Plugging Taxonomic Similarity in First-Order Logic Horn Clauses Comparison

S. Ferilli, M. Biba, N. Di Mauro, T.M.A. Basile, and F. Esposito

Dipartimento di Informatica
Università di Bari
via E. Orabona, 4 - 70125 Bari - Italia
{ferilli,biba,ndm,basile,esposito}@di.uniba.it

Abstract. Horn clause Logic is a powerful representation language exploited in Logic Programming as a computer programming framework and in Inductive Logic Programming as a formalism for expressing examples and learned theories in domains where relations among objects must be expressed to fully capture the relevant information. While the predicates that make up the description language are defined by the knowledge engineer and handled only syntactically by the interpreters, they sometimes express information that can be properly exploited only with reference to a taxonomic background knowledge in order to capture unexpressed and underlying relationships among the concepts described. This is typical when the representation predicates are not purposely engineered but rather derive from the particular words found in a text.

This work proposes the exploitation of a taxonomic background knowledge to better assess the similarity between two First-Order Logic (Horn clause) descriptions, beyond the simple syntactical matching between predicates. To this aim, an existing distance framework is extended by applying the underlying distance measure also to parameters coming from the taxonomic background knowledge. The viability of the solution is demonstrated on sample problems.

1 Introduction

First-Order Logic (*FOL* for short) is a powerful representation language that allows to express relationships among objects, which is often an unnegligible requirement in real-world and complex domains. Logic Programming [11] is a computer programming framework based on a FOL sub-language, which allows to perform reasoning on knowledge expressed in the form of Horn clauses. Inductive Logic Programming (ILP) [13] aims at learning automatically logic programs from known examples of behaviour, and has proven to be a successful Machine Learning approach in domains where relations among objects must be expressed to fully capture the relevant information. Many AI tasks can take advantage from techniques for descriptions comparison: subsumption procedures (to converge more quickly), flexible matching, instance-based classification techniques or clustering, generalization procedures (to focus on the components that are

R. Serra and R. Cucchiara (Eds.): AI*IA 2009, LNAI 5883, pp. 131–140, 2009.

more likely to correspond to each other). In FOL, this is a particularly complex task due to the problem of *indeterminacy* in mapping portions of one formula onto portions of another.

In the traditional approach, predicates that make up the description language are defined by the knowledge engineer that is in charge of setting up the reasoning or learning problem, and are uninterpreted by the systems. The knowledge engineer can also define and provide a background knowledge to be exploited in order to improve performance or effectiveness of the results. However, a particular kind of information that often needs to be expressed in the descriptions is taxonomic information, that can convey implicit relationships among the concepts described. Unfortunately, such a kind of information needs to be interpreted in order to be fully exploited, which requires a proper background knowledge to be set up. Unless the problem domain is very limited, the taxonomic background knowledge to be provided becomes huge: in these cases, the use of existing state-of-the-art taxonomies can be a definite advantage.

This work builds on previous results concerning a framework for similarity assessment beween FOL Horn clauses, where the overall similarity depends on the similarity of the pairs of literals associated by the least general generalization, the similarity of two literals in turn depends on the similarity of their corresponding arguments (i.e., terms), and the similarity between two terms is computed according to the predicates and positions in which they appear. Here, a novel and general approach to the assessment of similarity between concepts in a taxonomy is proposed, and its integration as an extension of the similarity framework for clauses including taxonomic information is described.

The rest of this paper is organized as follows. The next section identifies a sample problem/application in which defining a taxonomic similarity can be of help. Then, Section 3 introduces the basic formula and framework for the overall assessment of similarity between Horn clauses. Section 4 proposes an application of the same formula to compute the taxonomic similarity between two concepts or words, and introduces it in the previous framework. Section 5 shows experiments that suggest the effectiveness of the proposed approach. Lastly, Section 6 concludes the paper and outlines future work directions.

2 Why a Taxonomic Approach: Sample Problems

In this section, one of the many practical problems in which taxonomic information is present and relevant has been selected and discussed, in order to provide the reader with a better understanding of the concepts and methods presented in the paper. The same toy problem will be tackled later using the proposed method to show its behavior and viability. As already pointed out, setting up a general taxonomy is a hard work, for which reason the availability of an already existing resource can be a valuable help in carrying out the task. In this example we will refer to the most famous taxonomy available nowadays, WordNet (WN) [12], that provides both the conceptual and the lexical level.

First of all, let us show a case in which an effective taxonomic similarity assessment can be useful in itself. Consider the following words and concepts:

102330245 *mouse* (animal) : 'any of numerous small rodents typically resembling diminutive rats having pointed snouts and small ears on elongated bodies with slender usually hairless tails'
103793489 *mouse* (device) : 'a hand-operated electronic device that controls the coordinates of a cursor on your computer screen as you move it around on a pad; on the bottom of the device is a ball that rolls on the surface of the pad'
103082979 *computer* (device) : 'a machine for performing calculations automatically'
102121620 *cat* (pet) : 'feline mammal usually having thick soft fur and no ability to roar: domestic cats; wildcats'
102127808 *cat* (wild) : 'any of several large cats typically able to roar and living in the wild'
102129604 *tiger* (animal) : 'large feline of forests in most of Asia having a tawny coat with black stripes; endangered'
102084071 *dog* (pet) : 'a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds'
102374451 *horse* (animal) : 'solid-hoofed herbivorous quadruped domesticated since prehistoric times'
103624767 *horse* (chess) : 'a chessman shaped to resemble the head of a horse; can move two squares horizontally and one vertically (or vice versa)'

One might need to check how close two of such items are, in order to perform further processing such as logic deductions or Natural Language Processing. For instance, one might want to disambiguate a polysemous word (e.g., 'mouse') by comparing its candidate underlying concepts to the other concepts that are present in the same text (e.g., 'cat' and 'dog' rather than 'computer'). Or, one might be interested in ranking a set of candidate concepts by closeness with respect to a given concept (e.g., ranking 'dog (pet)', 'tiger (animal)' and 'cat (wild)' with respect to 'cat (pet)'), etc. etc.

Then, let us show an example in which the taxonomic similarity assessment can support other processes. Specifically, here we consider the problem of similarity assessment between natural language sentences represented by FOL Horn clauses. FOL might be exploited for representing relational features of natural language, such as the syntactic relationships among discourse components. Indeed, although much more computationally demanding than simple bag-of-word approaches traditionally exploited in the literature, techniques that take into account the syntactic structure of sentences are very important to fully capture the information they convey. Reporters know very well that, swapping the subject and the object in a sentence like "The dog bit the man", results in very different interest of the underlying news. Natural language typically requires huge taxonomic information, and the problems of synonimy and polisemy introduce further complexity. For instance, the following sentences:

1. "The boy wants a small dog"
2. "The girl desires a yellow canary"
3. "The hammer hits a small nail"

structurally exhibit the same grammatical pattern, thus no hint is available to assess which is more similar to which. Even worse, at the lexical level, the only

common word ('small') appears in sentences 1 and 3, which would suggest they are closer to each other than to sentence 2. However, it becomes clear that the first two are conceptually the most similar to each other as long as one knows and considers that 'boy' and 'girl' are two young persons, 'to want' and 'to desire' are synonyms and 'dog' and 'canary' are two pets.

For demonstration purposes, let us consider a simplified structural description language for natural language sentences:

subj(X,Y) : Y is the subject of sentence X
pred(X,Y) : Y is the predicate of sentence X
dir_obj(X,Y) : Y is the direct object of sentence X
ind_obj(X,Y) : Y is the in direct object of sentence X
noun(X,Y) : Y is a noun appearing in component X of the sentence
verb(X,Y) : Y is a verb appearing in component X of the sentence
adj(X,Y) : Y is an adjective appearing in component X of the sentence
adv(X,Y) : Y is an adverb appearing in component X of the sentence
prep(X,Y) : Y is a preposition appearing in component X of the sentence

Additionally, each noun, verb, adjective or adverb is described by the corresponding concept (or word) in the sentence, plus possible other properties expressed by ordinary unary predicates. For the three sentences reported above one gets:

```
s1 = sentence(s1) :- subj(s1,ss1), pred(s1,ps1), dir_obj(s1,ds1),
     noun(ss1,nss1), boy(nss1), verb(ps1,vps1), want(vps1),
     adj(ds1,ads1), small(ads1), noun(ds1,nds1), dog(nds1).
s2 = sentence(s2) :- subj(s2,ss2), pred(s2,ps2), dir_obj(s2,ds2),
     noun(ss2,nss2), girl(nss2), verb(ps2,vps2), desire(vps2),
     adj(ds2,ads2), yellow(ads2), noun(ds2,nds2), canary(nds2).
s3 = sentence(s3) :- subj(s3,ss3), pred(s3,ps3), dir_obj(s3,ds3),
     noun(ss3,nss3), hammer(nss3), verb(ps3,vps3), hit(vps3),
     adj(ds3,ads3), small(ads3), noun(ds3,nds3), nail(nds3).
```

Syntactically, the generalization between $s2$ and both $s1$ and $s3$ is:

```
sentence(X) :- subj(X,Y), noun(Y,Y1), pred(X,W), verb(W,W1),
               dir_obj(X,Z), adj(Z,Z1), noun(Z,Z2).
```

while the generalization between $s1$ and $s3$ is:

```
sentence(X) :- subj(X,Y), noun(Y,Y1), pred(X,W), verb(W,W1),
               dir_obj(X,Z), adj(Z,Z1), small(Z1), noun(Z,Z2).
```

so that the latter pair, having in the generalization an additional literal with respect to the former pairs, would appear to have a greater similarity value due to just the structural aspects, in spite of the very different content.

3 Similarity Framework

In [6], a framework for computing the similarity between two Datalog Horn clauses has been provided, which is summarized in the following. Let us preliminary recall some basic notions involved in Logic Programming. The *arity* of a

predicate is the number of arguments it takes. A *literal* is an n-ary predicate, applied to n terms, possibly negated. *Horn clauses* are logical formulæ usually represented in Prolog style as l_0 :- $l_1, \ldots, l_n$ where the l_i's are *literals*. It corresponds to an implication $l_1 \wedge \cdots \wedge l_n \Rightarrow l_0$ to be interpreted as "l_0 (called *head* of the clause) is true, provided that l_1 and ... and l_n (called *body* of the clause) are all true". Datalog [3] is, at least syntactically, a restriction of Prolog in which, without loss of generality [15], only variables and constants (i.e., no functions) are allowed as terms. A set of literals is *linked* if and only if each literal in the set has at least one term in common with another literal in the set. We will deal with the case of linked Datalog clauses. In the following, we will call *compatible* two sets or sequences of literals that can be mapped onto each other without yielding inconsistent term associations (i.e., a term in one formula cannot correspond to different terms in the other formula).

Intuitively, the evaluation of similarity between two items i' and i'' might be based both on parameters expressing the amounts of common features, which should concur in a positive way to the similarity evaluation, and of the features of each item that are not owned by the other (defined as the *residual* of the former with respect to the latter), which should concur negatively to the whole similarity value assigned to them [10]:

n , the number of features owned by i' but not by i'' (*residual* of i' wrt i'');
l , the number of features owned both by i' and by i'';
m , the number of features owned by i'' but not by i' (*residual* of i'' wrt i').

A similarity function that expresses the degree of similarity between i' and i'' based on the above parameters, and that has a better behaviour than other formulæ in the literature in cases in which any of the parameters is 0, is [6]:

$$sf(i', i'') = \mathrm{sf}(n, l, m) = 0.5\frac{l+1}{l+n+2} + 0.5\frac{l+1}{l+m+2} \tag{1}$$

It takes values in $]0, 1[$, which resembles the theory of probability and hence can help human interpretation of the resulting value. When $n = m = 0$ it tends to the limit of 1 as long as the number of common features grows. The full-similarity value 1 is never reached, being reserved to two items that are exactly the same ($i' = i''$), which can be checked in advance. Consistently with the intuition that there is no limit to the number of different features owned by the two descriptions, which contribute to make them ever different, it is also always strictly greater than 0, and will tend to such a value as long as the number of non-shared features grows. Moreover, for $n = l = m = 0$ the function evaluates to 0.5, which can be considered intuitively correct for a case of maximum uncertainty. Note that each of the two terms refers specifically to one of the two items under comparison, and hence they could be weighted to reflect their importance.

In FOL representations, usually terms denote objects, unary predicates represent object properties and n-ary predicates express relationships between objects; hence, the overall similarity must consider and properly mix all such components. The similarity between two clauses C' and C'' is guided by the

similarity between their structural parts, expressed by the n-ary literals in their bodies, and is a function of the number of common and different objects and relationships between them, as provided by their least general generalization $C = l_0 \text{ :- } l_1, \ldots, l_k$. Specifically, we refer to the θ_{OI} generalization model [5]. The resulting formula is the following:

$$\text{fs}(C', C'') = \text{sf}(k' - k, k, k'' - k) \cdot \text{sf}(o' - o, o, o'' - o) + \text{avg}(\{\text{sf}_s(l'_i, l''_i)\}_{i=1,\ldots,k})$$

where k' is the number of literals and o' the number of terms in C', k'' is the number of literals and o'' the number of terms in C'', o is the number of terms in C and $l'_i \in C'$ and $l''_i \in C''$ are generalized by l_i for $i = 1, \ldots, k$. The similarity of the literals is smoothed by adding the overall similarity in the number of overlapping and different literals and terms.

The similarity between two compatible n-ary literals l' and l'', in turn, depends on the multisets of n-ary predicates corresponding to the literals directly linked to them (a predicate can appear in multiple instantiations among these literals), called *star*, and on the similarity of their arguments:

$$\text{sf}_s(l', l'') = \text{sf}(n_s, l_s, m_s) + \text{avg}\{\text{sf}_o(t', t'')\}_{t'/t'' \in \theta}$$

where θ is the set of term associations that map l' onto l'' and S' and S'' are the stars of l' and l'', respectively:

$$n_s = |S' \setminus S''| \qquad l_s = |S' \cap S''| \qquad m_s = |S'' \setminus S'|$$

Lastly, the similarity between two terms t' and t'' is computed as follows:

$$\text{sf}_o(t', t'') = \text{sf}(n_c, l_c, m_c) + \text{sf}(n_r, l_r, m_r)$$

where the former component takes into account the sets of properties (unary predicates) P' and P'' referred to t' and t'', respectively:

$$n_c = |P' \setminus P''| \qquad l_c = |P' \cap P''| \qquad m_c = |P'' \setminus P'|$$

and the latter component takes into account how many times the two objects play the same or different roles in the n-ary predicates; in this case, since an object might play the same role in many instances of the same relation, the *multi*sets R' and R'' of roles played by t' and t'', respectively, are to be considered:

$$n_r = |R' \setminus R''| \qquad l_r = |R' \cap R''| \qquad m_r = |R'' \setminus R'|$$

4 Taxonomic Similarity

A lot of research has been devoted to develop and test similarity measures for concepts in a taxonomy (a survey for WN can be found in [2]). The most exploited relationship is generalization/specialization, relating concepts or classes to their super-/sub-concepts or classes. Various proposals are based on the length of the paths that link the concepts to be compared to their closest common ancestor, according to the intuition that, the closer a common ancestor of two concepts, the more they can be considered as similar to each other.

Since this work aims at extending a general similarity framework by considering taxonomic information, for compatibility and smooth integration we exploited the same function as the base framework[1]. In our case, (1) requires three parameters: one expressing the common information between the two objects to be compared, and the others expressing the information carried by each of the two but not by the other. If the taxonomy is a hierarchy, and hence can be represented as a tree, the path connecting any node (concept) to the root (the most general concept) is unique: given two concepts c' and c'', let us call $< p'_1, \ldots, p'_{n'} >$ the path related to c', and $< p''_1, \ldots, p''_{n''} >$ the path related to c''. Thus, given any two concepts, their closest common ancestor is uniquely identified, as the last element in common in the two paths: suppose this is the k-th element (i.e., $\forall i = 1, \ldots, k : p'_i = p''_i = p_i$). Consequently, three sub-paths are induced: the sub-path in common, going from the root to such a common ancestor ($< p_1, \ldots, p_k >$), and the two trailing sub-paths ($< p'_{k+1}, \ldots, p'_{n'} >$ and $< p''_{k+1}, \ldots, p''_{n''} >$). Now, the former can be interpreted as the common information, and the latter as the residuals, and hence their lengths $(n' - k, k, n'' - k)$ can serve as arguments (n, l, m) to apply the similarity formula. This represents a novelty with respect to other approaches in the literature, where only (one or both of) the trailing parts are typically exploited, and is also very intuitive, since the longest the path from the top concept to the common ancestor, the more they have in common, and the higher the returned similarity value.

Actually, in real-world domains the taxonomy is a heterarchy, not just a hierarchy, since by multiple inheritance a concept can specialize many other concepts. This is very relevant as regards the similarity criterion above stated, because the closest common ancestor and the paths linking two nodes are no more unique. Hence, many incomparable common ancestors and paths between concepts can be found, and going to the single common one would very often result in overgeneralization. Our novel solution to this problem is computing the whole set of ancestors of either concept, and then considering as common information (yielding l) the intersection of such sets, and as residuals (yielding n, m) the two symmetric differences. Again this is intuitive, since the number of common ancestors can be considered a good indicator of the shared features between the two concepts, just as the number of different ancestors can provide a reasonable estimation of the different information and features they own.

Dealing with natural language words, instead of explicit concepts, due to the problem of polysemy (a word may correspond to many concepts), their similarity must somehow combine the similarities between each pair of concepts underlying the words. Such a combination can consist, for instance, in the average or maximum similarity among such pairs, or can exploit the domain of discourse. A distance between groups of words (if necessary) can be obtained by couplewise working on the closest (i.e., taxonomically most similar) words in each group. Note that, in case of synonymy or polysemy, assuming consistency of domain among the

[1] According to the definition in [2], this yields a similarity measure rather than a full semantic relatedness measure, but we are currently working to extend it by taking into account other relationships as well.

words used in a same context [9], the similarity measure, by couplewise comparing all concepts underlying two words, can also suggest a ranking of which are the most probable senses for each, this way serving as a simple Word Sense Disambiguation [7] procedure, or as a support to a more elaborate one.

Since the taxonomic predicates represent further information about the objects involved in a description, in addition to their properties and roles, term similarity is the component where the corresponding similarity can be introduced in the overall framework. Of course, we assume that there is some way to distinguish taxonomic predicates from ordinary ones, so that they can be handled separately by the procedures. The similarity between two terms becomes:

$$\mathrm{sf}_o(t', t'') = \mathrm{sf}(n_c, l_c, m_c) + \mathrm{sf}(n_r, l_r, m_r) + \mathrm{sf}(n_t, l_t, m_t)$$

where the additional component refers to the number of common and different ancestors of the two concepts associated to the two terms, as specified above. In case the taxonomic information is expressed in the form of words instead of concepts, according to the one-domain-per-discourse assumption, these values can be referred to the closest pair of concepts associated to those words.

5 Application to the Sample Problems

Let us now go back to the sample problems presented in Section 2, and show how they can be tackled by properly setting and exploiting the general similarity framework proposed above. As to the list of concepts/words, Table 1 reports the similarity values corresponding to some more interesting couples. At the level of concepts, the similarity ranking is quite intuitive, in that less related concepts receive a lower value. The closest pairs are 'wild cat'-'tiger' and 'pet cat'-'tiger', followed by 'mouse animal'-'pet cat', then by 'mouse device'-'computer device', by 'pet cat'-'dog pet' and by 'dog pet'-'horse animal', all with similarity values above 0.5. Conversely, all odd pairs, mixing animals and devices or objects (including polysemic words), get very low values, below 0.4.

Table 1. Sample similarity values between WordNet words/concepts

Concept	**Concept**	**Similarity**
mouse (animal) [102330245]	computer (device) [103082979]	0.394
mouse (device) [103793489]	computer (device) [103082979]	0.727
mouse (device) [103793489]	cat (pet) [102121620]	0.384
mouse (animal) [102330245]	cat (pet) [102121620]	0.775
cat (domestic) [102121620]	computer (device) [103082979]	0.384
cat (pet) [102121620]	tiger (animal) [102129604]	0.849
cat (wild) [102127808]	tiger (animal) [102129604]	0.910
cat (pet) [102121620]	dog (pet) [102084071]	0.627
dog (pet) [102084071]	horse (domestic) [102374451]	0.542
horse (domestic) [102374451]	horse (chess) [103624767]	0.339
mouse (animal) [102330245]	mouse (device) [103793489]	0.394

As to the natural language sentences, the similarity between words is:

boy-girl = 0.75	boy-hammer = 0.436	girl-hammer = 0.436
want-desire = 0.826	want-hit = 0.361	desire-hit = 0.375
yellow-small = 0.563	small-small = 1	
dog-canary = 0.668	dog-nail = 0.75	canary-nail = 0.387

which allows to overcome the problem of wrong similarity assessment according to syntactic comparisons only[2]. Indeed, the first two sentences neatly get the largest similarity value with respect to the other combinations:

$$\text{fs(s1,s2)} = 1.770 \qquad \text{fs(s1,s3)} = 1.739 \qquad \text{fs(s2,s3)} = 1.683$$

This specific application can be compared to other works that combine in various shapes and for different purposes structural descriptions of sentences, some kind of similarity and WN. [16] concerns Question Answering: sentences are translated into first-order descriptions by directly mapping them on the taxonomy concepts and relations, rather than describing their syntactic structure; the similarity algorithm is original but based on the classical Dice's coheffi-cient and on a proprietary, domain-specific ontology, while WN is exploited to disambiguate word meanings with the user's intervention. Other works concern Textual Entailment. [8] uses WN but not the hyperonimy relation as in our case, focussing on relations that are considered more meaningful for entailment. [4] exploits WN's hyperonymy relation only for finding a direct implication between terms (and similarity and glosses for the rest). [1] does not consider the grammatical structure in word overlap, and exploits classical WN similarities based on synsets (they use the most common sense for each word, while we choose the maximum similarity among all possible couples of senses). [14] exploits exact structural correspondences between the two sentences (while our framework can suggest proper associations even for indeterminate structural parts), applies taxonomic similarities among all possible pairs of words and then chooses the best ones (while we selectively compute similarity between structurally corresponding words only), and uses other WN relations than hyperonimy.

6 Conclusions

Horn clause Logic is a powerful representation language for automated learning and reasoning in domains where relations among objects must be expressed to fully capture the relevant information. While the predicates in the description language are defined by the knowledge engineer and handled only syntactically by the interpreters, they sometimes express information that can be properly

[2] Note that all similarities agree with intuition, except the pair dog-nail that gets a higher similarity value than dog-canary, due to the interpretations of 'dog' as 'a hinged catch that fits into a notch of a ratchet to move a wheel forward or prevent it from moving backward' and 'nail' as 'a thin pointed piece of metal that is hammered into materials as a fastener'. Nevertheless, the overall correct similarity ranking between sentences is not affected.

exploited only with reference to a taxonomic background knowledge in order to capture unexpressed and underlying relationships among the concepts described. This paper presented a general strategy for evaluating the similarity between Horn clauses in which taxonomic descriptors are used in support of normal ones, and provided toy experiments to show its effectiveness.

Future work will concern fine-tuning of the taxonomic similarity computation methodology by exploiting other taxonomic relationships, and its application to other problems, such as Word Sense Disambiguation in phrase structure analysis and building refinement operators for incremental ILP systems.

References

[1] Agichtein, E., Askew, W., Liu, Y.: Combining lexical, syntactic, and semantic evidence for textual entailment classification. In: Proc. 1st Text Analysis Conference (TAC) (2008)
[2] Budanitsky, A., Hirst, G.: Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In: Proc. Workshop on WordNet and Other Lexical Resources, 2nd meeting of the North American Chapter of the Association for Computational Linguistics, Pittsburgh (2001)
[3] Ceri, S., Gottlöb, G., Tanca, L.: Logic Programming and Databases. Springer, Heidelberg (1990)
[4] Clark, P., Harrison, P.: Recognizing textual entailment with logical inference. In: Proc. 1st Text Analysis Conference (TAC) (2008)
[5] Esposito, F., Fanizzi, N., Ferilli, S., Semeraro, G.: A generalization model based on oi-implication for ideal theory refinement. Fundamenta Informaticæ 47(1-2), 15–33 (2001)
[6] Ferilli, S., Basile, T.M.A., Biba, M., Di Mauro, N., Esposito, F.: A general similarity framework for horn clause logic. Fundamenta Informaticæ 90(1-2), 43–46 (2009)
[7] Ide, N., Vronis, J.: Word sense disambiguation: The state of the art. Computational Linguistics 24, 1–40 (1998)
[8] Inkpen, D., Kipp, D., Nastase, V.: Machine learning experiments for textual entailment. In: Proc. 2nd PASCAL Recognising Textual Entailment Challenge (RTE-2) (2006)
[9] Krovetz, R.: More than one sense per discourse. In: NEC Princeton NJ Labs., Research Memorandum (1998)
[10] Lin, D.: An information-theoretic definition of similarity. In: Proc. 15th International Conf. on Machine Learning, pp. 296–304. Morgan Kaufmann, San Francisco (1998)
[11] Lloyd, J.W.: Foundations of Logic Programming, 2nd edn. Springer, Berlin (1987)
[12] Miller, G.A.: Wordnet: A lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
[13] Muggleton, S.: Inductive logic programming. New Generation Computing 8(4), 295–318 (1991)
[14] Pennacchiotti, M., Zanzotto, F.M.: Learning shallow semantic rules for textual entailment. In: Proc. International Conference on Recent Advances in Natural Language Processing (RANLP 2007) (2007)
[15] Rouveirol, C.: Extensions of inversion of resolution applied to theory completion. In: Inductive Logic Programming, pp. 64–90. Academic Press, London (1992)
[16] Vargas-Vera, M., Motta, E.: An ontology-driven similarity algorithm. Tech Report kmi-04-16. Knowledge Media Institute (KMi), The Open University, UK