Induction of Treebank-Aligned Lexical Resources

Tejaswini Deoskar and Mats Rooth

Department of Linguistics
Cornell University
td72, mr249 @cornell.edu

Abstract

We describe the induction of lexical resources from unannotated corpora that are aligned with treebank grammars, providing a systematic correspondence between features in the lexical resource and a treebank syntactic resource. We first describe a methodology based on parsing technology for augmenting a treebank database with linguistic features. A PCFG containing these features is created from the augmented treebank. We then use a procedure based on the inside-outside algorithm to learn lexical resources aligned with the treebank PCFG from large unannotated corpora. The method has been applied in creating a feature-annotated English treebank based on the Penn Treebank. The unsupervised estimation procedure gives a substantial error reduction (up to 31.6%) on the task of learning the subcategorization preference of novel verbs that are not present in the annotated training sample.

1. Introduction

The standard treebanks that are being created for the world's languages consist of labeled trees with indexed empty categories. The choice of a simple formal vocabulary for treebank annotation has advantages–for instance it enables simple search methods and search languages, and has spurred research on statistical parsing. For some other purposes, including aspects of linguistic research, lexicographic research and development, and research on high-end parsing, it is a drawback that features such as inflectional category, lemmas, sub-classification of clausal categories, subcategorization frames of verbs and nouns, and localized information about long distance dependencies are not overtly available in treebank annotations. In addition, lexical resources where there is a systematic correspondence between the lexicon and such fine-grained annotation in a treebank database would also be valuable. For instance, the lexical resource may contain features representing the subcategorization frames of verbs, which correspond to structural configurations that the verb occurs in, in a treebank. Given such an alignment, a treebank can be compiled into a lexicon by collecting the combinations of lexical entries and their local features found in the treebank. This paper focuses on these two problems: one focus is a development framework which allows existing treebanks to be annotated by linguistically relevant features. The framework includes creation of an augmented treebank database and the creation of treebank PCFGs, including probabilistic lexicons, based on the augmented treebank. The second focus is the induction from unannotated data of treebank-aligned probabilistic lexicons which encode lexical features such as verbal valence (subcategorization) in pre-terminal symbols. In this paper, we focus on learning verbal valence; however there are many such lexically oriented features for which treebank data is sparse, such as the attachment preferences of adverbs to nominal, verbal or sentential nodes, the valence of nouns and subclasses of adjectives. We use a method based on constraint solving to add feature annotations to the Penn Treebank of English (Marcus et al., 1993). Features are then incorporated in the symbols of a context free grammar and frequencies are collected, resulting in a probabilistic grammar and a probabilistic lexicon which encodes lexical features.

Previous research has argued that because of sparseness of lexical distributions, computational lexicons derived from corpora should be based on very large corpus samples, much larger than the roughly 50,000-sentence Penn Treebank (for example, (Briscoe and Carroll, 1997)). (Beil et al., 1999; im Walde, 2002) demonstrated that PCFG grammars and lexicons with incorporated valence features could be improved by iterative EM estimation; however their grammar was not a treebank grammar, and therefore could not be evaluated using standardized evaluation criteria. Our treebank-aligned grammar and lexicon allows us to evaluate lexical learning using a held-out portion of the treebank for testing.

On the task of identifying the valence of token occurrences of novel verbs, we get up to 23.38% error reduction following a standard inside-outside estimation procedure (Lari and Young, 1990). A modified inside-outside procedure which re-estimated lexical parameters while retaining syntactic parameters in the PCFG gives a reduction in error rate of 31.6%.

In the sections to follow, we first describe our methodology for augmenting treebanks(§2.), building a PCFG from the augmented treebank (§3.) and then a procedure based on the inside-outside algorithm to re-estimate the treebank-aligned lexicon using unannotated data (§4.). Finally, we present evaluations of the resulting lexicon on the task of verbal subcategorization detection. We also discuss the modularity of the components of the system, and alternate uses of the different modules in the distribution of the system.

2. Treebank Feature Augmentation

Our methodology for augmenting the treebank with features involves constraint-solving with a feature grammar whose backbone is the context-free grammar obtained from the treebank. First a feature constraint grammar is constructed by adding feature-constraints to a grammar obtained from the vanilla treebank. The formalism used is the Yap feature-constraint formalism (Schmid, 2000). The

feature constraint annotations are similar in some respects to those used in LFG frameworks like (O'Donovan et al., 2005)– however, unlike (O'Donovan et al., 2005) who create a treebank LFG grammar, our goal is to use this methodology for treebank transformation and to realize a PCFG in the end. In the first step of the transformation process, for each tree in the treebank, a trivial context-free shared forest is constructed that represents a single tree. In the second stage, the shared forest is passed to a constraint solver, which solves constraints in the shared forest. This stage adds features and may split a tree into several solutions, writing out a feature shared forest. For solving constraints, we use the parser Yap (Schmid, 2000), and a feature-constraint grammar that we create using the vanilla treebank grammar as backbone.

2.1. Feature Constraint Grammar

A context free grammar containing approximately $30,000$ rules is obtained as the set of local tree configurations found in a training portion of the treebank (in this paper, Sections 0-22 of the Penn Treebank). Feature constraints are added to the rules using Perl and Lisp programs which examine patterns in the rule shapes. The constraints assign constant values to features or specify feature equalities. As an illustration, consider auxiliary verb constructions. In the Penn Treebank II convention, any local tree with a VP parent and having both VP and verb children is an auxiliary verb construction. Hence, constraints suitable to auxiliary verb constructions should be added to any rule of this form. In the example below, the first line gives the original context free rule, and the subsequent lines the corresponding feature constraint rule. Feature constraints in the Yap formalism are enclosed in braces associated with context free categories. A constraint consists of a feature name, an equal sign and a value, followed by a semi-colon, and with possible feature values being variables, constants, lists, etc. (Schmid, 2000).

```
VP    ->     VB ADVP VP

VP {Vform=base; Slash=sl;} ->
         'VB { Val=aux; Prep=-; Vsel=vf;
               Prtcl=-; Sbj=-; }
          ADVP {}
          VP { Slash=sl; Vform=vf; }
```

The above rule is for auxiliary VPs. A VP licensed by the rule may have a variable `Vform` on the complement VP (the VP on the RHS of the above rule) which correlates with the particular auxiliary verb (VB on RHS). For instance, a progressive *be* conditions the present participle `Vform` value `g`). The dependence is expressed in the above rule by using a variable `vf` to match the `Vform` feature marked on the complement VP with the value of a `Vsel` feature marked on the auxiliary verb. The verb is marked as an auxiliary verb with the constraint `Val=aux`, using the `Val` attribute which is also used to describe the valences of main verbs. A slash feature is used in the standard way to express wh dependencies; the rule matches the `Slash` values of the parent and child VPs using a variable `sl`. The grammar includes features which constrain the distribution of common empty categories. The `Prep` (preposition), `Prtcl` (particle) and `Sbj` (subject) features on the verb have a default value in this rule—for VP constructions involving main verbs (non-auxiliary constructions), `Prep` gets a non-default value if the verb has a PP complement, `Prtcl` gets a non-default value for particle verbs, and `Sbj` characterizes the subject of an S complement. The backquote on VB is a head marking that is required by the Yap formalism, but which is irrelevant to our feature constraint grammar.

The next example illustrates the encoding of valence (subcategorization) on verbs. The VP rule below introduces the trace of a passive NP, which in our notation is the empty category +EI-NP+. A valence feature `Val` is marked on verbs. The valence value `ns` on the past tense verb VBD indicates a combination of NP and S complements. We use a vocabulary of 31 basic valence values. A partial key for the valence feature values used in examples in the paper is shown in Figure 1. The slash value is matched between the parent and the S child. The equation `Vform=n` on the parent identifies the passive verb phrase. In general, dependencies such as passive and raising are constrained with local features such as `Vform` and `Vsel`, reserving the slash feature for A-bar dependencies.

```
VP    ->    VBD  +EI-NP+  PP-TMP

VP {Vform=n; Slash=sl;} ->
       'VBD { Val=ns; Vsel=vf; Prep=-;
              Prtcl=-; Sbj=-; }
        +EI-NP+
        S { Sbj=X; Slash=sl; Vform=vf; };
```

This rule also illustrates the strategy of projecting information about the tree shape into a lexical item. A past tense (VBD) verb occurring in the above configuration will get marked with a specific valence value `ns`, and also with `Sbj` and `Vsel` values copied from the S complement, and default values for the features `Prep` and `Prtcl`.

The basic valence (indicated by `Val`) is sub-classified by additional features. The `Vsel` feature marks the `Vform` of a complement S, or for auxiliary verbs, the complement VP. This distinguishes, for instance, control verbs like *try* which select an S with `Vform=to`. The `Sbj` feature marks whether the complement, if it is an S, has an empty subject. For example, a control use of the verb *try* has `Sbj=ei`, marking an indexed null subject. The verb *considered* in the treebank sentence *they are officially considered strategic* gets pre-terminal values of `Val=s`, `Sbj=e`, and `Vsel=sc`. These values indicates a clausal complement (`s`) which has an empty subject (`e`) since the sentence is passive and is of the type *small clause* (`sc`). There are 81 realized combinations of values for `Val`, `Vsel`, and `Sbj`, providing a moderately fine-grained classification of valences. The features `Prtcl` and `Prep` further sub-classify verbs with particle and prepositional complements, by indicating the particular choice of particle or preposition. In a method described in the next section, the effect will be to construct a lexicon with fairly specific information about the tree shapes associated with lexical items, using information implicit in the treebank.

We also use features which are tree-geometric rather that linguistic in nature, in the style of (Johnson, 1998; Klein

z.-.-.-	intransitive	n.-.-.-	NP
p.-.-.-	PP	np.-.-.-	NP PP
s.-.-.-	S	b.-.-.-	SBAR
t.-.-.-	-PRD (predicate complement)		
s.e.to	control		
s.-.sc	active small clause complement		
s.e.sc	passive small clause complement		

Table 1: Verbal subcategorization features

and Manning, 2003)). These are relevant to producing a good PCFG model. An example is the `Vdom` feature marked on ADJP-PRD (predicative adjective phrase) in the rule below. The value `vd` for this feature has the interpretation that the bearer of the feature directly or indirectly dominates VP. Similarly, the `parent` attribute on PP is tree-geometric contextual feature marking the upward context.

```
ADJP-PRD      ->  ADJP  PP  S

ADJP-PRD {vdom=vd;} ->
   'ADJP {} PP { parent=adjp;} S {};
```

Since the feature constraint grammar is based on a treebank backbone, it has a large number of rules, and since the Yap formalism does not allow for factoring of constraints in an inheritance hierarchy, the rules have redundant patterns of feature constraints. This has some disadvantages in grammar development, since there is no concise localization of a given constraint. At the same time, the development environment proves to be a comfortable one, because the treebank nearly eliminates the issue of ambiguity. This allows the computational linguist to concentrate on correct analyses while developing the constraint grammar. We envision this setup as a simple and easily deployable platform for augmenting existing treebanks with features, creating lexical resources, and parsing.

The design for the grammar is largely motivated by the PCFG compilation application described in the next section. This has consequences for the complexity of the feature analyses—notably, only atomic-valued features are employed. It is clear that a grammar at this limited level of complexity misses linguistically real phenomena, and thus should be regarded as an approximation. The aim is to strike a balance between linguistic sophistication and computational and mathematical simplicity and tractability.

Figures 1 and 2 show sample trees in the transformed treebank. The node labels and tree shape are as in the treebank, except for simple transformations related to empty categories. The additional information consists of the atomic-valued features which annotate each node in a tree. The feature values make explicit certain information which is implicit in the treebank. Importantly, features are marked on lexical items—for instance a valence feature is marked on verbs. This is the basis for the procedure for compiling PCFG lexicons that have these features on lexical entries (described in §3.) and for re-estimating better parameter values for them (described in §4.).

tried	VBD.s.e.to.- 32.0	VBN.s.e.to.- 11.0
	VBN.n.-.-.- 5.0	VBD.z.-.-.- 1.0
	VBD.n.-.-.- 1.0	VBD.s.e.g.- 1.0
	VBN.z.-.-.- 1.0	
attacked	VBN.n.-.-.- 5.0	VBN.np.-.-.as 1.0
	VBD.np.-.-.as 2.0	VBD.z.-.-.- 1.0
attain	VB.n.-.-.- 2.0	
attest	VB.b.-.-.- 1.0	

Figure 3: Entries of verbs in the PCFG lexicon. The last two entries illustrate sparseness of the treebank lexicon.

29092.0	ROOT S.fin.-.-.root
14134.0	S.fin.-.-.- NP-SBJ.nvd.base.-.-.- VP.fin.-.-
13057.0	NP-SBJ.nvd.base.-.-.- PRP
13050.0	PP.nvd.of.np IN.of NP.nvd.base.-.-.-.-
11226.0	S.fin.-.-.root NP-SBJ.nvd.base.-.-.- VP.fin.-.- -PER-.stop
10760.0	VP.to.-.- TO VP.base.-.-

Figure 4: Syntactic rule frequencies in the treebank PCFG

3. PCFG Compilation and Parsing application

In treebank parsing applications, PCFGs are often created by incorporating features into context free grammar symbols (for example,(Klein and Manning, 2003)). We use a method which compiles a frequency table for a PCFG from the feature annotated treebank database (Privman, 2003). For each symbol, a list of attributes to be incorporated into the symbol is stipulated. For instance, it may be stipulated that VP incorporates the attributes `Vform` and `Slash`, and that verbs incorporate `Val`, `Vform` and `Sbj`. A program reads the shared forest structures produced by constraint solving, and collects frequencies of occurrences of local tree configurations, including context free symbols and incorporated features. In cases where constraint solving introduced ambiguity, frequencies are split by a non-probabilistic version of the inside-outside algorithm (the ratio algorithm). The result is a rule frequency table and frequency lexicon which can be used by a probabilistic parser. Figures 3 and 4 illustrate entries in the PCFG lexicon and grammar respectively.

PCFGs derived in this way can be used by a parser to construct maximal probability (Viterbi) parses. We evaluate the quality of the PCFG extracted from the transformed treebank using standard PARSEVAL measures. We obtain maximum probability (Viterbi) parses for all sentences in the standard test section of the Penn Treebank (Section 23), using the parser Bitpar (Schmid, 2004). Table 2 shows the labeled bracketing scores for an optimal combination of features incorporated in the PCFG symbols. The labeled bracketing scores are comparable to state-of-the-art unlexicalized grammars. Figure 5 shows an example viterbi parse using the PCFG. Incorporating different features into the PCFG changes the labeled bracketing score of the PCFG; recall that our framework allows us to stipulate which features from the feature constraint grammar are to be incorporated into the symbols of the PCFG. To illustrate this point, consider features on verbs and nouns related to valence. In

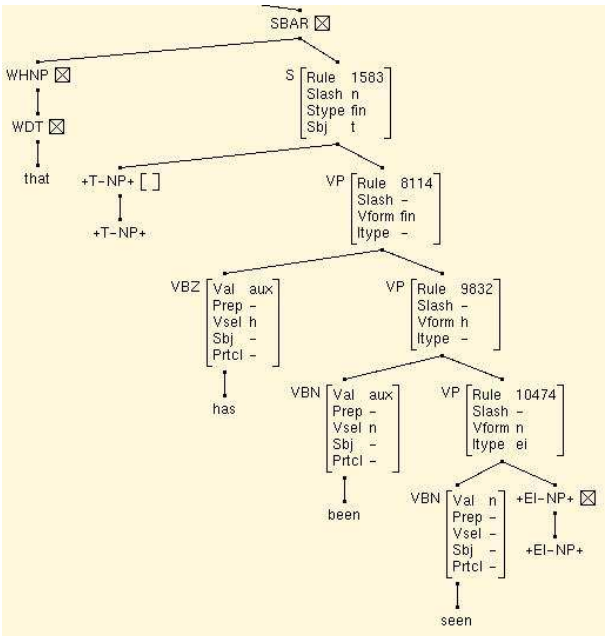

Figure 1: A relative clause in the transformed treebank: Empty categories are flanked by plus signs.

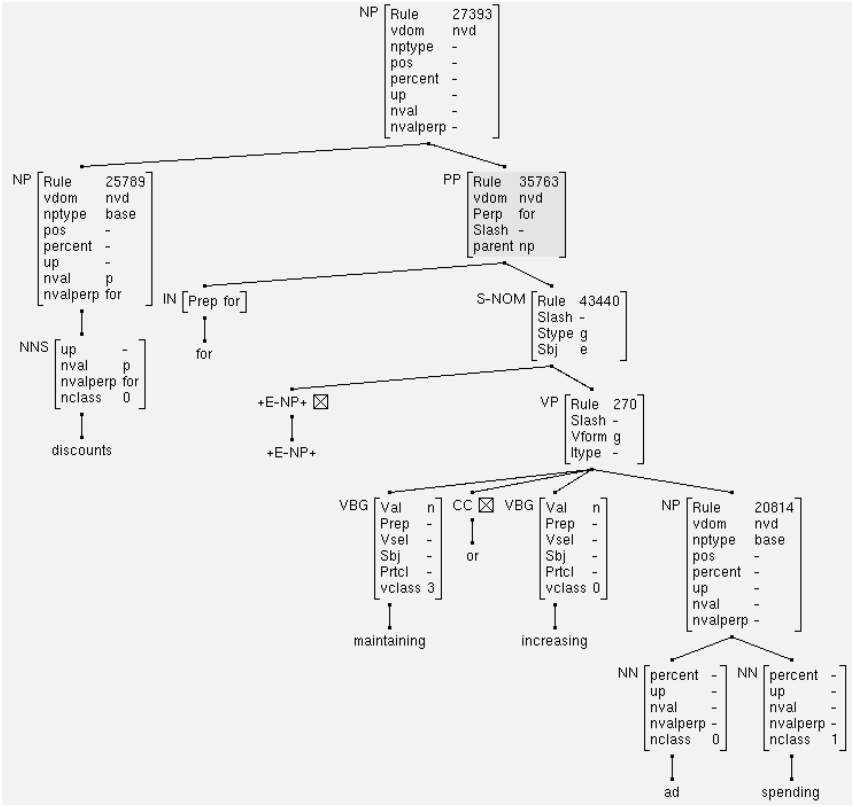

Figure 2: Prepositional complements of nouns are marked on the noun *discounts* (nval=p) along with the preposition (nvalperp=for)

the grammar version whose scores are reported in Table 2, features incorporated on verb and noun categories do not include the specific preposition for prepositional subcategorization frames. In another version of the grammar, we include specific prepositions from the prepositional complement into the verbal and nominal subcategorization frame. The effect of including specific prepositions on these categories may be to make the grammar too sparse, resulting in the reduction of the labeled bracketing score seen in Table 3. Nevertheless, including these features is interesting from the point of view of creation of lexical resources, since it enriches the lexical information that is represented.

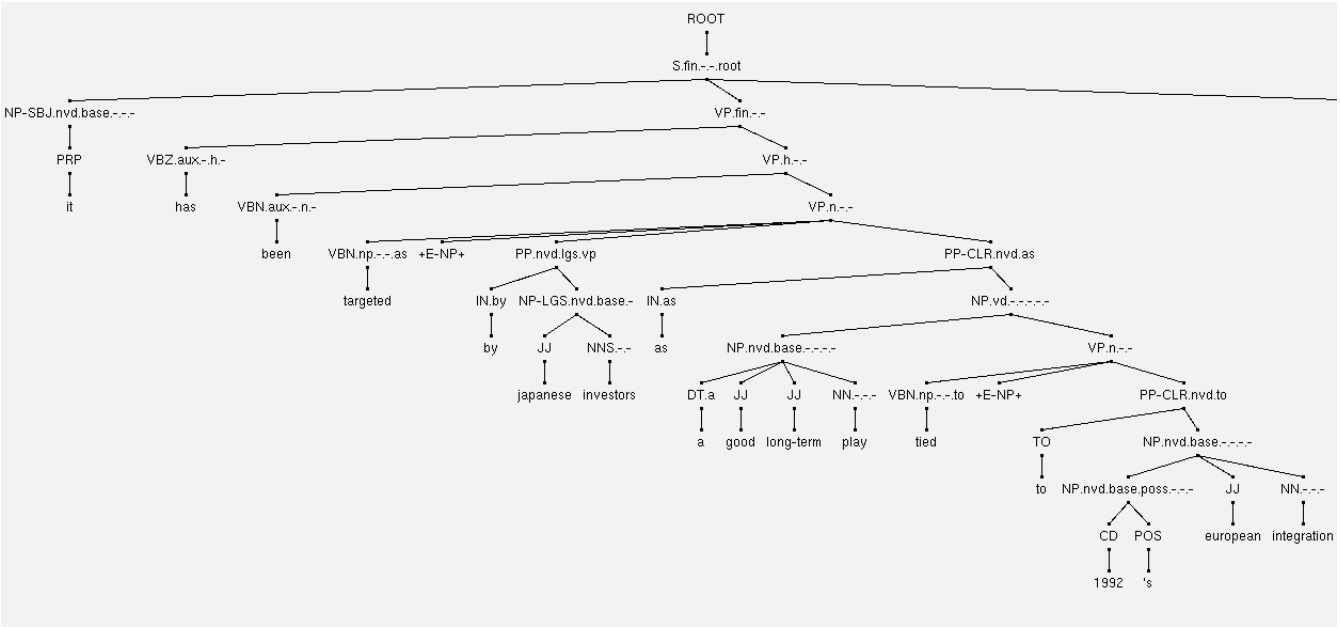

Figure 5: A viterbi parse tree generated by the PCFG.

	this paper	Schmid(2006)
Labeled Recall	86.5	86.3
Labeled Precision	86.7	86.9
Labeled F-score	86.6	86.6

Table 2: Labeled bracketing evaluation, Penn Treebank section 23.

	Prepositions on verbs	Prepositions on nouns
Labeled Recall	86.11	85.98
Labeled Precision	86.50	86.3
Labeled F-score	86.31	86.14

Table 3: Labeled bracketing evaluation, for PCFGs with prepositions incorporated in verbal and nominal categories, Penn Treebank section 23.

Since not all features are incorporated in the PCFG, a PCFG parse tree does not reflect a full analysis according to the feature grammar. However, as a result of the alignment between the PCFG and the constraint grammar, constraints can be solved in the maximal probability tree identified by the Viterbi algorithm, or in the sequence of N highest-probability trees. This will eliminate trees which are not consistent with the feature grammar, and annotate others with a complete set of feature values. This results in feature trees like 1 and 2 for novel sentences.

4. Re-estimating Lexical Parameters of the PCFG

The PCFG trained over the transformed treebank has parameters related to lexical properties of words such as subcategorization features on verbs, attachment preference of adverbs (sentential, nominal, verbal adverbs), and valence and prepositional preferences of nouns. However, since these parameters are tied to particular words, they are not well estimated in a treebank PCFG. In order to have a large-scale lexicon with an accurate representation of such preferences, it is necessary to learn parameters from data of much larger magnitude than available treebanks. We have experimented with learning these parameters over a large unannotated corpus using an unsupervised training method based on the inside-outside algorithm. The inside-outside algorithm iteratively re-estimates the parameters of a PCFG, given unannotated data. We used a modified version of the inside-outside procedure, in which values of lexical parameters are re-estimated from unannotated data, but values of syntactic parameters originally learnt from the treebank are retained in each iteration.

4.1. Smoothing the treebank model

The initial model used for the re-estimation procedure is a smoothed treebank model. A smoothing scheme is required in order to allocate frequency to combinations of words w and POS tags τ which are not present in the treebank and also to all possible incorporations of a part-of-speech (POS) tag. Otherwise, if the treebank model has zero frequency for some lexical parameter, the inside-outside estimate for that parameter would also be zero, and new lexical entries would never be induced. Given an unsmoothed treebank model t_0, the smoothed treebank model t is obtained as follows. First a POS tagger is run on the unsupervised corpus C, and tokens of words and POS tags are tabulated to obtain a frequency table $g(w, \tau)$. Each frequency $g(w, \tau)$ is split among possible incorporations ι in proportion to a ratio of marginal frequencies in t_0, as in equation 1. w is the word, τ is the part-of-speech tag, and ι is the sequence of

incorporated features on the tag.

$$g(w,\tau,\iota) = \frac{t_0(\tau,\iota)}{t_0(\tau)} g(w,\tau) \quad (1)$$

Then the smoothed model t is defined as an interpolation of g and t_0 for lexical parameters as shown in 2, with syntactic parameters copied from t_0.

$$t(w,\tau,\iota) = (1-\lambda_{\tau,\iota}) t_0(w,\tau,\iota) + \lambda_{\tau,\iota} g(w,\tau,\iota) \quad (2)$$

4.2. Re-estimation using Inside-Outside

Starting with the smoothed treebank model t and corpus C, two procedures are carried out. The first procedure is the standard iterative inside-outside procedure (Lari and Young, 1990). The second procedure has a frequency transformation step interleaved between the inside-outside iterations. In this transformation, lexical parameters from the re-estimated model and the original treebank model are linearly combined to give a transformed lexicon used in the next iteration. The values of syntactic parameters used in each iteration are taken directly from the treebank model. The lexical transformation is expressed in Equation 3, where d_i refer to the transformed model for iteration i.

$$d_i(w,\tau,\iota) = (1-\lambda_{\tau,\iota}) t(w,\tau,\iota) + \lambda_{\tau,\iota} \bar{c_i}(w,\tau,\iota) \quad (3)$$

t is the smoothed treebank lexical model and c refers to models estimated from the corpus C. The term $\bar{c_i}(w,\tau,\iota)$ is obtained by scaling the corpus frequencies in $c_i(w,\tau,\iota)$ as in 4.

$$\bar{c_i}(w,\tau,\iota) = \frac{t(\tau,\iota)}{c_i(\tau,\iota)} c_i(w,\tau,\iota). \quad (4)$$

$\lambda_{\tau,\iota}$ is a parameter with $0 < \lambda_{\tau,\iota} < 1$ which may depend on the tag and incorporation. The transformation preserves the marginal tag and incorporation frequencies seen in the treebank model.

5. Experimental setup

The treebank PCFG is trained over sections 0-22 of the transformed Penn Treebank (minus approximately 7000 sentences held out for testing). The corpus used for re-estimation is approximately 4 million words of unannotated Wall Street Journal text (year 1997), with sentence length restricted to less than 25 words. The re-estimation was carried out over a cluster of computers using Bitpar (Schmid, 2004) for inside-outside estimation. The parameter λ in Equation 3 was set to 0.5 for all τ and ι, giving equal weight to the treebank and the re-estimated lexicons. Starting from a smoothed treebank grammar t, we separately ran 6 iterations of the interleaved estimation procedure, and 4 iterations of standard inside-outside estimation. This gave us two series of models corresponding to the two procedures.
We constructed a test set from the Penn treebank to evaluate the learning of the subcategorization frames of novel verbs. First, we selected 117 verbs whose frequency in Treebank sections 0-22 is between 10-20 (mid-frequency verbs). These verbs have appropriately varied subcategorization frames. Prior to building the treebank PCFG, all sentences containing occurrences of these verbs were held out to form Testset I (1331 sentences). The effect of holding out these sentences from the PCFG training data is to make these 117 verbs novel (i.e. unseen in training).

6. Subcategorization Acquisition

We focus on the task of learning the subcategorization frames of verbs using the interleaved inside-outside re-estimation process from §4.2.. The subcategorization frame (SF) of verbs is a parameter of our PCFG – verbal tags in the PCFG are followed by an incorporation sequence that denotes the SF for the verb.
Lexical resources containing accurate and probabilistic verbal subcategorization information are important for various tasks like parsing, machine translation, etc. Creation of a resource containing such information from large corpora has received much attention in the community, with (Brent, 1991), (Ushioda et al., 1993), and (Manning, 1993) being early attempts at extracting frames from raw data. (Briscoe and Carroll, 1997) induce 163 pre-defined frame types, using apriori information about probabilities of particular frame types to filter the induced frames while (Korhonen, 2002) uses Levin classes to get better back-off estimates for hypothesis selection at the filtering stage. An approach similar to ours is used in (Carroll and Rooth, 1998) with a hand-written, head-lexicalised CFG and a raw corpus to iteratively estimate the distribution of subcategorization frames for particular predicates. Schulte im Walde (2002) also uses a head-lexicalised grammar for German to extract distributions for a large number of verbs from a German newspaper corpus. There has also been work to extract formalism-specific lexical resources from treebank data, for example (Chen and Vijay-Shankar, 2000) for LTAG, (Clark et al., 2002) for CCG, (Tsuruoka and Tsujii, 2004) for HPSG.

6.1. Evaluation

In order to evaluate the models obtained from the inside-outside procedures, we focus on the task of detecting the subcategorization frames of *novel* verbs (i.e. verbs that have not been seen in the treebank). We first obtain maximum-probability (viterbi) parses of all sentences in Testset I (described in §5.) using the re-estimated models. All tokens of the test verbs and their pre-terminal symbols are extracted from the viterbi parses. The pre-terminal symbol of verbs consists of a part-of-speech tag and an incorporation sequence encoding the SF. This tag-SF sequence is compared to a gold standard, and is scored correct if the two match exactly. Part-of-speech errors are scored as incorrect, even if the SF is correct. The gold standard is obtained from the transformed Penn-treebank trees. The incorporation sequence on verbs consists of 3 features (together referred to as the subcategorization frame), as described in §2.1.: `Val` (valence), `Vsel` (type of clause for for clausal complements) and `Sbj` (subject of the clausal complements).

Table 4 shows this error rate (i.e. the fraction of test items which receive incorrect tag-incorporations in viterbi parses) for various models obtained using the interleaved and standard re-estimation procedures. t_0 is the model with the test data from Testset I merged in (to account for unknown words) using the same smoothing scheme given in §4.1., with $\lambda = 0.001$. This model has no verb specific information for the test verbs. For each test verb, it has a smoothed SF distribution proportional to the SF distribution for all

Iteration i	Interleaved Procedure	Standard Procedure
t_0	33.36	33.36
1	24.40	28.69
2	23.45	25.56
3	23.05	27.86
4	22.89	28.41
5	22.81	-
6	22.83	-

Table 4: Subcategorization error for novel verbs.

verbs of that tag. The baseline error is about 33.4%. This means that there is enough information in the average distribution of all verbs and in the syntax to correctly assign the subcategorization frame to novel verbs about 66.6% of the time. For the models obtained using the interleaved re-estimation, the error rate falls to the lowest value of 22.81 % for the model obtained in the 5th iteration : an absolute reduction of 10.55 points, and a percentage error-reduction of 31.6%. The models obtained using standard re-estimation do not perform as well.

Amongst previous work on SF acquisition from corpora we find that relatively few parsing-based evaluations are reported. Since their goal is to build probabilistic SF dictionaries, these systems are evaluated either against existing dictionaries, or on distributional similarity measures. Most are evaluated on testsets of *high*-frequency verbs (unlike the present work), in order to gauge the effectiveness of the acquisition strategy. (Briscoe and Carroll, 1997) report a token-based evaluation for seven verb types– their system gets an average recall of 80.9% for these verbs (which appear to be high-frequency verbs). This is slightly lower than the present system (we have an overall accuracy of 83.16% on all verbs (novel and non-novel), evaluated on a separate test set consisting of 4300 sentences held out from the PTB) However, for low frequency verbs (exemplars $<$10) they report that results are around chance. We believe that an evaluation over token occurrences is relevant to NLP tasks. Table 5 shows the development of lexical entries for three representative test verbs in four iterations of the interleaved procedure. The frequencies are scaled according to the formulas in §4.2.; only the top five SFs are shown. Absolute frequencies in the unsupervised training sample are higher. The first column is the smoothed treebank model with an average distribution for these novel verbs. The column for the model from the 4th iteration can be compared to the last column, which shows lexical entries obtained from a treebank model which included these verbs (scaled by 0.5, since λ in eq.3 is 0.5).

7. Conclusions

We have presented a framework that allows for augmentation of a treebank with linguistically motivated features which also allows the building of a PCFG that can be further used in applications for learning of lexical information. The framework can be applied to languages with existing treebanks in order to obtain treebank-aligned resources and to bootstrap induction of lexical information from unannotated data. We plan to use the framework to learn other lexically dependent parameters such as the prepositional attachment preference of verbs and nouns, attachment preference (sentential, nominal, verbal) of adverbs, valence of nouns, etc. in order to create probabilistic lexicons useful for parsing where this type of information about lexical items is represented.

8. Distribution

The programs used to build the augmented treebank and the treebank-aligned PCFG from the Penn treebank files, and some of the resulting resources, are being distributed in a release with the following functionality and/or components. The software environment and/or additional required software or databases are listed in parentheses.

1. Regularize the treebank. (lisp, awk, PTBII mrg files)
2. Build feature constraint grammar from the output of 1. (perl, lisp)
3. Map each regularized treebank tree `ti.tb` to a trivial context free shared forest `ti.cpf` representing one tree. (lisp)
4. Solve feature constraints in each context-free shared forest `ti.cpf` to produce a feature shared forest `ti.fpf`. (yap-compiler, yap-parser, (Schmid, 2000))
5. Map feature shared forests to PCFG rules and lexical entries with incorporated features. Parameter files for several choices of incorporations are included. (yappf-fun (Privman, 2003), java)
6. Adapt PCFG lexicon to a test corpus by tagging the test corpus and smoothing the PCFG lexicon to include the word forms in the test corpus ((Schmid, 1994), perl).
7. PCFG Viterbi parsing with labeled bracket and valence evaluation (bitpar (Schmid, 2004), evalb, perl)
8. Lexicon smoothing for modified inside-outside procedure and re-estimation on unsupervised training corpus (perl, bitpar).
9. Constraint grammar files that are the output of 3 for trees in Treebank II sections 0-15 whose index modulo 10 is not 9.
10. PCFG grammar and lexicon files with incorporated features for parsing sentences in PTB sections 0-22 with index 9 modulo 10.
11. Smoothed PCFG lexicon file with incorporated features for parsing 4 million word WSJ corpus.
12. Re-estimated PCFG lexicon with incorporated features for 4 million word WSJ corpus.

The experiment is organized with a make file which allows the experiment to be built from the treebank distribution. The modular components can be used in ways other

	t	It 1	It 2	It 3	It4	PTB
disagree	VBP.n.-.- 0.0014	VBP.z.-.- 2.01	VBP.z.-.- 2.20	VBP.z.-.- 2.22	VBP.z.-.- 2.23	VBP.z.-.- 1.0
	VBP.t.-.- 0.0012	VBP.p.-.- 0.98	VBP.p.-.- 1.17	VBP.p.-.- 1.20	VBP.p.-.- 1.22	VBP.p.-.- 1.0
	VB.n.-.- 0.0011	VB.p.-.- 0.64	VB.z.-.- 0.60	VB.z.-.- 0.61	VB.z.-.- 0.61	VB.z.-.- 1.0
	VBP.aux.-.h 0.0009	VB.z.-.- 0.56	VB.p.-.- 0.54	VB.p.-.- 0.53	VB.p.-.- 0.53	VB.b.-.- 1.0
	VBP.b.-.- 0.0008	VBP.n.-.- 0.27	VBP.n.-.- 0.27	VBP.n.-.- 0.25	VBP.n.-.- 0.24	-
admit	VB.n.-.- 0.0040	VB.n.-.- 2.06	VB.n.-.- 2.12	VB.n.-.- 2.13	VB.n.-.- 2.16	VB.n.-.- 0.5
	VB.z.-.- 0.0009	VBP.b.-.- 1.27	VBP.b.-.- 1.49	VBP.b.-.- 1.48	VBP.b.-.- 1.48	VBP.p.-.- 0.5
	VBP.n.-.- 0.0008	VB.b.-.- 0.63	VB.b.-.- 0.99	VB.p.-.- 0.32	VB.b.-.- 0.76	VB.z.-.- 0.5
	VBP.t.-.- 0.0007	VBP.n.-.- 0.44	VB.p.-.- 0.32	VBP.n.-.- 0.31	VB.z.-.- 0.33	VBP.z.-.- 0.5
	VB.t.-.- 0.0006	VB.z.-.- 0.33	VBP.n.-.- 0.32	VB.z.-.- 0.30	VB.p.-.- 0.32	-
decides	VBZ.t.-.- 0.0014	VBZ.b.-.- 1.28	VBZ.s.e.to 1.14	VBZ.s.e.to 1.16	VBZ.s.e.to 1.16	VBZ.s.e.to 3.5
	VBZ.n.-.- 0.0011	VBZ.s.e.to 0.90	VBZ.b.-.- 1.09	VBZ.b.-.- 1.04	VBZ.b.-.- 1.06	VBZ.b.-.- 1.5
	VBZ.aux.-.h 0.0008	VBZ.z.-.- 0.63	VBZ.n.-.- 0.37	VBZ.n.-.- 0.36	VBZ.n.-.- 0.36	VBZ.n.-.- 0.5
	VBZ.b.-.- 0.0006	VBZ.s.-.fin 0.42	VBZ.z.-.- 0.32	VBZ.z.-.- 0.35	VBZ.z.-.- 0.35	VBZ.p.-.- 0.5
	VBZ.z.-.- 0.0005	VBZ.n.-.- 0.36	VBZ.p.-.- 0.29	VBZ.p.-.- 0.28	VBZ.p.-.- 0.28	-

Table 5: Lexical entries (top 5 SFs) for three novel test verbs in successive iterations. The frequencies are scaled. The last column shows the distribution of these verbs in a treebank model where they were not held-out. The verb tags are VB (base), VBP (non-3rd-person present tense) and VBZ (3rd person present tense). Interpretation of valences (those in the column for the fourth iteration) are b (that-clause), n (transitive), p (prepositional), s.e.to (control) and z (intransitive).

than the ones discussed here. For instance, feature constraints can be solved in the maximal probability tree resulting from Viterbi parsing or each each of the N trees with highest PCFG probability. This allows feature trees to be constructed for novel data.

Acknowledgements

Thanks to Helmut Schmid for providing distributions and source code for the context free and feature constraint parsers, and for responding to numerous requests.

9. References

F. Beil, G. Carroll, D. Prescher, S. Riezler, and M. Rooth. 1999. Inside-outside estimation of a lexicalized pcfg for german. In *ACL 37*.

M. Brent. 1991. Automatic acquisition of subcategorization frames from untagged text. In *ACL 29*.

Ted Briscoe and John Carroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied NLP*.

John Chen and K. Vijay-Shankar. 2000. Automatic extraction of tags from the penn treebank. In *Proceedings of the 6th International Workshop on Parsing Technologies*, Trento, Italy.

Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building deep dependency structures using a wide-coverage ccg parser. In *ACL40*, pages 00–00.

Sabine Schulte im Walde. 2002. A subcategorisation lexicon for german verbs induced from a lexicalised pcfg. In *LREC 2002*.

Mark Johnson. 1998. Pcfg models of linguistic tree representations. *Computational Linguistics*, 24(4).

Dan Klein and Christopher Manning. 2003. Accurate unlexicalized parsing. In *ACL 41*, Sapporo, Japan.

Anna Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, Univ. of Cambridge.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

C. Manning. 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In *ACL 31*.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

R. O'Donovan, M. Burke, A. Cahill, J. van Genabith, and A. Way. 2005. Large-scale induction and evaluation of lexical resources from the penn-ii and penn-iii treebanks. *Computational Linguistics*, 31:329–365.

Lior Privman. 2003. Yappffun: Java implemention of shared forest algorithms. Computational Linguistics Lab, Cornell University.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing.*

Helmut Schmid. 2000. *YAP - Parsing and Disambiguation With Feature-Based Grammars.* Ph.D. thesis, University of Stuttgart.

Helmut Schmid. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *COLING 2004*.

Yusuke Miyao Tsuruoka, Yoshimasa and Jun'ichi Tsujii. 2004. Towards efficient probabilistic hpsg parsing: integrating semantic and syntactic preference to guide the parsing. In *Proceedings of IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses*, Hainan Island, China.

A. Ushioda, D. Evans, T. Gibson, and A. Waibel. 1993. The automatic acquisition of frequencies of verb subcategorization frames from tagged corpora. In *SIGLEX ACL Workshop on the Acquisition of Lexical Knowledge from Text*.