Learning genic interactions without expert domain knowledge: Comparison of different ILP algorithms

Luboš Popelínský POPEL@FI.MUNI.CZ
Jan Blaťák XBLATAK@FI.MUNI.CZ
Knowledge Discovery Lab, Faculty of Informatics, Masaryk University in Brno, Czech Republic

Abstract

A novel two-step method is proposed in which rules for processing simple examples (sentences that contain a single pair of terms from the dictionary) are learned separately. We also describe an extension of domain knowledge when no domain expert is available, We show that this two-step method performs better on the test set than other learning algorithms.

1. Introduction

In the information extraction task, domain knowledge given by an expert plays a significant role. Especially in mining genic interactions from text, a performance of the learning algorithm is going to increase when knowing e.g. impossible combinations of an agent and a target. In this paper we focus on the situation when no expert domain knowledge is available. We extended the domain knowledge of the LLL challenge task only with predicates for natural language processing – part-of-speech tags, hyperonyma, and the predicate for finding position of a verb between two terms from the domain dictionary. Additionally, we learn syntactic relations as first-order frequent patterns and use them as new predicates.

We propose two-step learning method in which two theories – one for simple examples, another for all examples – are learned separately. We compare results obtained with the domain knowledge without the patterns and with the patterns with performance of class association rules (Liu et al., 1998). For comparison, we also bring results of learning from the data after propositionalization.

Appearing in *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Bonn, Germany, 2005.

2. Domain knowledge

Each sentence has been first morphologically tagged with Brill tagger (Brill, 1992). WordNet (WN) has been employed for finding semantic information, actually hyperonyma of the words. A predicate `ffverb` has been added that returns, for a pair of terms from the dictionary, a verb that appears between them. We also removed the `lemma` predicate because it almost never appeared in the learned rules, and the `word` predicate. It resulted in speed up of learning without any decrease of accuracy.

3. Learning algorithms

3.1. Aleph

We employed Aleph[1] and for each example learned its generalization (`induce_max` command). We used default settings except `clauselength=5` (an upper bound on the clause length), `minpos=2` (a lower bound on the number of positive examples covered by the clause), `evalfn=entropy` (Clause utility is p log p + (1-p) log (1-p) where p = P/(P + N) and P, N are the number of positive and negative examples covered) and `nodes=100000` (an upper bound on the nodes to be explored when searching for an acceptable clause).

We learned two kinds of rules, ones that for a given sentence return a pair of agent-target – called here *positive rules* – and also *disambiguation rules* that from all possible pairs in the given sentence aim at removing such pairs that are incorrect.

3.2. RAP

RAP (Blatak et al., 2003; Blatak et al., 2004) was used for finding new predicates as frequent Datalog queries. RAP generates frequent patterns by heuristic or random search what results in a faster acquisition of long patterns than the breadth first search.

[1] `http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/`

In this work, RAP learned frequent syntactic patterns. Only information contained in the `relation` predicate was exploited, together with `ffverb` and `follows(Word1,Word2)`[2]. The minimal support was 10% and patterns up to the length of 15 literals were generated with best-first search (entropy based heuristics that prefer emerging patterns[3]). A candidate pattern longer than 4 literals that was θ-subsumed by some of the frequent patterns found was removed.

Similarly as in the case of Aleph, both positive and negative rules were learned.

3.3. Propositionalization

All the frequent patterns generated with RAP – together 536 patterns – were transformed into boolean features. For processing this data we used J4.8 decison tree learner, Naive Bayes classifier and instance-based learner IB1 from Weka (Witten, Frank, 1999).

4. Finding genic interactions

The method combines positive and disambiguation rules by the following way. Given four parameters, POSRULES, MINPOS, DISRULES and MINNEG, a pair of two terms, A1 and A2, from the dictionary is a valid genic interaction pair (Agent,Target), if

1. at least POSRULES rules have fired, or
2. a singe rule has fired that covered at least MINPOS positive examples from the learning set, and
3. there is no (A2,A1) after application of all the positive rules.

If there are still unresolved pairs of terms, apply disambiguation rules. For all possible pairs of terms in the sentence remove a pair (A1,A2) if

1. at least DISRULES rules have fired, or
2. a singe rule has fired that covered at least MINNEG negative examples from the learning set.

To summarize, positive rules are applied to test examples first. Consequently, disambiguation rules are employed to remove the remaining ambiguities.

[2] `follows(S,X,Y)` succeeds if the word `X` appeared later in the sentence `S` than the word `Y`.

[3] A pattern is emerging if the coverage on positive and on the negative examples differ significantly.

5. Summary of results

The best result was received for two-step method (see Table 1, AL1). Precision (PRE) as well as recall (REC) and F-measure (F-M) was always higher than 40%.

Table 1. Overview of results

		PRE	REC	F-M
AL2	Aleph, 2-step method	46.5	50.0	48.2
AFP	Aleph + freq.patterns	37.6	64.8	47.6
AL1	Aleph,no freq.patterns	42.5	42.5	42.5
CAR	class association rules	37.2	29.6	32.9
PRO	propositionalization	28.0	29.6	28.8

In the case of AL2, only positive rules were learned. We first selected the sentences that contained only one couple of agent-target, and learned additional rules from this learning set. Then these rules have been added to the rules learned from whole learning set.

6. Discussion of results

6.1. Aleph

Influence of different setting of POSRULES (DISRULES) – the lower limit for the number of positive(disambiguation) rules that cover the example, and MINPOS (MINNEG) – the lower limit for coverage of a positive (disambiguation) rule – for two-step learning is in Table 2. Table 3 displays results for single-step learning. It need to be stressed that for all four learning methods – AL2, AFP, AL1 and CAR – the application of disambiguation rules resulted in increase of F-measure. The last line of Table 2 also displays the result submitted to the challenge.

Table 2. Two-step learning: top 5 results

MINPOS	POSRUL	MINNEG	DISRUL	F-M
5	3	3	2	48.2
6	3	3	2	46.7
5	3	2	2	45.7
5	3	0	0	45.6
4	2	0	0	45.1

6.2. Frequent patterns

We also added to the domain knowledge the frequent patterns learned with RAP. Performance of rules learned with Aleph on the test set did not overcome the best result reached (AL2), however, it was higher

Table 3. Single-step learning: top 5 results

MINPOS	POSRUL	MINNEG	DISRUL	F-M
4	2	3	2	42.5
3	2	-	3	41.8
3	2	3	2	41.5
3	2	2	1	40.0
3	2	0	0	38.0

than All if the disambiguation rules were also used. The appearance of the patterns in the rules learned with Aleph was about 10%. One of the interesting patterns that covers 14 positive and 58 negative examples is: *If there are two words, A and B, and there is a word W in* `comp` *relation with B, and B is right to a verb, then A is not an agent for B.*

6.3. Weka

All the frequent patterns learned with RAP have been taken that had non-zero coverage. These boolean features were used for learning with several propositional learners. The results can be found in Table 4.

Table 4. Results with propositionalized data

	PRE	REC	F-M
SVM	28.0	29.6	28.8
Decision tree	35.4	20.3	25.8
Naive Bayes	22.5	16.6	19.1
IB1	16.4	22.2	18.8

6.4. Domain knowledge

We also checked whether the new predicates – part-of-speech tags (`tag`), hyperonyma (`hyper`), `ffverb` – really help to improve the result. When the part-of-speech tags set by Brill tagger were removed, the F-measure decrease in about 10. Similar situation appears for `ffverb`. The withdrawal of `hyper` has smaller effect. To show how useful the new predicates are, we also checked appearance of these predicates in all the learned theories. `tag` and `ffverb` appeared in each third rule (32.4% and 34.3% respectively). `hyper` appeared in average in 15.6% of rules.

6.5. Maximizing precision

For the single-step learning with Aleph we also explored the possibility of increasing precision and preserving the number of the found pairs high. For the minimum number of correctly recognized agent-target pairs (COR) higher than 15, the highest precision reached 62.9%. The use of disambiguation rules resulted in additional increase of precision but with rapid decrease of correctly recognized agent-target pairs. The best results (PRE $\geq$ 60%) for positive rules are in Table 5.

Table 5. Maximizing precision

MINPOS	POSRUL	COR	PRE
6	5	17	62.9
6	4	17	60.7
7	4	17	60.7
7	3	18	60.0

7. Concluding remarks

We showed that two-step learning method outperformed other ILP approaches. However, neither precision nor recall are high enough for automatic extraction of genic interactions from medical text. Additional increase of precision cannot be reached without employing domain-oriented knowledge, e.g. Gene Ontology (`http://www.geneontology.org/`).

Acknowledgments

We thank to Peter Krutý for his help in the initial phase of this work. Authors have been partially supported by the internal grant of FI MU in Brno.

References

WordNet, a lexical database for the English language. `http://www.cogsci.princeton.edu/~wn/`

J. Blaťák, L. Popelínský, and M. Nepil. Feature construction with RAP. In *ILP'03 Works in Progress*, pages 1–10, 2003.

J. Blaťák, L. Popelínský Mining first-order maximal frequent patterns. Neural Network World 5, 4, pp. 381-390.

Brill, E., A simple rule-based part of speech tagger. Third Conference on Applied Natural Language Processing, Trento 1992.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. Knowledge Discovery and Data Mining, pages 80–86, 1998.

Witten, I. H., Frank, E. Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufman, 1999