USING VERY LARGE CORPORA TO DETECT RAISING AND CONTROL VERBS

Grzegorz Chrupała and Josef van Genabith
National Center for Language Technology
Dublin City University

Proceedings of the LFG07 Conference

Miriam Butt and Tracy Holloway King (Editors)

2007

CSLI Publications

http://csli-publications.stanford.edu/

Abstract

The distinction between raising and subject-control verbs, although crucial for the construction of semantics, is not easy to make given access to only the local syntactic configuration of the sentence. In most contexts raising verbs and control verbs display identical superficial syntactic structure. Linguists apply grammaticality tests to distinguish these verb classes. Our idea is to learn to predict the raising-control distinction by simulating such grammaticality judgments by means of pattern searches. Experiments with regression tree models show that using pattern counts from large unannotated corpora can be used to assess how likely a verb form is to appear in raising vs. control constructions. For this task it is beneficial to use the much larger but also noisier Web corpus rather than the smaller and cleaner Gigaword corpus. A similar methodology can be useful for detecting other lexical semantic distinctions: it could be used whenever a test employed to make linguistically interesting distinctions can be reduced to a pattern search in an unannotated corpus.

1 Introduction

In this paper we investigate to what degree very large unannotated corpora can be useful in acquiring detailed specifications of verbal subcategorization: specifically we attempt the task of detecting *raising* and *subject control* verbs.

The task of data-driven lexical acquisition is interesting from at least two points of view. First it can shed light on the process of lexical learning from linguistic input in humans. Second, it is relevant for Natural Language Engineering, where detailed information on subcategorization requirements of lexical items is useful for parsing.

Distinguishing between raising and control verbs is a small but interesting and seldom investigated aspect of automatically acquiring verbal lexical resources. In this paper we propose to make a somewhat non-standard use of large unannotated corpora to aid lexical acquisition. We extract features associated with raising and control verbs in a large unannotated corpus, learn a model which distinguishes the two classes using a small annotated (gold) corpus, and then verify how well our model predicts the two classes in a held-out portion of the gold corpus.

The errors our model makes may be partly be due to the limitations of the method we use, i.e. the features we extract or the learning mechanism we employ. More interestingly, they may also reveal mistakes or omissions in the small gold manually constructed resource when contrasted with usages in large amounts of naturally occurring data. In Section 6 we discuss those issues in more detail.

The structure of the paper is as follows: In Section 2 we briefly describe the raising-control distinction and its treatment in LFG. In Section 3 we briefly discuss previous work. In Section 4 we describe the methodology and resources used, while in Section 5 we present the experimental evaluation. Finally in Section 6 we discuss the implications of our results and present our conclusions.

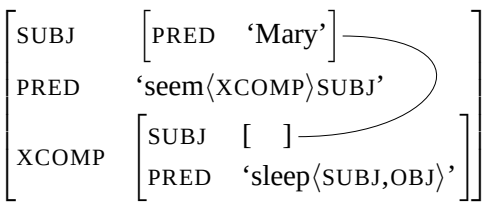

Figure 1: F-structure for *Mary seems to sleep* (raising - functional control)

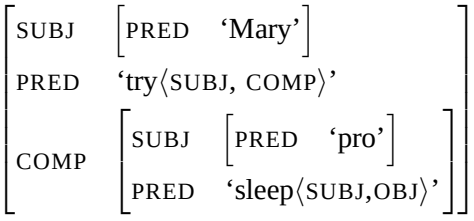

Figure 2: F-structure for *Mary tries to sleep* (anaphoric control)

2 Raising and control verbs

In English *raising* verbs are verbs such a *seem*. They require a syntactic subject which does not correspond to a semantic argument.

Subject control verbs are matrix verbs such as *try* one of whose arguments is shared with the the subordinate verb's SUBJ. In Dalrymple (2001) they receive a treatment in terms of obligatory anaphoric control, where the COMP's SUBJ's PRED value is bound to the matrix verb's SUBJ (see Fig. 2).

In Bresnan (2001) subject control verbs are treated in terms of functional control similar to raising verbs (see Fig. 3). In this type of analysis the only thing distinguishing raising constructions from control constructions is the subcat frame (semantic form): the fact that the subject argument is not a semantic argument of the raising verb is indicated notationally by putting it outside the angle brackets: 'seem⟨XCOMP⟩SUBJ'.

Whichever analysis one adopts, the distinction between raising and control verbs is important as it affects meaning: the predicate encoded by *seems* is unary whereas the one encoded by *try* is binary. Thus it is crucial when constructing the semantic argument structure for a verb with a non-finite complement.

There are a number of constructions which distinguish between those two verb classes:

(1) a. It seemed to rain.

b. There seems to be a problem.

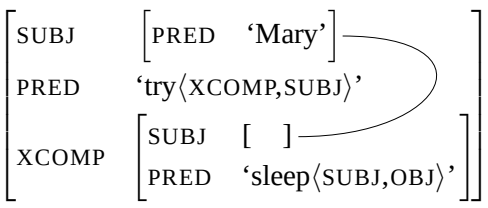

Figure 3: F-structure for *Mary tries to sleep* (functional control)

c. Did she leave? *She seemed.

(2) a. * It tried to rain.

b. * There tried to be a problem.

c. Did she leave? She tried.

English raising verbs appear with dummy subjects as in examples (1a) and (1b). They do not admit VP drop (1c). Control verbs exhibit the opposite behavior as shown in (2).

3 Previous work

In most contexts, raising verbs and control verbs display identical superficial syntactic structure. Many resources meant to provide training and evaluation material for data-driven computational methods do not encode the raising-control distinction in any way; examples include the Penn Treebank (Marcus et al., 1994), or the PARC 700 Dependency Bank (King et al., 2003). O'Donovan et al. (2005) implement a large scale system for acquiring LFG semantic forms using the Penn Treebank but do not differentiate between frames for raising and control verbs.

Briscoe and Carroll (1997) mention in passing that the fact that argument slots of different subcategorization frames for the same verb share the same semantic restrictions could be used to learn about alternations the verb participates in and thus make inferences about raising and control facts. However to our knowledge neither they nor other researchers have followed on these ideas and there have been no studies specifically focusing on acquiring the raising/control distinction.

In the following sections we investigate whether frequency counts from very large corpora can be used to reliably distinguish those two verb classes.

4 Methods

The raising-control distinction is not easy to make given access to only the local syntactic configuration of the sentence. However, speakers have little difficulty

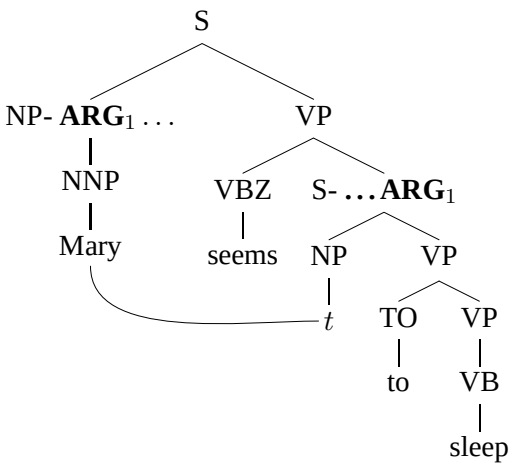

Figure 4: Propbank-style annotation for the raising construction with *seem*

in applying grammaticality tests such as those in example (1) to distinguish these verb classes. Our idea is to simulate making those grammaticality judgements. We hypothesize that the absence of evidence approximates evidence of absence: a simple construction, if it is grammatical, is bound to show up in a sufficiently large amount of naturally occurring language data. So a grammaticality test reduces to a pattern search in a corpus.

There are two complicating factors:

- the need for a very large corpus to minimize the chance that the absence of matches is accidental rather than systematic
- the inevitable presence of noise in the form of false positive matches, for example caused by misspellings, interlinguistic interference or automatically generated pseudo-language.

These two factors have to be traded off against each other: a corpus with carefully selected text samples is likely to be mostly free of noise but will probably be too small to avoid false negatives. Conversely, a terabyte-scale corpus will almost inevitably contain some proportion of false positives due to noise.

We use two types of corpora in our study. First we use a relatively small corpus annotated with syntactic structure and semantic roles, namely the English Propbank (Palmer et al., 2005). This contains the same text as the English Penn Treebank. Each verb form is annotated with the labeled semantic arguments it governs. The semantic roles are to a large extent verb-specific and are numbered as ARG_0 through ARG_5. In general ARG_0 can be said to correspond to a prototypical Agent (Dowty, 1991) and ARG_1 is the prototypical Patient. The higher-numbered roles are completely verb specific and no generalizations can be made about them.

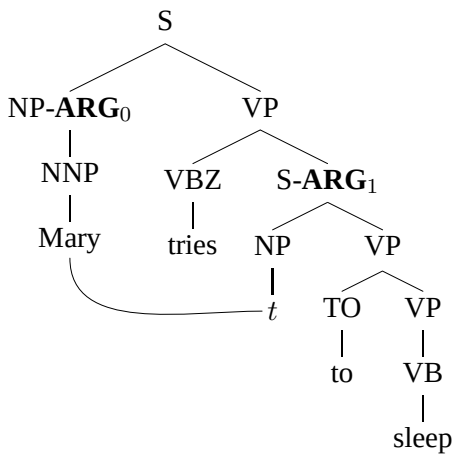

Figure 5: Propbank-style annotation for the control construction with *try*

Thanks to the information about semantic roles which Propbank annotations add to Penn treebank trees, it is possible to distinguish raising and control constructions. In Figures 4 and 5 we present the analyses that example raising and control verbs receive in Propbank. In the case of the raising construction with *seem* there is a single (discontinuous) semantic argument ARG_1. In contrast, in a control construction the verb *try* has two arguments ARG_0 and ARG_1.

We use the English Propbank to extract verb forms which appear at least 3 times in contructions with non-finite complements.[1]. For each verb form we also extract the form of the complement (to-infinitive or gerund). To each verb form v we assign the maximum-likelihood estimate of its *raising probability* $P_R(v)$, i.e. the proportion of times it appears in raising constructions. We take the presence of the ARG_0 semantic argument to indicate a subject control construction and its lack to indicate a raising construction. The resulting list of 120 verbs forms is randomly divided into a training set and test set of equal sizes.

The second type of resource we use is a large-scale unannotated corpus of English text. We experiment with two such corpora Gigaword (Graff, 2003) (1.7 billion words of newswire) and the English web pages indexed by Yahoo!.

Those large corpora are used to extract frequencies of occurrence of the verb forms in context that are indicative of the degree to which they can appear in raising contructions (i.e. $P_R(v)$). From those frequency counts we derive features used to train regression models that will predict $P_R(v)$ for each verb form.

There are a number of choices as to how to extract the most informative occurrence frequency counts. In this study we decided to try to mimic grammaticality

[1]The extraction is not 100% reliable, due to annotation errors in the Penn Treebank. For example in several cases the participle use of *said* as in *X is said to Y* is mistagged as past tense, which is why *said* appears among our 120 verb forms.

tests used by linguists in distinguishing between raising and control constructions. The assumption which enables us to approximate grammaticality judgements by corpus searches is that any simple grammatical construction is very likely to occur in a sufficiently large corpus. There are some important qualifications that need to be made about its validity. The construction in question should be as simple as possible and ideally contain high frequency lexical items. The semantics associated with it should be plausible. The search pattern itself should be possible to run on un-annotated data and still be resistant to noise.

Those are quite strict prerequisites and it can be hard to build search patterns that satisfy all of them. For example it is challenging to come up with a template based on the grammaticality test in (1a) and (2a) which will not suffer from some shortcomings: *it X to rain* depends on the lexical item *rain* which is not high frequency enough for most corpus sizes. Even in combination with the most common raising verb, *it seemed to rain* only occurs in two unique sentences in Gigaword. For the test in (1c) and (2c), with access just to un-annotated data it would very hard to detect those sentence-final strings such as "seemed" which are VP-drop. An additional complication is that Web search indexes such as Yahoo! do not typically include punctuation which makes it impossible to detect sentence boundaries. Thus in the experiments described below we use the search patterns based on the test b vs b, which we deemed the most robust.

For each verb form V tested, we build patterns using the following templates:

(3) a. there V to be
    b. there V being

(4) a. V to be
    b. V being

Version (a) or (b) is chosen depending on the complement type the verb takes. String (3) is our test pattern which is meant to check whether verb form X is grammatical in raising constructions. String (4) is the background frequency of verb form V with a non-finite complement. The ratio of (3) to (4) gives us the maximum likelihood estimate of the probability of dummy-*there* in nonfinite complement contexts.

Gigaword contains articles or portions of articles that are repeated: to correct for inflated counts caused by this we remove duplicate lines from the corpus in a preprocessing step. We match patterns by ignoring upper/lower case.

In the case of the Web we use the Yahoo! search API – we restrict the search to English-language pages, thus relying on Yahoo!'s language-detection method, and use the *total result available* number as our frequency count, thus trusting the estimate Yahoo! provides. All the web frequency counts were collected on a single day (July 1 2007) and stored to ensure consistency between experiments.

5 Experiments

We performed experiments with two corpora: Gigaword and the Web. We search for occurrences of the pattern strings (3) and (4) and for each verb form we gather the following scores:

- $C_1(v) =$ frequency of pattern (3)
- $C_2(v) =$ frequency of pattern (4)
- $C_1(v)/C_2(v)$

5.1 Models

We experiment with two baselines and a regression tree model to learn to predict $P_R(v)$ from training examples. As a metric for evaluating the quality of the models, both during cross-validating and for final evaluation, we use the Mean Squared Error (MSE). For the list of gold scores $\mathbf{v}$ and the list of predicted scores $\hat{\mathbf{v}}$ for n verb forms, this metric is defined as follows:

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(\mathbf{v}_i - \hat{\mathbf{v}}_i)^2 \tag{5}$$

Mean This is a very simple baseline: for each verb we form predict $P_R(v)$ to be the mean $\overline{P}_R$ in the training set.

Linear regression This baseline is the linear regression model fitted to training data using $C_1(v)/C_2(v)$ as the sole explanatory variable. The model for Gigaword data is $P_R = 13.2936 \times C_1(v)/C_2(v) + 0.2741$, while the Web model has the form $P_R = 11.5011 \times C_1(v)/C_2(v) + 0.2547$.

Regression tree This is the model obtained by inducing a regression tree. A regression tree is simply a type of decision tree where the response at each leaf is a real number. The tree is built using the recursive partitioning method of Breiman et al. (1984), as implemented in the *rpart* R package (Therneau et al., 2007; Therneau and Atkinson, 2000).

We chose this model because of its relative simplicity and transparency. At this stage our main goal was to gain insight from our data rather than simply maximize performance.

The algorithm starts by grouping all training examples in a single node. At each step a split (i.e. a value of one of the features) is chosen to partition the training examples at the current node T in such a way as to maximize the splitting criterion:

$$SS_T - (SS_L + SS_R) \tag{6}$$

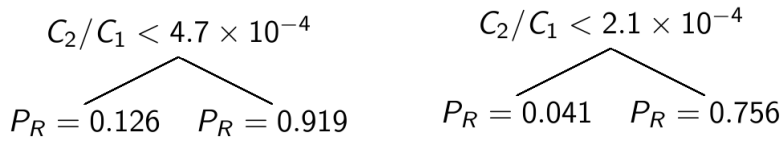

Figure 6: The regression tree model: left for Gigaword data, right for Web data

Model	Gigaword MSE	Yahoo Web MSE
Mean	0.194	0.194
Linear regression	0.165	0.164
Regression tree	**0.134**	**0.110**

Table 1: Evaluation results on the test set

SS_T is the within node sum of squares for the current node T, where y_i is the output value for the i^{th} training example at node T and $\overline{y}$ is the mean of the outputs of examples at node T:

$$SS_T = \sum_i (y_i - \overline{y})^2 \tag{7}$$

SS_L and SS_R are sums of squares for the left and right child given by the split under consideration.

The same step is applied recursively to both children nodes until the maximum number of splits is reached or no further splits are possible. For each node the predicted response is the mean of the instances in this node. The tree constructed in this fashion is then pruned using leave-one-out cross-validation in order to find the tree which minimizes Mean Squared Error.

In our experiments we start with all three features but the resulting pruned trees only use the ratio feature $C_1(v)/C_2(v)$: trees with more depth increase cross-validated error. Figure 6 shows the regression trees for both experiments. For the Gigaword tree the top node is split at $C_1(v)/C_2(v) < 4.7 \times 10^{-4}$ and for the Web tree at $C_1(v)/C_2(v) \geq 2.1 \times 10^{-4}$.

5.2 Results

In Table 1 we report the Mean Squared Error score on the test set for counts extracted from the Gigaword and the Yahoo Web achieved by the models.

Our results show that for **regression tree** the Web counts give models with lower error on test data in comparison to the Gigaword-based model.

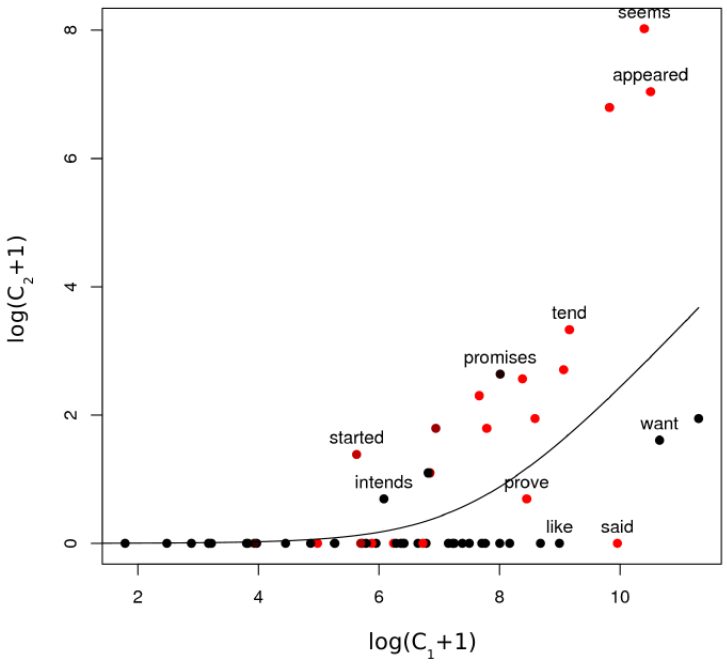

Figure 7: Results for Gigaword regression tree

Since both regression trees are of depth 2, in effect both trees partition verb forms into two classes: predominantly raising verbs and predominantly control verbs. Figures 7 and 8 illustrate how well that partition separates verb forms in the test data. Both figures plot C_2 against C_1 on a logarithmic scale. Each dot represents a verb form; the varying color indicates the following: black stands for gold $P_R(v)$ = 0 and red for $P_R(v)$ = 1, with intermediate colors encoding values between 0 and 1. The black curve on each plot separates points in the same fashion as the top node in the regression tree model, i.e. $C_2(v) = 4.7 \times 10^4 \times C_1(v)$ for the Gigaword tree and $C_2(v) = 2.1 \times 10^4 \times C_1(v)$ for the Web tree.

The complete results obtained by the regression tree models trained with the Gigaword and Web counts for the verb forms in the Propbank-derived test set are included in Tables 2 and 3. Column three shows the values of $P_R(v)$ estimated from Propbank; the following two columns show the predictions of the Gigaword model, the squared errors for that prediction, and analogous numbers for the Web model in the last two columns.

Among the 60 verb forms in the test set, the Gigaword regression tree has squared errors larger than 0.25 for 10 verb forms. The corresponding Web model has squared errors above 0.25 for 8 verb forms.

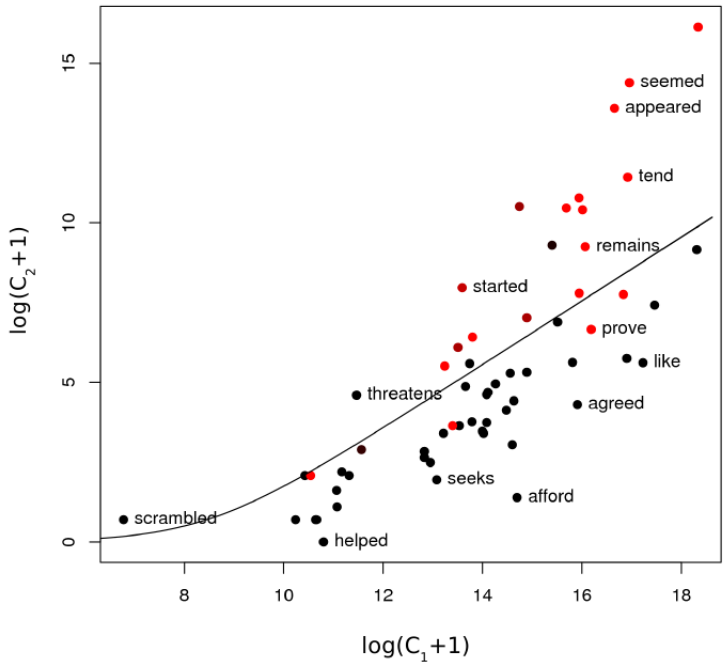

Figure 8: Results for Web regression tree

In some cases where the models disagree with the Propbank-derived gold standard they are not necessarily wrong. For example both the regression tree models give a high $P_R(promised)$ based on occurrences of strings such as *At $300 apiece there promised to be a tremendous profit in the thing* which seem genuine raising usages. However, all the uses of *promised to* in Propbank are classified as control, which results in a gold $P_R(promised) = 0$.

In our experiments we did not group all the inflected forms of each verb together – rather we treat each verb-form as a separate example. This means that we have more training and test examples; but also that there are fewer frequency counts for each individual example. Grouping the verb forms together might change our numbers somewhat but we do not expect this effect to be large.

6 Discussion

The experiments show that using pattern counts from large corpora can be used to assess how likely a verb form is to appear in raising vs. control constructions. We evaluated two simple models and showed that they perform much better than the baseline.

Table 2: Regression tree results on test set - part 1

Form	Complement	Gold P_R	Giga	Giga SE	Web	Web SE
afford	TO	0	0.126	0.0159	0.041	0.0017
agreed	TO	0	0.126	0.0159	0.041	0.0017
aims	TO	0	0.126	0.0159	0.041	0.0017
appeared	TO	1	0.919	0.0065	0.756	0.0594
attempt	TO	0	0.126	0.0159	0.041	0.0017
began	TO	0.609	0.919	0.0966	0.756	0.0218
begin	TO	1	0.126	0.7634	0.756	0.0594
came	TO	1	0.919	0.0065	0.756	0.0594
chose	TO	0	0.126	0.0159	0.041	0.0017
decide	TO	0	0.126	0.0159	0.041	0.0017
decline	TO	0	0.126	0.0159	0.041	0.0017
declined	TO	0	0.126	0.0159	0.041	0.0017
declines	TO	0	0.126	0.0159	0.041	0.0017
expected	TO	0	0.126	0.0159	0.041	0.0017
failed	TO	1	0.126	0.7634	0.756	0.0594
get	TO	0.667	0.919	0.0639	0.756	0.0080
happen	TO	1	0.919	0.0065	0.756	0.0594
helped	TO	0	0.126	0.0159	0.041	0.0017
hesitate	TO	0	0.126	0.0159	0.041	0.0017
hope	TO	0	0.126	0.0159	0.041	0.0017
hoped	TO	0	0.126	0.0159	0.041	0.0017
include	VBG	1	0.126	0.7634	0.041	0.9193
intend	TO	0	0.126	0.0159	0.041	0.0017
intended	TO	0	0.126	0.0159	0.041	0.0017
intends	TO	0	0.919	0.8454	0.041	0.0017
like	TO	0	0.126	0.0159	0.041	0.0017
likes	TO	0	0.126	0.0159	0.041	0.0017
moved	TO	0.2	0.126	0.0054	0.041	0.0252
offer	TO	0	0.126	0.0159	0.041	0.0017

Table 3: Regression tree results on test set - part 2

Form	Complement	Gold P_R	Giga	Giga SE	Web	Web SE
plan	TO	0	0.126	0.0159	0.041	0.0017
planned	TO	0	0.126	0.0159	0.041	0.0017
prefer	TO	0	0.126	0.0159	0.041	0.0017
prepared	TO	0	0.126	0.0159	0.041	0.0017
promised	TO	0	0.919	0.8454	0.756	0.5718
promises	TO	0.111	0.919	0.6534	0.756	0.4161
proposed	TO	0	0.126	0.0159	0.041	0.0017
prove	TO	1	0.126	0.7634	0.041	0.9193
refuse	TO	0	0.126	0.0159	0.041	0.0017
remains	TO	1	0.919	0.0065	0.756	0.0594
said	TO	1	0.126	0.7634	0.041	0.9193
scrambled	TO	0	0.126	0.0159	0.756	0.5718
seeks	TO	0	0.126	0.0159	0.041	0.0017
seemed	TO	1	0.919	0.0065	0.756	0.0594
seems	TO	1	0.919	0.0065	0.756	0.0594
serve	TO	0	0.126	0.0159	0.041	0.0017
served	TO	0	0.126	0.0159	0.041	0.0017
start	TO	0.667	0.126	0.2920	0.756	0.0080
started	TO	0.778	0.919	0.0201	0.756	0.0005
stood	TO	1	0.126	0.7634	0.041	0.9193
struggles	TO	0	0.126	0.0159	0.041	0.0017
tend	TO	1	0.919	0.0065	0.756	0.0594
threatens	TO	0	0.126	0.0159	0.756	0.5718
tries	TO	0	0.126	0.0159	0.041	0.0017
turn out	TO	1	0.919	0.0065	0.756	0.0594
turns out	TO	1	0.919	0.0065	0.756	0.0594
vote	TO	0	0.126	0.0159	0.041	0.0017
voted	TO	0	0.126	0.0159	0.041	0.0017
want	TO	0	0.126	0.0159	0.041	0.0017
wish	TO	0	0.126	0.0159	0.041	0.0017
worked	TO	0	0.126	0.0159	0.041	0.0017

It also seems that for this task it is beneficial to use the much larger but also noisier Web corpus rather than the relatively small and clean Gigaword. The method we used is to a certain extent robust to noise and benefits from the sheer quantity of data available on the web.

Similar methodology might be useful for detecting other lexical semantic distinctions: it could be used whenever a test employed to make linguistically interesting distinctions can be reduced to a pattern search in an unannotated corpus.

Acknowledgements

We gratefully acknowledge support from Science Foundation Ireland grant 04/IN/I527 for the research presented in this paper.

References

Breiman, Leo, Friedman, Jerome H., Olshen, R. A. and Charles J., Stone. 1984. *Classification and Regression Trees*. Belmont, CA: Wadsworth.

Bresnan, Joan. 2001. *Lexical-Functional Syntax*. Blackwell Publishing.

Briscoe, Ted and Carroll, John. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the fifth conference on Applied natural language processing*, pages 356–363.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*. Academic Press San Diego.

Dowty, David R. 1991. Thematic Proto-Roles and Argument Selection. *Language* 67(3), 547–619.

Graff, David. 2003. LDC English Gigaword Corpus.

King, Tracy Holloway, Crouch, Richard, Riezler, Stefan, Dalrymple, Mary and Kaplan, Ron. 2003. The PARC 700 Dependency Bank. *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)* pages 1–8.

Marcus, Mitchell P., Santorini, Beatrice and Marcinkiewicz, Mary Ann. 1994. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2), 313–330.

O'Donovan, R., Cahill, A., Way, A., Burke, M. and Van Genabith, J. 2005. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II and Penn-III Treebanks. *Computational Linguistics* 31(3), 329–365.

Palmer, Martha, Gildea, Daniel and Kingsbury, Paul. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics* 31(1), 71–106.

Therneau, Terry M., Atkinson, Beth and Ripley, Brian. 2007. *The rpart Package: Recursive Partitioning*.

Therneau, Terry M. and Atkinson, Elizabeth J. 2000. An introduction to recursive partitioning using the RPART routines. Technical Report.