Ad Hoc Treebank Structures

Markus Dickinson
Department of Linguistics
Indiana University
md7@indiana.edu

Abstract

We outline the problem of ad hoc rules in treebanks, rules used for specific constructions in one data set and unlikely to be used again. These include ungeneralizable rules, erroneous rules, rules for ungrammatical text, and rules which are not consistent with the rest of the annotation scheme. Based on a simple notion of rule equivalence and on the idea of finding rules unlike any others, we develop two methods for detecting ad hoc rules in flat treebanks and show they are successful in detecting such rules. This is done by examining evidence across the grammar and without making any reference to context.

1 Introduction and Motivation

When extracting rules from constituency-based treebanks employing flat structures, grammars often limit the set of rules (e.g., Charniak, 1996), due to the large number of rules (Krotov et al., 1998) and "leaky" rules that can lead to mis-analysis (Foth and Menzel, 2006). Although frequency-based criteria are often used, these are not without problems because low-frequency rules can be valid and potentially useful rules (see, e.g., Daelemans et al., 1999), and high-frequency rules can be erroneous (see., e.g., Dickinson and Meurers, 2005). A key issue in determining the rule set is rule generalizability: will these rules be needed to analyze new data? This issue is of even more importance when considering the task of porting a parser trained on one genre to another genre (e.g., Gildea, 2001). Infrequent rules in one genre may be quite frequent in another (Sekine, 1997) and their frequency may be unrelated to their usefulness for parsing (Foth and Menzel, 2006). Thus, we need to carefully consider the applicability of rules in a treebank to new text.

Specifically, we need to examine *ad hoc* rules, rules used for particular constructions specific to one data set and unlikely to be used on new data. This is why low-frequency rules often do not extend to new data: if they were only used once, it was likely for a specific reason, not something we would expect to see again. Ungeneralizable rules, however, do not extend to new text for a variety of reasons, not all of which can be captured strictly by frequency.

While there are simply phenomena which, for various reasons, are rarely used (e.g., long coordinated lists), other ungeneralizable phenomena are potentially more troubling. For example, when ungrammatical or non-standard text is used, treebanks employ rules to cover it, but do not usually indicate ungrammaticality in the annotation. These rules are only to be used in certain situations, e.g., for typographical conventions such as footnotes, and the fact that the situation is irregular would be useful to know if the purpose of an induced grammar is to support robust parsing. And these rules are outright damaging if the set of treebank rules is intended to accurately capture the grammar of a language. This is true of precision grammars, where analyses can be more or less preferred (see, e.g., Wagner et al., 2007), and in applications like intelligent computer-aided language learning, where learner input is parsed to detect what is correct or not (see, e.g., Vandeventer Faltin, 2003, ch. 2). If a treebank grammar is used (e.g., Metcalf and Boyd, 2006), then one needs to isolate rules for ungrammatical data, to be able to distinguish grammatical from ungrammatical input.

Detecting ad hoc rules can also reveal issues related to rule quality. Many ad hoc rules exist because they are erroneous. Not only are errors inherently undesirable for obtaining an accurate grammar, but training on data with erroneous rules can be detrimental to parsing performance (e.g., Dickinson and Meurers, 2005; Hogan, 2007) As annotation schemes are not guaranteed to be completely consistent, other ad hoc rules point to non-uniform aspects of the annotation scheme. Thus, identifying ad hoc rules can also provide feedback on annotation schemes, an especially important step if one is to use the treebank for specific applications (see, e.g., Vadas and Curran, 2007), or if one is in the process of developing a treebank.

Although statistical techniques have been employed to detect anomalous annotation (Ule and Simov, 2004; Eskin, 2000), these methods do not account for linguistically-motivated generalizations across rules, and no full evaluation has been done on a treebank. Our starting point for detecting ad hoc rules is also that they are dissimilar to the rest of the grammar, but we rely on a notion of equivalence which accounts for linguistic generalizations, as described in section 2. We generalize equivalence in a corpus-independent way in section 3 to detect ad hoc rules, using two different methods to determine when rules are dissimilar. The results in section 4 show the success of the method in identifying all types of ad hoc rules.

2 Background

2.1 Equivalence classes

To define dissimilarity, we need a notion of similarity, and, a starting point for this is the error detection method outlined in Dickinson and Meurers (2005). Since most natural language expressions are endocentric, i.e., a category projects to a phrase of the same category (e.g., X-bar Schema, Jackendoff, 1977), daughters lists with more than one possible mother are flagged as potentially containing an error. For example, IN NP[1] has nine different mothers in the Wall Street Journal (WSJ) portion of the Penn Treebank (Marcus et al., 1993), six of which are errors.

This method can be extended to increase recall, by treating similar daughters lists as equivalent (Dickinson, 2006, 2008). For example, the daughters lists ADVP RB ADVP and ADVP , RB ADVP in (1) can be put into the same equivalence class, because they predict the same mother category. With this equivalence, the two different mothers, PP and ADVP, point to an error (in PP).

(1) a. to slash its work force in the U.S. , [$_{PP}$ [$_{ADVP}$ as] soon/RB [$_{ADVP}$ as next month]]
 b. to report ... [$_{ADVP}$ [$_{ADVP}$ immediately] ,/, not/RB [$_{ADVP}$ a month later]]

Anything not contributing to predicting the mother is ignored in order to form equivalence classes. Following the steps below, 15,989 daughters lists are grouped into 3783 classes in the WSJ.

1. Remove daughter categories that are always non-predictive to phrase categorization, i.e., always adjuncts, such as punctuation and the parenthetical (PRN) category.
2. Group head-equivalent lexical categories, e.g., NN (common noun) and NNS (plural noun).
3. Model adjacent identical elements as a single element, e.g., NN NN becomes NN.

While the sets of non-predictive and head-equivalent categories are treebank-specific, they require only a small amount of manual effort.

2.2 Non-equivalence classes

Rules in the same equivalence class not only predict the same mother, they provide support that the daughters list is accurate—the more rules within a class, the better evidence that the annotation scheme legitimately licenses that sequence. A lack of similar rules indicates a potentially anomalous structure.

Of the 3783 equivalence classes for the whole WSJ, 2141 are unique, i.e., have only one unique daughters list. For example, in (2), the daughters list RB TO JJ NNS is a daughters list with no correlates in the treebank; it is erroneous because *close to wholesale* needs another layer of structure, namely adjective phrase (ADJP) (Bies et al., 1995, p. 179).

[1] Appendix A lists all categories used in this paper.

(2) they sell [merchandise] for $[_{NP}$ close/RB to/TO wholesale/JJ prices/NNS]

Using this strict equivalence to identify ad hoc rules is quite successful (Dickinson, 2008), but it misses a significant number of generalizations. These equivalences were not designed to assist in determining linguistic patterns from non-linguistic patterns, but to predict the mother category, and thus many correct rules are incorrectly flagged. To provide support for the correct rule NP → DT CD JJS NNP JJ NNS in (3), for instance, we need to look at some highly similar rules in the treebank, e.g., the three instances of NP → DT CD JJ NNP NNS, which are not strictly equivalent to the rule in (3).

(3) $[_{NP}$ the/DT 100/CD largest/JJS Nasdaq/NNP financial/JJ stocks/NNS]

3 Rule dissimilarity and generalizability

3.1 Criteria for rule equivalence

With a notion of (non-)equivalence as a heuristic, we can begin to detect ad hoc rules. First, however, we need to redefine equivalence to better reflect syntactic patterns.

Firstly, in order for two rules to be in the same equivalence class—or even to be similar—the mother must also be the same. This captures the property that identical daughters lists with different mothers are distinct (cf. Dickinson and Meurers, 2005). For example, looking back at (1), the one occurrence of ADVP → ADVP , RB ADVP is very similar to the 4 instances of ADVP → RB ADVP, whereas the one instance of PP → ADVP RB ADVP is not and is erroneous. Daughters lists are thus now only compared to rules with the same mother.

Secondly, we use only two steps to determine equivalence: 1) remove non-predictive daughter categories, and 2) group head-equivalent lexical categories.[2] While useful for predicting the same mother, the step of Kleene reduction is less useful for our purposes since it ignores potential differences in argument structure. It is important to know how many identical categories can appear within a given rule, to tell whether it is reliable; VP → VB NP and VP → VB NP NP, for example, are two different rules.[3]

Thirdly, we base our scores on token counts, in order to capture the fact that the more often we observe a rule, the more reliable it seems to be. This is not entirely true, as mentioned above, but this prevents frequent rules such as NP → EX (1075 occurrences) from being seen as an anomaly.

With this new notion of equivalence, we can now proceed to accounting for similar rules in detecting ad hoc rules.

3.2 Reliability scores

In order to devise a scoring method to reflect similar rules, the simplest way is to use a version of edit distance between rules, as we do under the *Whole daughters scoring* below. This reflects the intuition that rules with similar lists of daughters reflect the same properties. This is the "positive" way of scoring rules, in that we start with a basic notion of equivalence and look for more positive evidence that the rule is legitimate. Rules without such evidence are likely ad hoc.

Our goal, though, is to take the results and examine the anomalous rules, i.e., those which lack strong evidence from other rules. We can thus more directly look for "negative" evidence that a rule is ad hoc. To do this, we can examine the weakest parts of each rule and compare those across the corpus, to see which anomalous patterns emerge; we do this in the *Bigram scoring* section below.

Because these methods exploit different properties of rules and use different levels of abstraction, they have complementary aspects. Both start with the same assumptions about what makes rules equivalent, but diverge in how they look for rules which do not fit well into these equivalences.

Whole daughters scoring The first method to detect ad hoc rules directly accounts for similar rules across equivalence classes. Each rule type is assigned a reliability score, calculated as follows:

1. Map a rule to its equivalence class.

2. For every rule token within the equivalence class, add a score of 1.

[2] See Dickinson (2006) for the full mappings.

[3] Experiments done with Kleene reduction show that the results are indeed worse.

3. For every rule token within a highly similar equivalence class, add a score of $\frac{1}{2}$.

Positive evidence that a rule is legitimate is obtained by looking at similar classes in step #3, and then rules with the lowest scores are flagged as potentially ad hoc (see section 4.1). To determine similarity, we use a modified Levenshtein distance, where only insertions and deletions are allowed; a distance of one qualifies as *highly similar*.[4] Allowing two or more changes would be problematic for unary rules (e.g., (4a), and in general, would allow us to add and subtract dissimilar categories. We thus remain conservative in determining similarity.

Also, we do not utilize substitutions: while they might be useful in some cases, it is too problematic to include them, given the difference in meaning of each category. Consider the problematic rules in (4). In (4a), which occurs once, if we allow substitutions, then we will find 760 "comparable" instances of VP → VB, despite the vast difference in category (verb vs. adverb). Likewise, the rule in (4b), which occurs 8 times, would be "comparable" to the 602 instances of PP → IN PP, used for multi-word prepositions like *because of*.[5] To maintain these true differences, substitutions are not allowed.

(4) a. VP → RB
    b. PP → JJ PP

This notion of similarity captures many generalizations, e.g., that adverbial phrases are optional. For example, in (5), the rule reduces to S → PP ADVP NP ADVP VP. With a strict notion of equivalence, there are no comparable rules. However, the class S → PP NP ADVP VP, with 198 members, is highly similar, indicating more confidence in this correct rule.

(5) [$_S$ [$_{PP}$ During his years in Chiriqui] ,/, [$_{ADVP}$ however] ,/, [$_{NP}$ Mr. Noriega] [$_{ADVP}$ also] [$_{VP}$ revealed himself as an officer as perverse as he was ingenious] ./.]

Bigram scoring The other method of detecting ad hoc rules calculates reliability scores by focusing specifically on what the classes do not have in common. Instead of examining and comparing rules in their entirety, this method abstracts a rule to its component parts, similar to features using information about n-grams of daughter nodes in parse reranking models (e.g., Collins and Koo, 2005).

We abstract to bigrams, including added *START* and *END* tags, as longer sequences risk missing generalizations; e.g., unary rules would have no comparable rules. We score rule types as follows:

1. Map a rule to its equivalence class, resulting in a *reduced rule*.
2. Calculate the frequency of each <mother,bigram> pair in a reduced rule: for every reduced rule token with the same pair, add a score of 1 for that bigram pair.
3. Assign the score of the least-frequent bigram as the score of the rule.

We assign the score of the lowest-scoring bigram because we are interested in anomalous sequences. This is in the spirit of Květon and Oliva (2002), who define invalid bigrams for POS annotation sequences in order to detect annotation errors..

As one example, consider (6), where the reduced rule NP → NP DT NNP is composed of the bigrams START NP, NP DT, DT NNP, and NNP END. All of these are relatively common (more than a hundred occurrences each), except for NP DT, which appears in only two other rule types. Indeed, DT is an incorrect tag (NNP is correct): when NP is the first daughter of NP, it is generally a possessive, precluding the use of a determiner.

(6) (NP (NP ABC 's) (`` ``) (DT This) (NNP Week))

The whole daughters scoring misses such problematic structures because it does not explicitly look for anomalies. The disadvantage of the bigram scoring, however, is its missing of the big picture: for example, the erroneous rule NP → NNP CC NP gets a large score (1905) because each subsequence is quite common. But this exact sequence is rather rare (NNP and NP are not generally coordinated), so the whole daughters scoring assigns a low score (4.0).

[4] The score is thus more generally $\frac{1}{1+distance}$, although we ascribe no theoretical meaning to this

[5] Rules like PP → JJ PP might seem to be correct, but this depends upon the annotation scheme. Phrases starting with *due to* are sometimes annotated with this rule, but they also occur as ADJP or ADVP or with *due* as RB. If PP → JJ PP is correct, identifying this rule actually points to other erroneous rules.

4 Evaluation

To gauge our success in detecting ad hoc rules, we evaluate the reliability scores in two main ways: 1) whether unreliable rules generalize to new data (section 4.1), and, more importantly, 2) whether the unreliable rules which do generalize are ad hoc in other ways—e.g., erroneous (section 4.2). To measure this, we use sections 02-21 of the WSJ corpus as training data to derive scores, section 23 as testing data, and section 24 as development data.

4.1 Ungeneralizable rules

To compare the effectiveness of the two scoring methods in identifying ungeneralizable rules, we examine how many rules from the training data do not appear in the heldout data, for different thresholds. In figure 1, for example, the method identifies 3548 rules with scores less than or equal to 50, 3439 of which do not appear in the development data, resulting in an ungeneralizability rate of 96.93%.

To interpret the figures below, we first need to know that of the 15,246 rules from the training data, 1832 occur in the development data, or only 12.02%, corresponding to 27,038 rule tokens. There are also 396 new rules in the development data, making for a total of 2228 rule types and 27,455 rule tokens.

4.1.1 Development data results

The results are shown in figure 1 for the whole daughters scoring method and in figure 2 for the bigram method. Both methods successfully identify rules with little chance of occurring in new data, the whole daughters method performing slightly better.

Thresh.	Rules	Unused	Ungen.
1	311	311	100.00%
25	2683	2616	97.50%
50	3548	3439	96.93%
100	4596	4419	96.15%

Figure 1: Whole daughter ungeneralizability (devo.)

Thresh.	Rules	Unused	Ungen.
1	599	592	98.83%
5	1661	1628	98.01%
10	2349	2289	97.44%
15	2749	2657	96.65%
20	3120	2997	96.06%

Figure 2: Bigram ungeneralizability (devo.)

4.1.2 Comparing across data

Is this ungeneralizability consistent over different data sets? To evaluate this, we use the whole daughters scoring method, since it had a higher ungeneralizability rate in the development data, and we use section 23 of the WSJ and the Brown corpus portion of the Penn Treebank.

Given different data sizes, we now report the coverage of rules in the heldout data, for both type and token counts. For instance, in figure 3, for a threshold of 50, 108 rule types appear in the development data, and they appear 141 times. With 2228 total rule types and 27,455 rule tokens, this results in coverages of 4.85% and 0.51%, respectively.

In figures 3, 4, and 5, we observe the same trends for all data sets: low-scoring rules have little generalizability to new data. For a cutoff of 50, for example, rules at or below this mark account for approximately 5% of the rule types used in the data and half a percent of the tokens.

	Types		Tokens	
Thresh.	Used	Cov.	Used	Cov.
10	23	1.03%	25	0.09%
25	67	3.01%	78	0.28%
50	108	4.85%	141	0.51%
100	177	7.94%	263	0.96%
All	1832	82.22%	27,038	98.48%

Figure 3: Coverage of rules in WSJ, section 24

	Types		Tokens	
Thresh.	Used	Cov.	Used	Cov.
10	33	1.17%	39	0.08%
25	82	2.90%	117	0.25%
50	155	5.49%	241	0.51%
100	242	8.57%	416	0.88%
All	2266	80.24%	46,375	98.74%

Figure 4: Coverage of rules in WSJ, section 23

Note in the results for the larger Brown corpus that the percentage of overall rule types from the

	Types		Tokens	
Thresh.	Used	Cov.	Used	Cov.
10	187	1.51%	603	0.15%
25	402	3.25%	1838	0.45%
50	562	4.54%	2628	0.64%
100	778	6.28%	5355	1.30%
All	4675	37.75%	398,136	96.77%

Figure 5: Coverage of rules in Brown corpus

training data is only 37.75%, vastly smaller than the approximately 80% from either WSJ data set. This illustrates the variety of the grammar needed to parse this data versus the grammar used in training.

We have isolated thousands of rules with little chance of being observed in the evaluation data, and, as we will see in the next section, many of the rules which appear are problematic in other ways. The ungeneralizabilty results make sense, in light of the fact that reliability scores are based on token counts. Using reliability scores, however, has the advantage of being able to identify infrequent but correct rules (cf. example (5)) and also frequent but unhelpful rules. For example, in (7), we find erroneous cases from the development data of the rules WHNP → WHNP WHPP (*five* should be NP) and VP → NNP NP (*OKing* should be VBG). These rules appear 27 and 16 times, respectively, but have scores of only 28.0 and 30.5, showing their unreliability. Future work can separate the effect of frequency from the effect of similarity (see also section 4.3).

(7) a. [$_{WHNP}$ [$_{WHNP}$ five] [$_{WHPP}$ of whom]]

b. received hefty sums for * [$_{VP}$ OKing/NNP [$_{NP}$ the purchase of ...]]

4.2 Other ad hoc rules

The results in section 4.1 are perhaps unsuprising, given that many of the identified rules are simply rare. What is important, therefore, is to figure out why some rules appeared in the heldout data at all. As this requires qualitative analysis, we hand-examined the rules appearing in the development data. We set out to examine about 100 rules, and so we report only for the corresponding threshold, finding that ad hoc rules are predominant.

For the whole daughters scoring, at the 50 threshold, 55 (50.93%) of the 108 rules in the development data are errors. Adding these to the ungeneralizable rules, 98.48% (3494/3548) of the 3548 rules are unhelpful for parsing, at least for this data set. An additional 12 rules cover non-English or fragmented constructions, making for 67 clearly ad hoc rules.

For the bigram scoring, at the 20 threshold, 67 (54.47%) of the 123 rules in the development data are erroneous, and 8 more are ungrammatical. This means that 97.88% (3054/3120) of the rules at this threshold are unhelpful for parsing this data, still slightly lower than the whole daughters scoring.

4.2.1 Problematic cases

But what about the remaining rules for both methods which are not erroneous or ungrammatical? First, as mentioned at the outset, there are several cases which reveal non-uniformity in the annotation scheme or guidelines. This may be justifiable, but it has an impact on grammars using the annotation scheme. Consider the case of NAC (not a constituent), used for complex NP premodifiers. The description for tagging titles in the guidelines (Bies et al., 1995, p. 208-209) covers the exact case found in section 24, shown in (8a). This rule, NAC → NP PP, is one of the lowest-scoring rules which occurs, with a whole daughters score of 2.5 and a bigram score of 3, yet it is correct. Examining the guidelines more closely, however, we find examples such as (8b). Here, no extra NP layer is added, and it is not immediately clear what the criteria are for having an intermediate NP.

(8) a. a " [$_{NAC}$ [$_{NP}$ Points] [$_{PP}$ of Light]] " foundation

b. The Wall Street Journal " [$_{NAC}$ American Way [$_{PP}$ of Buying]] " Survey

Secondly, rules with mothers which are simply rare are prone to receive lower scores, regardless of their generalizability. For example, the rules dominated by SINV, SQ, or SBARQ are all correct (6 in whole daughters, 5 in bigram), but questions are not very frequent in this news text: SQ appears only 350 times and SBARQ 222 times in the training data. One might thus consider normalizing the scores based on the overall frequency of the parent.

Finally, and most prominently, there are issues with coordinate structures. For example, NP → NN CC DT receives a low whole daughters score of 7.0, despite the fact that NP → NN and NP → DT are very common rules. This is a problem for both methods: for the whole daughters scoring, of the 108, 28 of them had a conjunct (CC or CONJP) in the daughters list, and 18 of these were correct. Likewise, for the bigram scoring, 18 had a conjunct, and 12 were correct. Reworking similarity scores to reflect coordinate structures and handle each case separately would require treebank-specific knowledge: the Penn Treebank, for instance, distinguishes unlike coordinated phrases (UCP) from other coordinated phrases, each behaving differently.

4.2.2 Comparing the methods

There are other cases in which one method outperforms the other, highlighting their strengths and weaknesses. In general, both methods fare badly with clausal rules, i.e., those dominated by S, SBAR, SINV, SQ, or SBARQ, but the effect is slightly greater on the bigram scoring, where 20 of the 123 rules are clausal, and 16 of these are correct (i.e., 80% of them are misclassified). To understand this, we have to realize that most modifiers are adjoined at the sentence level when there is any doubt about their attachment (Bies et al., 1995, p. 13), leading to correct but rare subsequences. In sentence (9), for example, the reduced rule S → SBAR PP NP VP arises because both the introductory SBAR and the PP are at the same level. This SBAR PP sequence is fairly rare, resulting in a bigram score of 13.

(9) [$_{S}$ [$_{SBAR}$ As the best opportunities for corporate restructurings are exhausted * of course] ,/, [$_{PP}$ at some point] [$_{NP}$ the market] [$_{VP}$ will start * to reject them] ./.]

Whole daughters scoring, on the other hand, assigns this rule a high reliability score of 2775.0, due to the fact that both SBAR NP VP and PP NP VP sequences are common. For rules with long modifier sequences, whole daughters scoring seems to be more effective since modifiers are easily skipped over in comparing to other rules. Whole daughters scoring is also imprecise with clausal rules (10/12 are misclassified), but identifies less of them, and they tend to be for rare mothers (see above).

Various cases are worse for the whole daughters scoring. First are quantifier phrases (QPs), which have a highly varied set of possible heads and arguments. QP is "used for multiword numerical expressions that occur within NP (and sometimes ADJP), where the QP corresponds frequently to some kind of complex determiner phrase" (Bies et al., 1995, p. 193). This definition leads to rules which look different from QP to QP. Some of the lowest-scoring, correct rules are shown in (10). We can see that there is not a great deal of commonality about what comprises quantifier phrases, even if subparts are common and thus not flagged by the bigram method.

(10) a. [$_{QP}$ only/RB three/CD of/IN the/DT nine/CD] justices

b. [$_{QP}$ too/RB many/JJ] cooks

c. 10 % [$_{QP}$ or/CC more/JJR]

Secondly, whole daughters scoring relies on complete sequences, and thus whether Kleene reduction (step #3 in section 2) is used makes a marked difference. For example, in (11), the rule NP → DT JJ NNP NNP JJ NN NN is completely correct, despite its low whole daughters score of 15.5 and one occurrence. This rule is similar to the 10 occurrences of NP → DT JJ NNP JJ NN in the training set, but we cannot see this without performing Kleene reduction. For noun phrases at least, using Kleene reduction might more accurately capture comparability. This is less of an issue for bigram scoring, as all the bigrams are perfectly valid, resulting here in a relatively high score (556).

(11) [$_{NP}$ the/DT basic/JJ Macintosh/NNP Plus/NNP central/JJ processing/NN unit/NN]

4.3 Discriminating rare rules

In an effort to determine the effectiveness of the scores on isolating structures which are not linguistically sound, in a way which factors out frequency, we sampled 50 rules occurring only once in the training data. We marked for each whether it was correct or how it was ad hoc, and we did this blindly, i.e., without knowledge of the rule scores. Of these 50, only 9 are errors, 2 cover ungrammatical constructions, and 8 more are unclear. Looking at the bottom 25 scores, we find that the whole daughters and bigrams methods both find 6 errors, or 67% of them, additionally finding 5 unclear cases for the whole daughters and 6 for the bigrams method. Erroneous rules in the top half appear to be ones which happened to be errors, but could actually be correct in other contexts (e.g.,NP → NN NNP NNP CD). Although it is a small data set, the scores seem to be effectively sorting rare rules.

5 Summary and Outlook

We have outlined the problem of ad hoc rules in treebanks—ungeneralizable rules, erroneous rules, rules for ungrammatical text, and rules which are not necessarily consistent with the rest of the annotation scheme. Based on the idea of finding rules unlike any others, we have developed methods for detecting ad hoc rules in flat treebanks, simply by examining properties across the grammar and without making any reference to context.

We have been careful not to say how to use the reliability scores. First, without 100% accuracy, it is hard to know what their removal from a parsing model would mean. Secondly, assigning confidence scores to rules, as we have done, has a number of other potential applications. Parse reranking techniques, for instance, rely on knowledge about features other than those found in the core parsing model in order to determine the best parse (e.g., Collins and Koo, 2005; Charniak and Johnson, 2005). Active learning techniques also require a scoring function for parser confidence (e.g., Hwa et al., 2003), and often use uncertainty scores of parse trees in order to select representative samples for learning (e.g., Tang et al., 2002). Both could benefit from more information about rule reliability.

Given the success of the methods, we can strive to make them more corpus-independent, by removing the dependence on equivalence classes. In some ways, comparing rules to similar rules already naturally captures equivalences among rules. In this process, it will also be important to sort out the impact of similarity from the impact of frequency on identifying ad hoc structures.

Acknowledgments

Thanks to the three anonymous reviewers for their helpful comments. This material is based upon work supported by the National Science Foundation under Grant No. IIS-0623837.

A Relevant Penn Treebank categories

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
RB	Adverb
TO	*to*
VB	Verb, base form
VBG	Verb, gerund or present participle

Figure 6: POS tags in the PTB (Santorini, 1990)

ADJP	Adjective Phrase
ADVP	Adverb Phrase
CONJP	Conjunction Phrase
NAC	Not A Constituent
NP	Noun Phrase
PP	Prepositional Phrase
PRN	Parenthetical
QP	Quantifier Phrase
S	Simple declarative clause
SBAR	Clause introduced by subordinating conjunction
SBARQ	Direct question introduced by *wh*-word/phrase
SINV	Inverted declarative sentence
SQ	Inverted yes/no question
UCP	Unlike Coordinated Phrase
VP	Verb Phrase
WHNP	*Wh*-noun Phrase
WHPP	*Wh*-prepositional Phrase

Figure 7: Syntactic categories in the PTB (Bies et al., 1995)

References

Bies, Ann, Mark Ferguson, Karen Katz and Robert MacIntyre (1995). *Bracketing Guidelines for Treebank II Style Penn Treebank Project*. University of Pennsylvania.

Charniak, Eugene (1996). *Tree-Bank Grammars*. Tech. Rep. CS-96-02, Department of Computer Science, Brown University, Providence, RI.

Charniak, Eugene and Mark Johnson (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL-05*. Ann Arbor, MI, USA, pp. 173–180.

Collins, Michael and Terry Koo (2005). Discriminative Reranking for Natural Language Parsing. *Computational Linguistics* 31(1), 25–69.

Daelemans, Walter, Antal van den Bosch and Jakub Zavrel (1999). Forgetting Exceptions is Harmful in Language Learning. *Machine Learning* 34, 11–41.

Dickinson, Markus (2006). Rule Equivalence for Error Detection. In *Proceedings of TLT 2006*. Prague, Czech Republic.

Dickinson, Markus (2008). Similarity and Dissimilarity in Treebank Grammars. In *18th International Congress of Linguists (CIL18)*. Seoul.

Dickinson, Markus and W. Detmar Meurers (2005). Prune Diseased Branches to Get Healthy Trees! How to Find Erroneous Local Trees in a Treebank and Why It Matters. In *Proceedings of TLT 2005*. Barcelona, Spain.

Eskin, Eleazar (2000). Automatic Corpus Correction with Anomaly Detection. In *Proceedings of NAACL-00*. Seattle, Washington, pp. 148–153.

Foth, Kilian and Wolfgang Menzel (2006). Robust Parsing: More with Less. In *Proceedings of the workshop on Robust Methods in Analysis of Natural Language Data (ROMAND 2006)*.

Gildea, Daniel (2001). Corpus Variation and Parser Performance. In *Proceedings of EMNLP-01*. Pittsburgh, PA.

Hogan, Deirdre (2007). Coordinate Noun Phrase Disambiguation in a Generative Parsing Model. In *Proceedings of ACL-07*. Prague, pp. 680–687.

Hwa, Rebecca, Miles Osborne, Anoop Sarkar and Mark Steedman (2003). Corrected Co-training for Statistical Parsers. In *Proceedings of ICML-2003*. Washington, DC.

Jackendoff, Ray (1977). *X' Syntax: A Study of Phrase Structure*. Cambridge, MA: MIT Press.

Krotov, Alexander, Mark Hepple, Robert J. Gaizauskas and Yorick Wilks (1998). Compacting the Penn Treebank Grammar. In *Proceedings of ACL-98*. pp. 699–703.

Květon, Pavel and Karel Oliva (2002). Achieving an Almost Correct PoS-Tagged Corpus. In *Text, Speech and Dialogue (TSD)*. pp. 19–26.

Marcus, M., Beatrice Santorini and M. A. Marcinkiewicz (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2), 313–330.

Metcalf, Vanessa and Adriane Boyd (2006). Head-lexicalized PCFGs for Verb Subcategorization Error Diagnosis in ICALL. In *Workshop on Interfaces of Intelligent Computer-Assisted Language Learning*. Columbus, OH.

Santorini, Beatrice (1990). *Part-Of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd printing)*. Tech. Rep. MS-CIS-90-47, The University of Pennsylvania, Philadelphia, PA.

Sekine, Satoshi (1997). The Domain Dependence of Parsing. In *Proceedings of ANLP-96*. Washington, DC.

Tang, Min, Xiaoqiang Luo and Salim Roukos (2002). Active Learning for Statistical Natural Language Parsing. In *Proceedings of ACL-02*. Philadelphia, pp. 120–127.

Ule, Tylman and Kiril Simov (2004). Unexpected Productions May Well be Errors. In *Proceedings of LREC 2004*. Lisbon, Portugal, pp. 1795–1798.

Vadas, David and James Curran (2007). Adding Noun Phrase Structure to the Penn Treebank. In *Proceedings of ACL-07*. Prague, pp. 240–247.

Vandeventer Faltin, Anne (2003). Syntactic error diagnosis in the context of computer assisted language learning. Thèse de doctorat, Université de Genève, Genève.

Wagner, Joachim, Jennifer Foster and Josef van Genabith (2007). A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors. In *Proceedings of EMNLP-CoNLL 2007*. pp. 112–121.