Adapting a WSJ-Trained Parser to Grammatically Noisy Text

Jennifer Foster, Joachim Wagner and Josef van Genabith
National Centre for Language Technology
Dublin City University
Ireland
jfoster, jwagner, josef@computing.dcu.ie

Abstract

We present a robust parser which is trained on a treebank of ungrammatical sentences. The treebank is created automatically by modifying Penn treebank sentences so that they contain one or more syntactic errors. We evaluate an existing Penn-treebank-trained parser on the ungrammatical treebank to see how it reacts to noise in the form of grammatical errors. We re-train this parser on the training section of the ungrammatical treebank, leading to an significantly improved performance on the ungrammatical test sets. We show how a classifier can be used to prevent performance degradation on the original grammatical data.

1 Introduction

The focus in English parsing research in recent years has moved from Wall Street Journal parsing to improving performance on other domains. Our research aim is to improve parsing performance on text which is mildly ungrammatical, i.e. text which is well-formed enough to be understood by people yet which contains the kind of grammatical errors that are routinely produced by both native and non-native speakers of a language. The intention is not to detect and correct the error, but rather to *ignore* it. Our approach is to introduce grammatical noise into WSJ sentences while retaining as much of the structure of the original trees as possible. These sentences and their associated trees are then used as training material for a statistical parser. It is important that parsing on grammatical sentences is not harmed and we introduce a parse-probability-based classifier which allows both grammatical and ungrammatical sentences to be accurately parsed.

2 Background

Various strategies exist to build robustness into the parsing process: grammar constraints can be relaxed (Fouvry, 2003), partial parses can be concatenated to form a full parse (Penstein Rosé and Lavie, 1997), the input sentence can itself be transformed until a parse can be found (Lee et al., 1995), and mal-rules describing particular error patterns can be included in the grammar (Schneider and McCoy, 1998). For a parser which tends to fail when faced with ungrammatical input, such techniques are needed. The overgeneration associated with a statistical data-driven parser means that it does not typically fail on ungrammatical sentences. However, it is not enough to return some analysis for an ungrammatical sentence. If the syntactic analysis is to guide semantic analysis, it must reflect as closely as possible what the person who produced the sentence was trying to express. Thus, while statistical, data-driven parsing has solved the robustness problem, it is not clear that it is has solved the *accurate* robustness problem.

The problem of adapting parsers to accurately handle ungrammatical text is an instance of the domain adaptation problem where the target domain is grammatically noisy data. A parser can be adapted to a target domain by training it on data from the new domain – the problem is to quickly produce high-quality training material. Our solution is to simply modify the existing training material so that it resembles material from the noisy target domain.

In order to tune a parser to syntactically ill-formed text, a treebank is automatically transformed into an ungrammatical treebank. This transformation process has two parts: 1. the yield of each tree is transformed into an ungrammatical sentence by introducing a syntax error; 2. each tree is minimally transformed, but left intact as much as possible to reflect the syntactic structure of the original “intended” sentence prior to error insertion. Artificial ungrammaticalities have been used in various NLP tasks (Smith and Eisner, 2005; Okanohara and Tsujii, 2007)

The idea of an automatically generated ungrammatical treebank was proposed by Foster (2007). Foster generates an ungrammatical version of the WSJ treebank and uses this to train two statistical parsers. The performance of both parsers significantly improves on the artificially created ungrammatical test data, but significantly degrades on the original grammatical test data. We show that it is possible to obtain significantly improved performance on ungrammatical data without a concomitant performance decline on grammatical data.

3 Generating Noisy Treebanks

Generating Noisy Sentences We apply the error introduction procedure described in detail in Foster (2007). Errors are introduced into sentences by applying the operations of word substitution, deletion and insertion. These operations can be iteratively applied to generate increasingly noisy sentences. We restrict our attention to ungrammatical sentences with a edit-distance of one or two words from the original sentence, because it is reasonable to expect a parser's performance to degrade as the input becomes more ill-formed. The operations of substitution, deletion and insertion are not carried out entirely at random, but are subject to some constraints derived from an empirical study of ill-formed English sentences (Foster, 2005). Three types of word substitution errors are produced: agreement errors, real word spelling errors and verb form errors. Any word that is not an adjective or adverb can be deleted from any position within the input sentence, but some part-of-speech tags are favoured over others, e.g. it is more likely that a determiner will be deleted than a noun. The error creation procedure can insert an arbitrary word at any position within a sentence but it has a bias towards inserting a word directly after the same word or directly after a word with the same part of speech. The empirical study also influences the frequency at which particular errors are introduced, with missing word errors being the most frequent, followed by extra word errors, real word spelling errors, agreement errors, and finally, verb form errors. Table 1 shows examples of the kind of ill-formed sentences that are produced when we apply the procedure to Wall Street Journal sentences.

Generating Trees for Noisy Sentences The tree structures associated with the modified sentences are also modified, but crucially, this modification is minimal, since a truly robust parser should return an analysis for a mildly ungrammatical sentence that remains as similar as possible to the analysis it returns for the original grammatical sentence.

Assume that (1) is an original treebank tree for the sentence *A storm is brewing*. Example (2) is then the tree for the ungrammatical sentence containing an *is/it* confusion. No part of the original tree structure is changed apart from the yield.

(1) *(S (NP A storm) (VP (VBZ is) (VP (VBG brewing))))*
(2) *(S (NP A storm) (VP (VBZ* ***it****) (VP (VBG brewing))))*

An example of a missing word error is shown in (3) and (4). A pre-terminal dominating an empty node is introduced into the tree at the point where the word has been omitted.

(3) *(S (NP Annotators) (VP (VBP parse) (NP the sentences)))*
(4) *(S (NP Annotators) (VP* ***(-NONE- 0)*** *(NP the sentences)))*

An example of an extra word error is shown in (5), (6) and (7). For this example, two ungrammatical trees, (6) and (7), are generated because there are two possible positions in the original tree where the extra word can be inserted which will result in a tree with the yield *He likes of the cake* and which will not result in the creation of any additional structure.

(5) *(S (NP He) (VP (VBZ likes) (NP (DT the) (NN cake))))*
(6) *(S (NP He) (VP (VBZ likes)* ***(IN of)*** *(NP (DT the) (NN cake))))*
(7) *(S (NP He) (VP (VBZ likes) (NP* ***(IN of)*** *(DT the) (NN cake))))*

4 Parser Adaptation Experiments

In order to obtain training data for our parsing experiments, we introduce syntactic noise into the usual WSJ training material, Sections 2-21, using the procedures outlined in Section 3, i.e. for every sentence-tree pair in *WSJ2-21*, we introduce an error into the sentence and then transform the tree so that it covers the newly created ungrammatical sentence. For 4 of the 20 sections in *WSJ2-21*, we apply the noise introduction procedure to its own output to

Error Type	***WSJ00***
Missing Word	*likely to **bring** new attention to the problem → likely to new attention to the problem*
Extra Word	*the $ 5.9 million it posted → the $ 5.9 million **I** it posted*
Real Word Spell	*Mr Vinken **is** chairman of Elsevier → Mr. Vinken **if** chairman of Elsevier*
Agreement	***this** event took place 35 years ago → **these** event took place 35 years ago*
Verb Form	*But the Soviets might still **face** legal obstacles → But the Soviets might still **faces** legal obstacles*

Table 1: Automatically Generated Ungrammatical Sentences

create even noisier data. Our first development set is a noisy version of *WSJ00*, *Noisy00*, produced by applying the noise introduction procedure to the 1,921 sentences in *WSJ00*. Our second development set is an even noisier version of *WSJ00*, *Noisiest00*, which is created by applying our noise introduction procedure to the output of *Noisy00*. We apply the same process to *WSJ23* to obtain our two test sets.

For all our parsing experiments, we use the June 2006 version of the two-stage parser reported in Charniak and Johnson (2005). Evaluation is carried out using Parseval labelled precision/recall. For extra word errors, there may be more than one gold standard tree (see (6) and (7)). When this happens the parser output tree is evaluated against all gold standard trees and the maximum f-score is chosen.

We carry out five experiments. In the first experiment, *E0*, we apply the parser, trained on well-formed data, to noisy input. The purpose of *E0* is to ascertain how well a parser trained on grammatical sentences, can ignore grammatical noise. *E0* provides a baseline against which the subsequent experimental results can be judged. In the *E1* experiments, the parser is retrained using the ungrammatical version of *WSJ2-21*. In experiment *E1error*, the parser is trained on ungrammatical material only, i.e. the noisy version of *WSJ2-21*. In experiment *E1mixed*, the parser is trained on grammatical and ungrammatical material, i.e. the original *WSJ2-21* is merged with the noisy *WSJ2-21*. In the *E2* experiments, a classifier is applied to the input sentence. If the sentence is classified as ungrammatical, a version of the parser that has been trained on ungrammatical data is employed. In the *E2ngram* experiment, we train a J48 decision tree classifier. Following Wagner et al. (2007), the decision tree features are part-of-speech n-gram frequency counts, with n ranging from 2 to 7 and with a subset of the BNC as the frequency reference corpus. The decision tree is trained on the original *WSJ2-21* and the ungrammatical *WSJ2-21*. In the *E2prob* experiment, the input sentence is parsed with two parsers, the original parser (the *E0* parser) and the parser trained on ungrammatical material (either the *E1error* or the *E1mixed* parser). A very simple classifier is used to decide which parser output to choose: if the *E1* parser returns a higher parse probability for the most likely tree than the *E0* parser, the *E1* parser output is returned. Otherwise the *E0* parser output is returned.

The baseline $E0$ results are in the first column of Table 2. As expected, the performance of a parser trained on well-formed input degrades when faced with ungrammatical input. It is also not surprising that its performance is worse on *Noisiest00* (-8.8% f-score) than it is on *Noisy00* (-4.3%) since the *Noisiest00* sentences contain two errors rather than one.

The *E1* results occupy the second and third columns of Table 2. An up arrow indicates a statistically significant improvement over the baseline results, a down arrow a statistically significant decline and a dash a change which is not statistically significant ($p < 0.01$). Training the parser on ungrammatical data has a positive effect on its performance on *Noisy00* and *Noisiest00* but has a negative effect on its performance on *WSJ00*. Training on a combination of grammatical and ungrammatical material gives the best results for all three development sets. Therefore, for the *E2* experiments we use the *E1mixed* parser rather than the *E1error* parser.

The *E2* results are shown in the last two columns of Table 2 and the accuracy of the two classifiers in Table 3. Over the three test sets, the *E2prob* classifier outperforms the *E2ngram* classifier. Both classifiers misclassify approximately 45% of the *Noisy00* sentences. However, the sentences misclassified by the *E2prob* classifier are those that are handled well by the *E0* parser, and this is reflected in the parsing results for *Noisy00*. An important feature of the

Dev Set	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
	E0			*E1-error*			*E1-mixed*			*E2prob*			*E2ngram*		
WSJ00	91.5	90.3	90.9	91.0–	89.4 ↓	90.2	91.3–	89.8 ↓	90.5	91.5–	90.2–	90.9	91.3–	89.9↓	90.6
Noisy00	87.5	85.6	86.6	89.4 ↑	86.6 ↑	88.0	89.4 ↑	86.8 ↑	88.1	89.1 ↑	86.8 ↑	87.9	88.7↑	86.2↑	87.5
Noisiest00	83.5	80.8	82.1	87.6 ↑	83.6 ↑	85.6	87.6 ↑	83.8 ↑	85.7	87.2 ↑	83.7 ↑	85.4	86.6↑	83.0↑	84.8

Table 2: Results of Parsing Experiments

Development Set	**E2prob**	**E2ngram**
WSJ00	76.7%	63.3%
Noisy00	55.1%	55.6%
Noisiest00	70.2%	66.0%

Table 3: $E2$ Classifier Accuracy

Test Set	**P**	**R**	**F**	**P**	**R**	**F**
	E0			*E2prob*		
WSJ23	91.7	90.8	91.3	91.7–	90.7–	91.2
Noisy23	87.4	85.6	86.5	89.2 ↑	87.0 ↑	88.1
Noisiest23	83.2	80.8	82.0	87.4 ↑	84.1 ↑	85.7

Table 4: Final Results for Section 23 Test Sets

E2prob classifier is that its use results in a constant performance on the grammatical data - with no significant degradation from the baseline.

Taking the *E2prob* results as our optimum, we carry out the same experiment again on our *WSJ23* test sets. The results are shown in Table 4. The same effect can be seen for the test sets as for the development sets - a significantly improved performance on the ungrammatical data *without* an accompanying performance decrease for the grammatical data. The *Noisy23* breakdown by error type is shown in Table 5. The error type which the original parser is most able to ignore is an agreement error. For this error type alone, the ungrammatical training material seems to hinder the parser. The biggest improvement occurs for real word spelling errors.

5 Conclusion

We have shown that it is possible to tune a WSJ-trained statistical parser to ungrammatical text *without affecting its performance on grammatical text*. This has been achieved using an automatically generated ungrammatical version of the WSJ treebank and a simple binary classifier which compares parse probabilities. The next step in this research is to see how the method copes on 'real' errors - this will require manual parsing of a suitably large test set.

Error Type	**P**	**R**	**F**	**P**	**R**	**F**
	E0			*E2-prob*		
Missing Word	88.5	83.7	86.0	88.9	84.3	86.5
Extra Word	87.2	89.4	88.3	89.2	89.7	89.4
Real Word Spell	84.3	83.0	83.7	89.5	88.2	88.9
Agreement	90.4	88.8	89.6	90.3	88.6	89.4
Verb Form	88.6	87.0	87.8	89.1	87.9	88.5

Table 5: *Noisy23*: Breakdown by Error Type

Acknowledgments We thank the IRCSET Embark Initiative (postdoctoral fellowship P/04/232) for supporting this research.

References

Eugene Charniak and Mark Johnson. 2005. Course-to-fine n-best-parsing and maxent discriminative reranking. In *Proceedings of ACL-2005*.

Jennifer Foster. 2005. *Good Reasons for Noting Bad Grammar: Empirical Investigations into the Parsing of Ungrammatical Written English*. Ph.D. thesis, University of Dublin, Trinity College.

Jennifer Foster. 2007. Treebanks gone bad: Parser evaluation and retraining using a treebank of ungrammatical sentences. *IJDAR*, 10(3-4), December.

Frederik Fouvry. 2003. *Robust Processing for Constraint-based Grammar Formalisms*. Ph.D. thesis, University of Essex.

Kong Joo Lee, Cheol Jung Kweon, Jungyun Seo, and Gil Chang Kim. 1995. A robust parser based on syntactic information. In *Proceedings of EACL-1995*.

Daisuke Okanohara and Jun'ichi Tsujii. 2007. A discriminative language model with pseudo-negative examples. In *Proceedings of ACL-2007*.

Carolyn Penstein Rosé and Alon Lavie. 1997. An efficient distribution of labor in a two stage robust interpretation process. In *Proceedings of EMNLP-1997*.

David Schneider and Kathleen McCoy. 1998. Recognizing syntactic errors in the writing of second language learners. In *Proceedings of ACL/COLING-1998*.

Noah A. Smith and Jason Eisner. 2005. Contrastive Estimation: Training Log-Linear Models on Unlabeled Data. In *Proceedings of ACL-2005*.

Joachim Wagner, 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*.