept hierarchy generation (step 4).

The presented algorithm seems to be similar to the *Formal Concept Analysis* (FCA) [GW96] but the procedure as well as the result is different in both methods. Contrary to FCA, in this algorithm a concept is composed of only one attribute (adjective or verb) and a set of objects (nouns). Additionally this algorithm produces exclusively binary trees and in the given example the hierarchy generated by FCA is much shallower.

An efficient *approach to learning taxonomies or concept hierarchies from text*, which uses FCA and creates domain ontologies, is presented in [CHS04].

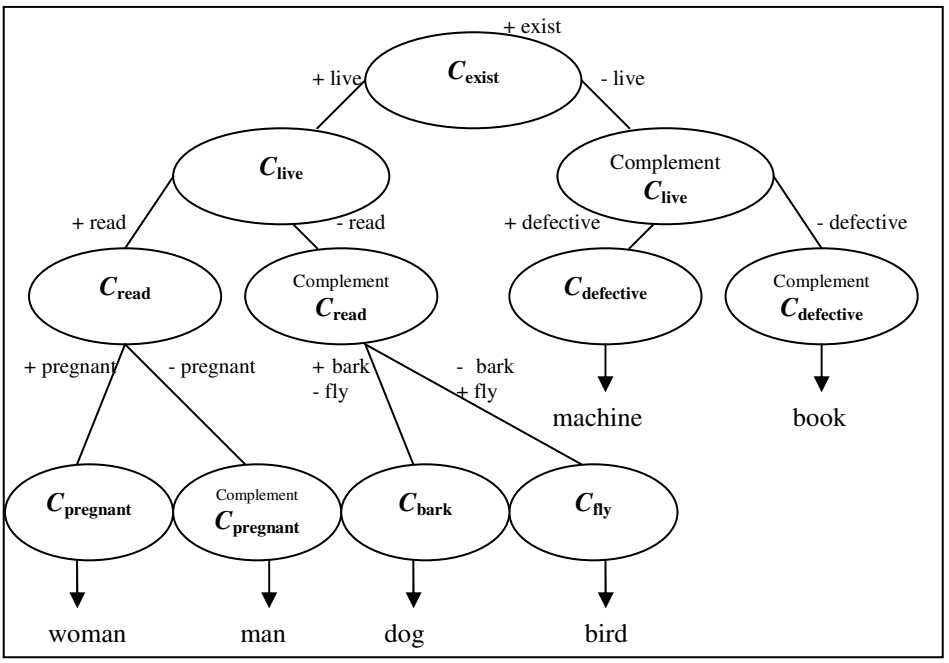

Figure 6: Generated ontological structure.

An oval stands for a set of nouns, respectively for a concept, which includes all the sub concepts represented by the nouns. An edge symbolizes the relationship between the two linked concepts: The concept below includes all nouns, to which the considered property or action is applicable. So C_{read} contains all concepts from C_{live} (existing living objects), which possess the capability to read. The arrows indicate concepts from sets, which possess just one element.

4 Related work

The proposed method is also similar to the underlying ideas of the platform OLE [NS05] since OLE deals with *automatic acquisition of semantic relations from texts* [NS05]. But there are some differences between the two methods. OLE is a *platform for bottom-up generation and merging of ontologies* and uses a *pattern-driven concept extraction process* based on proposals by M. A. Hearst [He92]. The method presented here is a kind of top-down clustering. Moreover it searches directly for nouns, adjectives and verbs in the given texts and uses data mining techniques. Also the OntoLearn system [NVM04] aims at *extracting knowledge from electronic documents to support the rapid construction of a domain ontology*. But its machine concept learning is an iterative process and is initially *based on the use of external, generic knowledge sources*. So it uses WordNet [Pr10] *as a start-up ontology*. Furthermore, OntoLearn uses statistical techniques.

5 Conclusions

The presented method is a good beginning, but there are still some problems. To get a useable ontological structure for commonsense knowledge, a very large number of nouns and adjectives or verbs need to be cumulated. In addition to this, these words must be sufficiently different from each other to gain a full commonsense ontology. Every concept of the resulting hierarchy must possess a unique characteristic to be different from other concepts. A further problem results from the need to separate original and metaphorical meanings of words, which must appear at different places in the hierarchy. This is not avoidable, if one wants to get an unambiguous and reasonable ontological structure. But in normal texts words are used in their original meaning as well as in metaphorical meanings. Possibly the usage of data mining techniques is already a solution. These methods give a '+' only to combinations of substantives and verbs or adjectives, which reach a high support. So if we suppose that, compared to the original meaning, metaphorical meanings occur scarce in texts from lexicons like Wikipedia [Wi10a], they will be ignored. Finally, the presented method to automatically gain commonsense ontologies is yet a conception. The next steps will be: completely implementing the method and executing empirical tests.

Bibliography

[BKI06] Beierle, C.; Kern-Isberner, G.: Data Mining und Wissensfindung in Daten. In: Methoden wissensbasierter Systeme. 3. Vieweg Verlag, 2006; chapter 5.5, pp. 141-154.

[CHS04] Cimiano P.; Hotho A.; Staab S.: Clustering Ontologies from Text. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC). 2004; pp. 1721-1724.

[Cy10] Cycorp Inc.: Cycorp, Inc.. http://www.cyc.com/.

[GW96] Ganter, B.; Wille R.: Formale Begriffsanalyse: mathematische Grundlagen. Springer-Verlag, 1996.

[He92] Hearst, M. A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics. Association for Computational Linguistics, Morristown, NJ, USA, 1992; pp. 539-545.

[NS05] Nováček, V.; Smrž, P.: OLE - A New Ontology Learning Platform. In: Proceedings of International Workshop on Text Mining Research, Practice and Opportunities. Incoma Ltd., 2005; pp. 12-16.

[NVM04] Navigli, R.; Velardi, P.; Missikoff, M.: Web Ontology Learning and Engineering: An Integrated Approach. In (Zhong N.; Liu J. Eds.): Intelligent technologies for information analysis. Springer-Verlag, 2004; chapter 10, pp. 223-242.

[Pr10] Princeton University: About WordNet. http://wordnet.princeton.edu/.

[Sa07] Saba, Walid S.: Language, logic and ontology: Uncovering the structure of commonsense knowledge. In: International Journal of Human-Computer Studies 65, March 2007; pp. 610-623.

[Wi05] Witbrock, M.; Matuszek, C.; Brusseau, A.; Kahlert, R.C.; Fraser, C.B.; Lenat, D.B.: Knowledge Begets Knowledge: Steps towards Assisted Knowledge Acquisition in Cyc. In: Papers from the 2005 AAAI Spring Symposium on Knowledge Collection from Volunteer Contributors (KCVC). Stanford, California, 2005; pp. 99-105.

[Wi10a] Wikimedia Foundation, Inc.: Wikipedia. http://www.wikipedia.org/.

[Wi10b] Wikimedia Foundation, Inc.: Wiktionary. http://wiktionary.org/.