LASSY: LARGE SCALE SYNTACTIC ANNOTATION OF WRITTEN DUTCH

Gertjan van Noord

Deliverable 6-3: Case Study 3

Self-Trained Bilexical Preferences to Improve Disambiguation Accuracy

Gertjan van Noord

1 Motivation

In parse selection, the task is to select the correct syntactic analysis of a given sentence from a set of parses generated by some other mechanism. On the basis of correctly labeled examples, supervised parse selection techniques can be employed to obtain reasonable accuracy. Although parsing has improved enormously over the last few years, even the most successful parsers make very silly, sometimes embarrassing, mistakes. In our experiments with a large wide-coverage stochastic attribute-value grammar of Dutch, we noted that the system sometimes is insensitive to the naturalness of the various lexical combinations it has to consider. Although parsers often employ lexical features which are in principle able to represent preferences with respect to word combinations, the size of the manually labeled training data will be too small to be able to learn the relevance of such features.

In maximum-entropy parsing, the supervised parsing technique that we use in our experiments, arbitrary features can be defined which are employed to characterize different parses. So it is possible to construct features for any property that is thought to be important for disambiguation. However, such features can be useful for disambiguation only in case the training set contains a sufficient number of occurrences of these features. This is problematic, in practice, for features that encode bilexical preferences such as selection restrictions, because typical training sets are much too small to estimate the relevance of features representing co-occurrences of two words. As a simple example consider the ambiguous Dutch sentence

(1) Melk drinkt de baby niet
Milk drinks the baby not
The baby doesn't drink milk / Milk doesn't drink the baby

The standard model of the parser we experimented with employs a wide variety of features including syntactic features and lexical features. In particular, the model also includes features which encode whether or not the subject or the object is fronted in a parse. Since subjects, in general, are fronted much more frequently than objects, the model has learned to prefer readings in which the fronted constituent is analyzed as the subject. Although the model also contains features to distinguish whether e.g. `milk` occurs as the subject or the object of `drink`, the model has not learned a preference for either of these features, since there were no sentences in the training data that involved both these two words.

To make this point more explicit, we found that in about 200 sentences of our parsed corpus of 27 million sentences `milk` is the head of the direct object of the verb `drink`. Suppose that we would need at least perhaps 5 to 10 sentences in our training corpus in order to be able to learn the specific preference between `milk` and `drink`. The implication

is that we would need a (manually labeled!) training corpus of approximately 1 million sentences (20 million words). In contrast, the disambiguation model of the Dutch parser we are reporting on in this paper is trained on a manually labeled corpus of slightly over 7,000 sentences (145,000 words). It appears that semi-supervised or un-supervised methods are required here.

Note that the problem not only occurs for artificial examples such as (1); here are a few mis-parsed examples actually encountered in a large parsed corpus:

(2) a. Campari moet **u** gedronken hebben
Campari must you drunk have
Campari must have drunk you / You must have drunk Campari

b. De wijn die **Elvis** zou hebben gedronken als hij wijn zou hebben
The wine which Elvis would have drunk if he wine would have
gedronken
drunk
The wine Elvis would have drunk if he had drunk wine /
The wine that would have drunk Elvis if he had drunk wine

c. De paus heeft **tweehonderd daklozen** te eten gehad
The pope has two-hundred homeless-people to eat had
The pope had two hundred homeless people for dinner

In this paper, we describe an alternative approach in which we employ point-wise mutual information association score in the maximum entropy disambiguation model. The association scores used here are estimated using a very large parsed corpus of 500 million words (27 million sentences), a preliminary version of the Lassy Large automatically annotated corpus of Dutch. We show that the incorporation of this additional knowledge source improves parsing accuracy.

Most of the report has been published as [24]. Practical details concerning the extraction of the relevant information from the Lassy Large treebank is documented in a newly added appendix. In addition, the appendix contains further examples of the results which were left out in the published version.

2 Previous research

Automatically learning selection restrictions from corpora using a parser goes back to [6, 7] They proposed the use of point-wise mutual information [8] to estimate the strength of association between verbs and head nouns of direct objects. Preprocessing of the corpus included the application of a robust parser (the Fidditch parser).

In [20], an alternative association metric is formulated which takes into account classes of arguments. For instance, verbs are associated with preferences for particular classes of head nouns as direct objects, rather than individual nouns. A number of variants of Resnik's metric are described in [21], and Ribas performs a number of experiments comparing these variants.

Clearly, the idea that selection restrictions ought to be useful for parsing accuracy is not new. However, as far as we know this is the first time that automatically acquired selection restrictions have been shown to improve parsing accuracy results for a wide-coverage full parsing task. For instance, [21] describes potential NLP tasks which could benefit from selectional restrictions, including syntactic ambiguity resolution. In his conclusions he mentions that '. . . the technique still seems far from practical application to NLP tasks. . . '.

Earlier work *has* shown that selection restrictions can be good predictors for certain types of attachment ambiguity. Based on an empirical study, [27] conclude that PP attachment decisions are predictable on the basis of lexical preferences of nouns, verbs and prepositions.

Furthermore, [9] describes a corpus-based technique to learn so-called *co-restrictions* automatically, and the paper illustrate that these could be useful for parsing by showing that for a particular attachment resolution task, the availability of co-restrictions improves over a right association baseline.

[1] and [12] describe how statistical information between verbs and case elements is collected on the basis of large automatically analyzed corpora. In a recent paper [13] they show that these case frames help disambiguate coordinations.

The association scores employed in this paper are estimated on the basis of a large corpus that is parsed by the parser that we aim to improve upon. Therefore, this technique can be described as a somewhat particular instance of self-training. Self-training has been investigated for statistical parsing before. Although naively adding self-labeled material to extend training data is normally not successful, there have been successful variants of self-learning for parsing as well. For instance, in [16] self-learning is used to improve a two-phase parser reranker, with very good results for the classical Wall Street Journal parsing task.

3 Background: Alpino parser

The experiments are performed using the Alpino parser for Dutch. In this section we briefly describe the parser, as well as the corpora that we have used in the experiments described later.

3.1 Grammar and Lexicon

The Alpino system is a linguistically motivated, wide-coverage grammar and parser for Dutch in the tradition of HPSG. It consists of over 700 grammar rules and a large lexicon of over 100,000 lexemes and various rules to recognize special constructs such as named entities, temporal expressions, etc. The grammar takes a 'constructional' approach, with rich lexical representations and a large number of detailed, construction specific rules. Both the lexicon and the rule component are organized in a multiple inheritance hierarchy. Heuristics have been implemented to deal with unknown words and word sequences, and ungrammatical or out-of-coverage sentences (which may nevertheless contain fragments

that are analyzable). The Alpino system includes a POS-tagger which greatly reduces lexical ambiguity, without an observable decrease in parsing accuracy [19].

3.2 Parser

Based on the categories assigned to words, and the set of grammar rules compiled from the HPSG grammar, a left-corner parser finds the set of all parses, and stores this set compactly in a packed parse forest. All parses are rooted by an instance of the top category, which is a category that generalizes over all maximal projections (S, NP, VP, ADVP, AP, PP and some others). If there is no parse covering the complete input, the parser finds all parses for each sub-string. In such cases, the robustness component will then select the best sequence of non-overlapping parses (i.e., maximal projections) from this set.

In order to select the best parse from the compact parse forest, a best-first search algorithm is applied. The algorithm consults a maximum entropy disambiguation model to judge the quality of (partial) parses. Since the disambiguation model includes inherently non-local features, efficient dynamic programming solutions are not directly applicable. Instead, a best-first beam-search algorithm is employed [25, 23].

3.3 Maximum Entropy disambiguation model

The maximum entropy model is a conditional model which assigns a probability to a parse t for a given sentence s. Furthermore, $f_i(t)$ are the feature functions which count the occurrence of each feature i in a parse t. Each feature i has an associated weight λ_i. The score ϕ of a parse t is defined as the sum of the weighted feature counts:

$$\phi(t) = \sum_i \lambda_i f_i(t)$$

If t is a parse of s, the conditional probability is given by the following, where $T(s)$ are all parses of s:

$$P(t|s) = \frac{\exp(\phi(t))}{\sum_{u \in T(s)} \exp(\phi(u))}$$

If we only want to select the best parse we can ignore the actual probability, and use the score ϕ to rank competing parses.

The maximum entropy model employs a large set of features. The standard model uses about 42,000 features. Features describe various properties of parses. For instance, the model includes features which signal the application of particular grammar rules, as well as local configurations of grammar rules. There are features signaling specific POS-tags and subcategorization frames. Other features signal local or non-local occurrences of extraction (WH-movement, relative clauses etc.), the grammatical role of the extracted element (subject vs. non-subject etc.), features to represent the distance of a relative clause and the noun it modifies, features describing the amount of parallelism between conjuncts in

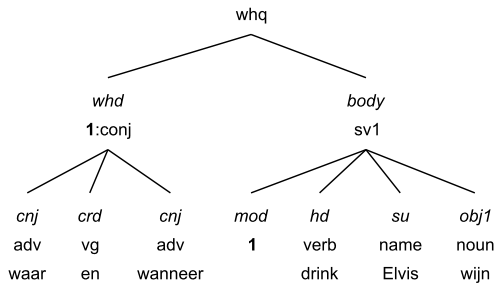

Figure 1: Dependency graph example.

a coordination, etc. In addition, there are lexical features which represent the co-occurrence of two specific words in a specific dependency, and the occurrence of a specific word as a specific dependent for a given POS-tag. Each parse is characterized by its feature vector (the counts for each of the 42,000 features). Once the model is trained, each feature is associated with its weight λ (a positive or negative number, typically close to 0). To find out which parse is the best parse according to the model, it suffices to multiply the frequency of each feature with its corresponding weight, and sum these weighted frequencies. The parse with the highest sum is the best parse. Formal details of the disambiguation model are presented in [25]. For training the maximum entropy models, we use an implementation by [14].

3.4 Dependency structures

Although Alpino is not a dependency grammar in the traditional sense, dependency structures are generated by the lexicon and grammar rules as the value of a dedicated feature `dt`. The dependency structures are based on CGN (Corpus Gesproken Nederlands, Corpus of Spoken Dutch) [10], D-Coi and LASSY [26]. Such dependency structures are somewhat idiosyncratic, as can be observed in the example in figure 1 for the sentence:

(3) waar en wanneer dronk Elvis wijn?
where and when drank Elvis wine?
Where and when did Elvis drink wine?

In such a CGN dependency structure, heads are represented as a daughter leaf node of an abstract non-terminal node. Different types of head receive a different relation label such as `hd` for ordinary heads and `whd` (for WH-phrases). Other types of heads include coordinators (`crd`), relative pronouns (`rhd`) and complementizers (`cmp`). Non-leaf nodes are decorated further with a category specification, and leaf-nodes similarly have a POS-tag.

As a further peculiarity, nodes can be linked to more than a single mother node. In such cases, dependency structures are really graphs. In CGN, the term *secondary edge* was used for such cases. As in attribute-value structures with reentrancies, such graphs are visualized by displaying trees where co-indexed nodes indicate sharing. In this case,

for example, the WH-phrase is both the *whd* element of the top-node, as well as a *mod* dependent of the verbal cluster headed by *drink*, as indicated by the index 1.

3.5 Named dependency relations

Often we do not work with the dependency structures themselves, but we extract named lexical dependencies from the dependency structure. The dependency graph in figure 1 is represented with the following set of dependencies:

crd/cnj(`en`, `waar`)	crd/cnj(`en`, `wanneer`)
whd/body(`en`, `drink`)	hd/mod(`drink`, `en`)
hd/obj1(`drink`, `wijn`)	hd/su(`drink`, `Elvis`)

For a given node in a dependency structure, a dependency exists between the root form associated with the head daughter (the daughter labeled with one of the designated labels indicating heads) and the root forms associated with each of the non-head daughters. The root form of a dependency structure for non-leaf nodes is the root form associated with the head daughter of that structure. A named lexical dependency is written as $r_1/r_2(w_1, w_2)$ where the head daughter has dependency label r_1, the non-head daughter has dependency label r_2, and the root forms associated with the head daughter and the non-head daughter are w_1 and w_2 respectively. Below, we often write $r(w_1, w_2)$ with the understanding that r is a pair such as `hd/obj1` or `whd/body`.

3.6 Evaluation

The output of the parser is evaluated by comparing the generated dependency structure for a corpus sentence to the gold standard dependency structure in a treebank. For this comparison, we represent the dependency structure as a set of named dependency relations, as illustrated in the previous paragraph.

Comparing these sets, we count the number of dependencies that are identical in the generated parse and the stored structure, which is expressed traditionally using precision, recall and f-score [5].

Let D_p^i be the number of dependencies produced by the parser for sentence i, D_g^i is the number of dependencies in the treebank parse, and D_o^i is the number of correct dependencies produced by the parser. If no superscript is used, we aggregate over all sentences of the test set, i.e.,:

$$D_p = \sum_i D_p^i \qquad D_o = \sum_i D_o^i \qquad D_g = \sum_i D_g^i$$

We define precision as the total number of correct dependencies returned by the parser, divided by the overall number of dependencies returned by the parser; recall is the number of correct system dependencies divided by the total number of dependencies in the treebank:

$$\text{precision} = \frac{D_o}{D_p} \qquad \text{recall} = \frac{D_o}{D_g}$$

As usual, precision and recall are combined in a single f-score metric:

$$\text{f-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

An alternative similarity score for dependency structures is based on the observation that for a given sentence of n words, a parser would be expected to return (about) n dependencies. In such cases, we can simply use the percentage of correct dependencies as a measure of accuracy. To allow for some discrepancies between the number of expected and returned dependencies, we divide by the maximum of both. This leads to the following definition of *concept accuracy*. A similar definition can be found, for instance, in [2]. The number of returned dependencies can be greater than the number of expected dependencies, in cases where the gold parse includes fewer secondary edges than the proposed parse.

$$\text{CA} = \frac{D_o}{\sum_i \max(D_g^i, D_p^i)}$$

The concept accuracy metric can be characterized as the mean of a per-sentence minimum of recall and precision. The resulting CA score therefore is typically slightly lower than the corresponding f-score.

The standard version of Alpino that we use here as baseline system is trained on the 145,000 word Alpino treebank, which contains dependency structures for the `cdbl` (newspaper) part of the Eindhoven corpus. The parameters for training the model are the same for the baseline model, as well as the model that includes the self-trained bilexical preferences (introduced below). These parameters include the Gaussian penalty, thresholds for feature selection, etc. Details of the training procedure are described in [25].

3.7 Parsed Corpora

Over the course of about a year, Alpino has been used to parse a large amount of sentences from various corpora. We included all Dutch newspaper texts from the Twente Newspaper Corpus [18], the full Dutch Wikipedia (the version made available to the CLEF2007 participants), and the Dutch part of Europarl (available from `www.statmt.org/europarl`).

We used the 200 node Beowulf Linux cluster of the High-Performance Computing center of the University of Groningen. The dependency structures are stored in XML. The XML files can be processed and searched in various ways, for instance, using XPATH, XSLT and Xquery [3, 4]. Some quantitative information of this parsed corpus is listed in table 1. In the experiments described below, we do not distinguish between full and fragment parses; sentences without a parse are simply ignored.

4 Bilexical preferences

In this section, we describe how association scores for lexical dependencies are defined, and how the scores are applied in the disambiguation model.

Table 1: Approximate counts of the number of sentences and words in the parsed corpus. About 0.3% of the sentences did not get a parse, for computational reasons (out of memory, or maximum parse time exceeded).

number of sentences	100.0%	30,000,000
number of words		500,000,000
number of sentences without parse	0.3%	100,000
number of sentences with fragments	8.0%	2,500,000
number of single full parse	92.0%	27,500,000

Table 2: Number of lexical dependencies in parsed corpora (approximate counts)

tokens	480,000,000
types	100,000,000
types with frequency ≥ 20	2,350,000

In the first subsection, we show in detail how point-wise mutual information scores are computed on the basis of a large parsed corpus. In the second subsection, we extend lexical dependencies for an improved treatment of relative clauses and coordination. In the third subsection, we describe how the bilexical preferences are integrated in the disambiguation model.

4.1 Association Score

The parsed corpora described in the previous section have been used in order to compute association scores between lexical dependencies. The parses constructed by Alpino are dependency structures. From these dependency structures, we extract all named dependencies. In the following table, we list the number of named dependencies extracted from the parsed corpora.

Named dependencies that occur fewer than 20 times are ignored, because the mutual information score that we use below is unreliable for low frequencies. An additional benefit of a frequency threshold is a manageable size of the resulting data-structures.

Bilexical preference between two root forms w_1 and w_2 is computed using an association score based on *point-wise mutual information*, as defined by [8] and used for a similar purpose in [7], as well as in many other studies in corpus linguistics. The association score is defined here as follows:

$$I(r(w_1, w_2)) = \log \frac{f(r(w_1, w_2))}{f(r(w_1, \_))f(\_(\_, w_2))}$$

where $f(X)$ is the relative frequency of X. In the above formula, the underscore is a place holder for an arbitrary relation or an arbitrary word. The association score I compares the actual frequency of w_1 and w_2 with dependency r, with the frequency we would expect if the words were independent. For instance, to compute $I(\text{hd/obj1}(\texttt{drink}, \texttt{melk}))$ we

Table 3: Pairs involving a direct object relationship with the highest point-wise mutual information score.

bijltje gooi_neer, duimschroef draai_aan, goes by time, kostje scharrel, peentje zweet, traantje pink_weg, boontje dop, centje verdien_bij, champagne_fles ontkurk, dorst les, fikkie stook, gal spuw, garen spin, geld_kraan draai_dicht, graantje pik_mee, krediet_kraan, draai_dicht, kruis_band scheur_af, kruit verschiet, olie_kraan draai_open, onderspit delf, oven_schaal vet_in, pijp_steel regen, proef_ballonnetje laat_op, scepter zwaai, spuigat loop_uit, subsidie_kraan draai_dicht, vin verroer, wereld_zee bevaar, woordje spreek_mee

Table 4: Pairs involving a direct object relationship with the highest point-wise mutual information score for the verbs `drink` and `eat`.

biertje, borreltje, glaasje, pilsje, pintje, pint, wijntje, alcohol, bier, borrel, cappuccino, champage, chocolademelk, cola, espresso, koffie, kopje, limonade, liter, pils, slok, vruchtensap, whisky, wodka, cocktail, drankje, druppel, frisdrank, glas, jenever, liter, melk, sherry, slok, thee, wijn, blikje, bloed, drank, flesje, fles, kop, liter, urine, beker, dag, water, hoeveelheid, veel, wat

boterhammetje, hapje, Heart, mens_vlees, patatje, work, biefstuk, boer_kool, boterham, broodje, couscous, drop, frietje, friet, fruit, gebakje, hamburger, haring, home, ijsje, insect, kaas, kaviaar, kers, koolhydraat, kroket, mossel, oester, oliebol, pannenkoek, patat, pizza, rundvlees, slak, soep, spaghetti, spruitje, stam_pot, sushi, taartje, varkensvlees, vlees, aardappel, aardbei, appel, asperge, banaan, boon, brood, chocolade, chocola, garnaal, gerecht, gras, groente, hap, kalkoen, kilo, kip, koekje, kreeft, maaltijd, paling, pasta, portie, rijst, salade, sla, taart, toetje, vet, visje, vis, voedsel, voer, worst,bordje, bord, chip, dag, ei, gram, ijs, kilo, knoflook, koek, konijn, paddestoel, plant, service, stukje, thuis, tomaat, vrucht, wat, wild, zalm

look up the number of times `drink` occurs with a direct object out of all 462,250,644 dependencies (15,713) and the number of times `melk` occurs as a dependent (10,172). If we multiply the two corresponding relative frequencies, we get the expected relative frequency for hd/obj1(`drink`, `melk`). Multiplying the expected relative frequency with the corpus size (all 462M dependencies) gives an expected absolute frequency of 0.35. The actual frequency, 195, is about 560 times as big. Taking the log of 560 gives us the association score (6.33) for this bi-lexical dependency.

The pairs involving a direct object relationship with the highest scores are listed in table 3. Focusing on the verbs `drinken` (to drink) and `eten` (to eat), we provide in table 4 the corresponding highest scoring heads of objects.

Selection restrictions are often associated only with direct objects. We include bilexical association scores for all types of dependencies. We found that association scores for other types of dependencies also captures both collocational preferences as well as weaker co-occurrence preferences. Some examples including modifiers are listed in table 5. Such

Table 5: Highest scoring pairs involving a modifier relationship between a verb and an adverbial.
overlangs snijd_door, ten hele dwaal, welig tier, dunnetjes doe_over, omver kegel, on_zedelijk betast, stief_moederlijk bedeel, stierlijk verveel, straal loop_voorbij, uitein rafel, aaneen smeed, bestraf spreek_toe, cum laude studeer_af, deerlijk vergis, des te meer klem, door en door verrot, glad strijk_af, glazig fruit, hermetisch grendel_af, ingespannen tuur, instemmend knik, kat_kwaad haal_uit, kostelijk amuseer, kwistig strooi, lijdzaam zie_toe, luchtig spatel, neer plof, neer vlij, on_geforceerd verfilm, ongenadig krijg_langs, on_heus bejegen, onverdroten ga_voort, oraal bevredig, rakelings scheer, reikhals kijk_uit, standrechtelijk executeer, ten halve keer, tussenuit knijp, vergenoegd wrijf, voort borduur, voort kabbel, wagenwijd zet_open, wijd sper_open, woord voor woord zing_mee

preferences are useful for disambiguation as well. Consider the ambiguous Dutch sentence

(4) omdat we lauw bier dronken
because we cold-warm beer drank
because we drank warm beer / because we drank beer indifferently

The adjective `lauw` (cold, lukewarm, warm, indifferently) can be used to modify both nouns and verbs; this latter possibility is exemplified in:

(5) We hebben lauw gereageerd
We have cold-warm reacted
We reacted indifferently

If we incorporate bilexical preferences between heads and modifiers, then we can hope to disambiguate such cases as well.

Association scores can be negative if two words in a lexical dependency occur less frequently than one would expect if the words were independent. However, since association scores are unreliable for low frequencies (including, often, frequencies of zero), and since such negative associations involve low frequencies by their nature, we only take into account positive association scores.

4.2 Extending pairs

The CGN dependencies that we work with fail to relate pairs of words in certain syntactic constructions for which it can be reasonably assumed that bilexical preferences should be useful. We have identified two such constructions, namely relative clauses and coordination, and for these constructions we generalize our method, to take such dependencies into account too (both during dependency extraction from the parsed corpus, and during disambiguation) Consider coordinations such as:

(6) Bier of wijn drinkt Elvis niet
Beer or wine drinks Elvis not

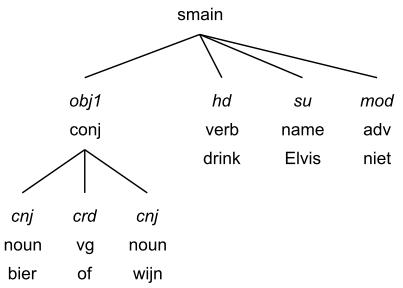

Figure 2: Dependency structure produced for coordination

Elvis does not drink beer or wine

The dependency structure of the intended analysis is given in figure 2. The set of named dependencies for this example illustrates that the coordinator is treated as the head of the conjunction:

hd/obj1(`drink`, `of`)	crd/cnj(`of`, `bier`)
crd/cnj(`of`, `wijn`)	hd/su(`drink`, `elvis`)
hd/mod(`drink`, `niet`)	

So there are no direct dependencies between the verb and the individual conjuncts. For this reason, we add additional dependencies $r(A, C)$ for every pair of dependency $r(A, B)$, crd/cnj(B, C).

Relative clauses are another syntactic phenomenon where we extend the set of dependencies. For a noun phrase such as:

(7) Wijn die Elvis niet dronk
Wine which Elvis not drank
Wine which Elvis did not drink

there is no direct dependency between `wijn` and `drink`, as can be seen in the dependency structure given in figure 3. Sets of dependencies are extended in such cases, to make the relation between the noun and the role it plays in the relative clause explicit: if a noun w_2 is modified by a relative headed by a relative pronoun, and this pronoun is a dependent r of a verb w_1, then a new dependency $r(w_1, w_2)$ is added.

4.3 Using association scores as features

The association scores for all dependencies are used in our disambiguation model by means of a technique developed by [11] to incorporate auxiliary distributions in stochastic attribute value grammar.

Auxiliary distributions offer the possibility to incorporate information from additional sources into a maximum entropy disambiguation model. In more detail, auxiliary distributions are integrated by considering the logarithm of the probability given by an auxiliary

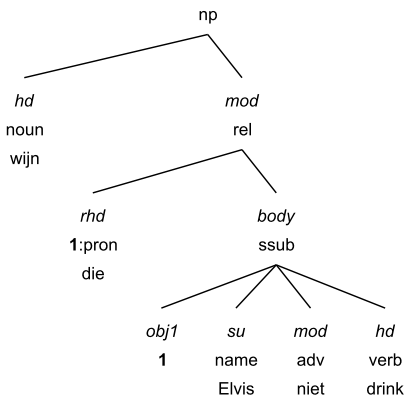

Figure 3: Dependency structure produced for relative clause

distribution as an additional, real-valued feature. More formally, given k auxiliary distributions $Q_i(t)$, then k new *auxiliary features* $f_{m+1}, ..., f_{m+k}$ are added such that

$$f_{m+i}(t) = \log Q_i(t)$$

In [11] it is noted that $Q_i(t)$ do not need to be proper probability distributions, however they must be strictly positive. In our case, we use auxiliary distributions $Q_{t,r}$ for each of the major POS-tag labels t (verb, noun, adjective, adverb, ...) and each of the dependency relations r (subject, object, ...).

More concretely, we introduce additional auxiliary features $z(t, r)$ for each of the major POS labels t and each of the dependency relations r. The 'count' of such features is determined by the association scores for actually occuring dependency pairs. For example, if in a given parse a given verb v has a direct object dependent n, then we compute the association of this particular pair, and use the resulting number as the count of that feature. Of course, if there are multiple dependencies of this type in a single parse, the corresponding association scores are all summed, to arrive at the count for the feature $z(t, r)$.

To illustrate this technique, consider the dependency structure given earlier in figure 2. For this example, there are four of these new features with a non-zero count. The counts are given by the corresponding association scores as follows:

$$\begin{aligned}
z(\text{verb}, \text{hd/su}) &= I(\text{hd/su}(\texttt{drink}, \texttt{elvis})) \\
z(\text{verb}, \text{hd/mod}) &= I(\text{hd/mod}(\texttt{drink}, \texttt{niet})) \\
z(\text{verb}, \text{hd/obj1}) &= I(\text{hd/obj1}(\texttt{drink}, \texttt{of})) \\
&+ I(\text{hd/obj1}(\texttt{drink}, \texttt{bier})) \\
&+ I(\text{hd/obj1}(\texttt{drink}, \texttt{wijn})) \\
z(\text{conj}, \text{crd/cnj}) &= I(\text{crd/cnj}(\texttt{of}, \texttt{bier})) \\
&+ I(\text{crd/cnj}(\texttt{of}, \texttt{wijn}))
\end{aligned}$$

It is crucial to observe that the new features do not include any direct reference to actual words. This means that there will be only a fairly limited number of new features

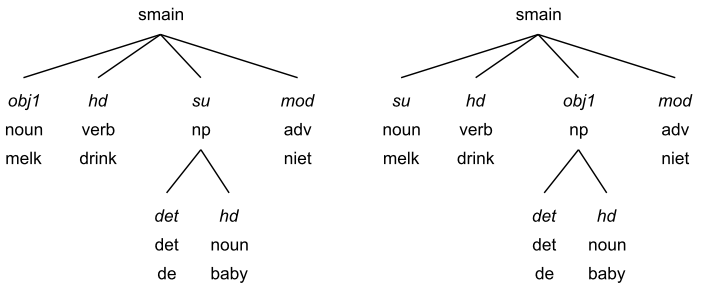

Figure 4: Two competing dependency structures for the sentence *Melk drinkt de baby niet*

Table 6: Relevant features and their counts and weights for two readings of the sentence *Melk drinkt de baby niet*

Correct reading				Wrong reading			
feature	count	weight	sum	feature	count	weight	sum
non-subj-topic	1	-0.015	-0.015	subj-topic	1	+0.043	+0.043
z(verb,hd/obj1)	6	+0.009	+0.054				
z(verb,hd/su)	4	+0.010	+0.040				
		ϕ	+0.079			ϕ	+0.043

(depending on the number of tags t and relations r, in the experiments below there are slightly over 100 new features), and we can expect that these features are frequent enough to be able to estimate their weights in training material of limited size.

With the new features present, the model is re-trained on the original training data. As a result, the features including the new $z(t, r)$ features are assigned weights. These new weights can then be used in parse selection. As an example, consider the two parses in figure 4 for sentence (1), repeated here for convenience:

(8) Melk drinkt de baby niet
Milk drinks the baby not
The baby doesn't drink milk / Milk doesn't drink the baby

In table 6 we show some of the relevant features to distinguish the two readings, with the corresponding counts and weights. In this example, the bias of the model for topicalized subjects is properly out-weighted by the inclusion of the new lexical preference features. Therefore the model correctly selects the desired reading in this case.

5 Experiments

We report on two experiments. In the first experiment, we report on the results of ten-fold cross-validation on the Alpino treebank. This is the material that is standardly used for training and testing. For each of the sentences of this corpus, the system produces

Table 7: Results with ten-fold cross-validation on the Eindhoven-cdbl part of the Alpino treebank. In these experiments, the models are used to select a parse from a given set of at most 1000 parses per sentence.

	fscore %	error-reduction %	exact %	CA %
baseline	74.02	0.00	16.0	73.48
oracle	91.97	100.00	100.0	91.67
standard	87.41	74.60	52.0	87.02
+bilexical preferences	87.91	77.38	54.8	87.51

at most the first 1000 parses. For every parse, we compute the quality by comparing its dependency structure with the gold standard dependency structure in the treebank. For training, at most 100 parses are selected randomly for each sentence. For (ten-fold cross-validated) testing, we use all available parses for a given sentence. In order to test the quality of the model, we check for each given sentence which of its 1000 parses is selected by the disambiguation model. The quality of that parse is used in the computation of the accuracy, as listed in table 7. The column labeled *exact* measures the proportion of sentences for which the model selected (one of) the best possible parse(s) (there can be multiple best possible parses). The *baseline* row reports on the quality of a disambiguation model which simply selects the first parse for each sentence. The *oracle* row reports on the quality of the best-possible disambiguation model, which would (by magic) always select the best possible parse (this number is lower than 100, because some parses are outside the coverage of the system, and some parses are generated only after more than 1000 inferior parses). The *error reduction* column measures which part of the disambiguation problem (difference between the baseline and oracle scores) is solved by the model.[1]

The results show a small, but clear, increase in error reduction, if the standard model (without the association score features) is compared with a (retrained) model that includes the association score features. The relatively large improvement of the *exact* score suggests that the bilexical preference features are particularly good at choosing between very good parses.

For the second experiment, we evaluate how well the resulting model performs in the full system. First of all, this is the only really convincing evaluation which measures progress for the system as a whole by virtue of including bilexical preferences. The second motivation for this experiment is for methodological reasons: we now test on a truly unseen test-set. The first experiment can be criticized on methodological grounds as follows. The Alpino treebank was used to train the disambiguation model which was used to construct the large parsed treebank from which we extracted the counts for the association scores. Those scores might somehow therefore indirectly reflect certain aspects of the Alpino treebank training

[1]Note that the error reduction numbers presented in the table are lower than those presented in [25]. The reason is that we report here on experiments in which parses are generated with a version of Alpino with the POS-tagger switched on. The POS-tagger already reduces the number of ambiguities, and in particular solves many of the 'easy' cases. The resulting models, however, are more effective in practice (where the model also is applied after the POS-tagger).

Table 8: Results on the WR-P-P-H part of the D-Coi corpus (2267 sentences from the newspaper Trouw, from 2001). In these experiments, we report on the full system. In the full system, the disambiguation model is used to guide a best-first beam-search procedure which extracts a parse from the parse forest. Difference in CA was found to be significant (using paired T-test on the per sentence CA scores).

	precision %	recall %	fscore %	CA %
standard	90.77	90.49	90.63	90.32
+bilexical preferences	91.19	90.89	91.01	90.73

data. Testing on that data later (with the inclusion of the association scores) is therefore not sound.

For this second experiment we used the WR-P-P-H (newspaper) part of the D-Coi corpus. This part contains 2256 sentences from the newspaper Trouw (2001). In table 8 we show the resulting f-score and CA for a system with and without the inclusion of the $z(t, r)$ features. The improvement found in the previous experiment is confirmed.

6 Conclusion and Outlook

One might wonder why self-training works in the case of selection restrictions, at least in the set-up described above. One may argue that, in order to learn that *milk* is a good object for *drink*, the parser has to analyze examples of *drink milk* in the raw data correctly. But if the parser is capable of analyzing these examples, why does it need selection restrictions? The answer appears to be that the parser (without selection restrictions) is able to analyze the large majority of cases correctly. These cases include the many easy occurrences where no (difficult) ambiguities arise (case marking, number agreement, and other syntactic characteristics often force a particular reading). The easy cases outnumber the mis-parsed difficult cases, and therefore the selection restrictions can be learned. Using these selection restrictions as additional features, the parser is then able to also get some of the difficult, ambiguous, cases right.

There are various aspects of our method that need further investigation. First of all, existing techniques that involve selection restrictions (e.g., [20]) typically assume classes of nouns, rather than individual nouns. In future work, we hope to generalize our method to take classes into account, where the aim is to learn class membership also on the basis of large parsed corpora.

Another aspect of the technique that needs further research involves the use of a threshold in establishing the association score, and perhaps related to this issue, the incorporation of negative association scores (for instance for cases where a large number of co-occurrences of a pair would be expected but where in fact none or very few were found).

There are also some more practical issues that perhaps had a negative impact on our results. First, the large parsed corpus was collected over a period of about a year, but during that period, the actual system was not stable. In particular, due to various improvements

of the dictionary, the root form of words that was used by the system changed over time. Since we used root forms in the computation of the association scores, this could be harmful in some specific cases. A further practical issue concerns repeated sentences or even full paragraphs. This happens in typical newspaper material for instance in the case of short descriptions of movies that may be repeated weekly for as long as that movie is playing. Pairs of words that occur in such repeated sentences receive association scores that are much too high. The method should be adapted to take this into account, perhaps simply by removing duplicated sentences.

The association scores are defined with respect to root forms. This may not be optimal. In our dictionary, verbs are often associated with many different subcategorization frames. Sometimes, the meaning of a verb can be dependent on the choice of subcategorization frame. For instance, the meaning of the intransitive use of *eindigen* (to end) is quite different from its transitive use, as the following two examples illustrate:

(9) a. Het verhaal eindigt hier
The story ends here
The story ends here
b. Hij eindigde zijn voordracht
He ended his presentation
He ended his presentation

In the ideal case, we might want to have access to the information that, for this verb, the subject phrase in the intransitive use of the verb is thematically related to the direct object of the transitive use of the verb. Currently, this information is not available to the system; rather the subjects of both the intransitive as well as the transitive use of the verb are all treated together.

A better alternative might be, to define mutual information scores with respect to pairs of root forms and subcategorization frames; however this would probably be harmful for cases such as *eten* (to eat), where the subject of both the transitive and intransitive use of the verb appear to share the thematic role. One interesting direction would be to try integrate the research on automatic, corpus-based, verb classification [17, 22, 15].

The insight that selection restrictions are useful for parsing is not new. However, as far as we know this is the first time that automatically acquired selection restrictions have been shown to improve parsing accuracy results for a wide-coverage full parsing task.

Appendix: practical details

Extracting information from the Lassy Large treebank can be done in a number of ways. Since the syntactic analyses are all stored in XML, a natural choice is to use XSLT stylesheets, or XQuery scripts. These options are the most general. An alternative approach that is explored here, is the use of the Alpino parser itself. Alpino is capable of extracting dependency triples of a given XML-file containing a dependency relation. A

command such as:

```
Alpino  -treebank_dep_features Treebank/cdb/4.xml
```

will output the dependency features found in the XML file specified on the command line.

The output contains more information than we need for our purposes. The interesting lines are the lines of the following type:

```
dep35(DepRoot,DepPos,Rel,Pos,Root)
ldep35(DepRoot,DepPos)
rdep35(Rel,Pos,Root)
tdep35
```

For the correct analysis of example (1) we will get the following output:

```
dep35(melk,noun,hd/obj1,verb,drink)
dep35(baby,noun,hd/su,verb,drink)
dep35(de,det,hd/det,noun,baby)
dep35(niet,adv,hd/mod,verb,drink)
tdep35
ldep35(melk,noun)
rdep35(hd/obj1,verb,drink)
tdep35
ldep35(baby,noun)
rdep35(hd/su,verb,drink)
tdep35
ldep35(de,det)
rdep35(hd/det,noun,baby)
tdep35
ldep35(niet,adv)
rdep35(hd/mod,verb,drink)
```

The actual dependencies are output as `dep35` terms. The other terms can be used for normalization purposes. The `ldep` term is output for every dependent of a dependency pair. The `rdep` term is output for every head of a dependency pair. Finally, somewhat redundantly, `tdep35` is output for every dependency pair.

How can we produce this type of output for all of the XML files of the Lassy Large corpus? The `dtlist` command, part of the Alpino distribution, is very useful here. It will list the names of all of the XML files in a given (part of the) corpus, irrespective of whether the corpus is stored as single XML files, or as a compresses concatenation of many XML files. The following command shows how this works:

```
dtlist --recursive SONAR/
```

if this command is executed in the relevant directory of the Lassy Large distribution, then the names of all the XML files in the SONAR sub-corpus will be listed on standard output. If we want all files in one step, we can simply do:

```
dtlist --recursive .
```

. . . and we can wait until all the millions of file names have been listed.

In order to combine this with the command to extract the dependencies given above, we use `xargs`:

```
dtlist --recursive . \
     | xargs Alpino -treebank_dep_features
```

After we select the appropriate terms from the output, what remains is simply a matter of sorting and counting, to get the information that is needed:

```
dtlist --recursive . \
     | xargs Alpino -treebank_dep_features \
     | grep ^dep35 \
     | sort \
     | uniq -c
```

A similar command is required for the other terms of interest. The counts collected in this way are all we need to compute the pointwise mutual information scores required in the proposal above.

Appendix: more examples

Here we list a number of examples, which suggest that selection restrictions can also be important for dependencies, other than direct objects.

High scoring pairs involving a subject relationship with a verb:

alarmbel	rinkel
champagnekurk	knal
gij	echtbreek
haan	kraai
kikker	kwaak
rups	verpop
vonk	overspring
zweet	parel
belletje	rinkel
brievenbus	klepper

High scoring pairs involving a modifier relationship with a noun:

in vitro	fertilisatie
Hubble	ruimtetelescoop
zelfrijzend	bakmeel
bezittelijk	voornaamwoord
ingegroeid	teennagel
knapperend	haardvuur
levendbarend	hagedis
onbevlekt	ontvangenis
ongeblust	kalk

High scoring pairs involving a predicative complement relationship with a verb:

beetgaar	kook
beuk	murw
schuimig	klop
suf	peins
suf	pieker
doormidden	scheur
ragfijn	hak
stuk	bijt
au serieux	neem
in duigen	val
lam	leg

High scoring pairs involving an apposition relationship with a noun:

jongensgroep	Boyzone
communicatiesysteem	C2000
blindeninstituut	De Steffenberg
haptonoom	Ted Troost
gebedsgenezeres	Greet Hofmans
rally	Parijs-Dakar
tovenaar	Gandalf
aartsengel	Gabriel
keeperstrainer	Joep Hiele
basketbalcoach	Ton Boot
partizaan	Tito

High scoring pairs involving a measure phrase relationship with an adjective:

graadje	erger
lichtjaar	verwijderd
mijlenver	verwijderd
niets	liever
eindje	verderop
graad	warmer
illusie	armer
kilogram	wegend
onsje	minder
maatje	te groot
knip	waard

References

[1] Takeshi Abekawa and Manabu Okumura. Japanese dependency parsing using co-occurrence information and a combination of case elements. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 833–840, Sydney, Australia, July 2006. Association for Computational Linguistics.

[2] M. Boros, W. Eckert, F. Gallwitz, G. Görz, G. Hanrieder, and H. Niemann. Towards understanding spontaneous speech: Word accuracy vs. concept accuracy. In *Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP 96)*, Philadelphia, 1996.

[3] Gosse Bouma and Geert Kloosterman. Querying dependency treebanks in XML. In *Proceedings of the Third international conference on Language Resources and Evaluation (LREC)*, pages 1686–1691, Gran Canaria, Spain, 2002.

[4] Gosse Bouma and Geert Kloosterman. Mining syntactically annotated corpora using XQuery. In *Proceedings of the Linguistic Annotation Workshop*, Prague, June 2007. ACL.

[5] Ted Briscoe, John Carroll, Jonathan Graham, and Ann Copestake. Relational evaluation schemes. In *Proceedings of the Beyond PARSEVAL Workshop at the 3rd International Conference on Language Resources and Evaluation*, pages 4–8, Las Palmas, Gran Canaria, 2002.

[6] Kenneth W. Church, William A. Gale, Patrick Hanks, and Donald Hindle. Parsing, word association and typical predicate argument relations. In *1st International Workshop on Parsing Technologies (IWPT '89)*, Carnegie Mellon University, Pittsburg, 1989.

[7] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

[8] Robert Mario Fano. *Transmission of Information: A Statistical Theory of Communications*. MIT Press, Cambridge, MA, 1961.

[9] Pablo Gamallo, Alexandre Agustini, and Gabriel P. Lopes. Learning subcategorisation information to model a grammar with "co-restrictions". *TAL*, 44(1):93–117, 2003.

[10] Heleen Hoekstra, Michael Moortgat, Bram Renmans, Machteld Schouppe, Ineke Schuurman, and Ton van der Wouden. *CGN Syntactische Annotatie*, December 2003.

[11] Mark Johnson and Stefan Riezler. Exploiting auxiliary distributions in stochastic unification-based grammars. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 154–161, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[12] Daisuke Kawahara and Sadao Kurohashi. A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 176–183, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[13] Daisuke Kawahara and Sadao Kurohashi. Coordination disambiguation without any similarities. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 425–432, Manchester, August 2008.

[14] Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, Taipei, 2002.

[15] Diana McCarthy. *Lexical Acquisition at the Syntax-Semantics Interface: Diathesis ALternations, Subcategorization Frames and Selectional Preferences*. PhD thesis, University of Sussex, 2001.

[16] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA, June 2006. Association for Computational Linguistics.

[17] Paola Merlo and Suzanne Stevenson. Automatic verb classification based on statistical distributions of argument structure. *Computational Linguistics*, 27(3):373–408, 2001.

[18] Roeland Ordelman, Franciska de Jong, Arjan van Hessen, and Hendri Hondorp. TwNC: a multifaceted Dutch news corpus. *ELRA Newsletter*, 12(3/4):4–7, 2007.

[19] Robbert Prins. *Finite-State Pre-Processing for Natural Language Analysis*. PhD thesis, University of Groningen, 2005.

[20] Philip Stuart Resnik. *Selection and information: a class-based approach to lexical relationships*. PhD thesis, University of Pennsylvania, Philadelphia, PA, USA, 1993.

[21] Francesc Ribas. On learning more appropriate selectional restrictions. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, pages 112–118, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[22] Sabine Schulte im Walde. The induction of verb frames and verb classes from corpora. In Anke Lüdeling and Merja Kytö, editors, *Corpus Linguistics. An International Handbook*. Mounton de Gruyter, Berlin, to appear.

[23] Gertjan van Noord. **A**t **L**ast **P**arsing **I**s **N**ow **O**perational. In *TALN 2006 Verbum Ex Machina, Actes De La 13e Conference sur Le Traitement Automatique des Langues naturelles*, pages 20–42, Leuven, 2006.

[24] Gertjan van Noord. Self-trained bilexical preferences to improve disambiguation accuracy. In Paola Merlo Harry Bunt and Joakim Nivre, editors, *Trends in Parsing Technology. Dependency Parsing, Domain Adaptation, and Deep Parsing*. Springer, 2010. to appear.

[25] Gertjan van Noord and Robert Malouf. Wide coverage parsing with stochastic attribute value grammars. Draft available from http://www.let.rug.nl/~vannoord. A preliminary version of this paper was published in the Proceedings of the IJCNLP workshop Beyond Shallow Analyses, Hainan China, 2004., 2005.

[26] Gertjan van Noord, Ineke Schuurman, and Vincent Vandeghinste. Syntactic annotation of large corpora in STEVIN. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy, 2006.

[27] Greg Whittemore, Kathleen Ferrara, and Hans Brunner. Empirical study of predictive powers of simple attachment schemes for post-modifier prepositional phrases. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 23–30, Pittsburgh, Pennsylvania, USA, June 1990. Association for Computational Linguistics.