Judging Grammaticality with Tree Substitution Grammar Derivations

Matt Post
Human Language Technology Center of Excellence
Johns Hopkins University
Baltimore, MD 21211

Abstract

In this paper, we show that local features computed from the derivations of tree substitution grammars — such as the identify of particular fragments, and a count of large and small fragments — are useful in binary grammatical classification tasks. Such features outperform n-gram features and various model scores by a wide margin. Although they fall short of the performance of the hand-crafted feature set of Charniak and Johnson (2005) developed for parse tree reranking, they do so with an order of magnitude fewer features. Furthermore, since the TSGs employed are learned in a Bayesian setting, the use of their derivations can be viewed as the automatic discovery of tree patterns useful for classification. On the BLLIP dataset, we achieve an accuracy of 89.9% in discriminating between grammatical text and samples from an n-gram language model.

1 Introduction

The task of a language model is to provide a measure of the grammaticality of a sentence. Language models are useful in a variety of settings, for both human and machine output; for example, in the automatic grading of essays, or in guiding search in a machine translation system. Language modeling has proved to be quite difficult. The simplest models, n-grams, are self-evidently poor models of language, unable to (easily) capture or enforce long-distance linguistic phenomena. However, they are easy to train, are long-studied and well understood, and can be efficiently incorporated into search procedures, such as for machine translation. As a result, the output of such text generation systems is often very poor grammatically, even if it is understandable.

Since grammaticality judgments are a matter of the syntax of a language, the obvious approach for modeling grammaticality is to start with the extensive work produced over the past two decades in the field of parsing. This paper demonstrates the utility of local features derived from the fragments of tree substitution grammar derivations. Following Cherry and Quirk (2008), we conduct experiments in a classification setting, where the task is to distinguish between real text and "pseudo-negative" text obtained by sampling from a trigram language model (Okanohara and Tsujii, 2007). Our primary points of comparison are the latent SVM training of Cherry and Quirk (2008), mentioned above, and the extensive set of local and nonlocal feature templates developed by Charniak and Johnson (2005) for parse tree reranking. In contrast to this latter set of features, the feature sets from TSG derivations require no engineering; instead, they are obtained directly from the identity of the fragments used in the derivation, plus simple statistics computed over them. Since these fragments are in turn learned automatically from a Treebank with a Bayesian model, their usefulness here suggests a greater potential for adapting to other languages and datasets.

2 Tree substitution grammars

Tree substitution grammars (Joshi and Schabes, 1997) generalize context-free grammars by allowing nonterminals to rewrite as tree fragments of arbitrary size, instead of as only a sequence of one or

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:shortpapers, pages 217–222,
Portland, Oregon, June 19-24, 2011.

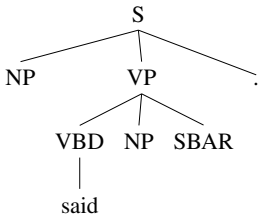

Figure 1: A Tree Substitution Grammar fragment.

more children. Evaluated by parsing accuracy, these grammars are well below state of the art. However, they are appealing in a number of ways. Larger fragments better match linguists' intuitions about what the basic units of grammar should be, capturing, for example, the predicate-argument structure of a verb (Figure 1). The grammars are context-free and thus retain cubic-time inference procedures, yet they reduce the independence assumptions in the model's generative story by virtue of using fewer fragments (compared to a standard CFG) to generate a tree.

3 A spectrum of grammaticality

The use of large fragments in TSG grammar derivations provides reason to believe that such grammars might do a better job at language modeling tasks. Consider an extreme case, in which a grammar consists entirely of complete parse trees. In this case, ungrammaticality is synonymous with parser failure. Such a classifier would have perfect precision but very low recall, since it could not generalize at all. On the other extreme, a context-free grammar containing only depth-one rules can basically produce an analysis over any sequence of words. However, such grammars are notoriously leaky, and the existence of an analysis does not correlate with grammaticality. Context-free grammars are too poor models of language for the linguistic definition of grammaticality (a sequence of words in the language of the grammar) to apply.

TSGs permit us to posit a spectrum of grammaticality in between these two extremes. If we have a grammar comprising small and large fragments, we might consider that larger fragments should be less likely to fit into ungrammatical situations, whereas small fragments could be employed almost anywhere as a sort of ungrammatical glue. Thus, on average, grammatical sentences will license derivations with larger fragments, whereas ungrammatical sentences will be forced to resort to small fragments. This is the central idea explored in this paper.

This raises the question of what exactly the larger fragments are. A fundamental problem with TSGs is that they are hard to learn, since there is no annotated corpus of TSG derivations and the number of possible derivations is exponential in the size of a tree. The most popular TSG approach has been Data-Oriented Parsing (Scha, 1990; Bod, 1993), which takes *all* fragments in the training data. The large size of such grammars (exponential in the size of the training data) forces either implicit representations (Goodman, 1996; Bansal and Klein, 2010) — which do not permit arbitrary probability distributions over the grammar fragments — or explicit approximations to all fragments (Bod, 2001). A number of researchers have presented ways to address the learning problems for explicitly represented TSGs (Zollmann and Sima'an, 2005; Zuidema, 2007; Cohn et al., 2009; Post and Gildea, 2009a). Of these approaches, work in Bayesian learning of TSGs produces intuitive grammars in a principled way, and has demonstrated potential in language modeling tasks (Post and Gildea, 2009b; Post, 2010). Our experiments make use of Bayesian-learned TSGs.

4 Experiments

We experiment with a binary classification task, defined as follows: given a sequence of words, determine whether it is grammatical or not. We use two datasets: the Wall Street Journal portion of the Penn Treebank (Marcus et al., 1993), and the BLLIP '99 dataset,[1] a collection of automatically-parsed sentences from three years of articles from the Wall Street Journal.

For both datasets, positive examples are obtained from the leaves of the parse trees, retaining their tokenization. Negative examples were produced from a trigram language model by randomly generating sentences of length no more than 100 so as to match the size of the positive data. The language model was built with SRILM (Stolcke, 2002) using interpolated Kneser-Ney smoothing. The average sentence lengths for the positive and negative data were 23.9 and 24.7, respectively, for the Treebank data

[1] LDC Catalog No. LDC2000T43.

dataset	training	devel.	test
Treebank	3,836	2,690	3,398
	91,954	65,474	79,998
BLLIP	100,000	6,000	6,000
	2,596,508	155,247	156,353

Table 1: The number of sentences (first line) and words (second line) using for training, development, and testing of the classifier. Each set of sentences is evenly split between positive and negative examples.

and 25.6 and 26.2 for the BLLIP data.

Each dataset is divided into training, development, and test sets. For the Treebank, we trained the n-gram language model on sections 2 - 21. The classifier then used sections 0, 24, and 22 for training, development, and testing, respectively. For the BLLIP dataset, we followed Cherry and Quirk (2008): we randomly selected 450K sentences to train the n-gram language model, and 50K, 3K, and 3K sentences for classifier training, development, and testing, respectively. All sentences have 100 or fewer words. Table 1 contains statistics of the datasets used in our experiments.

To build the classifier, we used `liblinear` (Fan et al., 2008). A bias of 1 was added to each feature vector. We varied a cost or regularization parameter between $1e-5$ and 100 in orders of magnitude; at each step, we built a model, evaluating it on the development set. The model with the highest score was then used to produce the result on the test set.

4.1 Base models and features

Our experiments compare a number of different feature sets. Central to these feature sets are features computed from the output of four language models.

1. Bigram and trigram language models (the same ones used to generate the negative data)
2. A Treebank grammar (Charniak, 1996)
3. A Bayesian-learned tree substitution grammar (Post and Gildea, 2009a)[2]
4. The Charniak parser (Charniak, 2000), run in language modeling mode

The parsing models for both datasets were built from sections 2 - 21 of the WSJ portion of the Treebank. These models were used to score or parse the training, development, and test data for the classifier. From the output, we extract the following feature sets used in the classifier.

- **Sentence length** (l).
- **Model scores** (S). Model log probabilities.
- **Rule features** (R). These are counter features based on the atomic unit of the analysis, i.e., individual n-grams for the n-gram models, PCFG rules, and TSG fragments.
- **Reranking features** (C&J). From the Charniak parser output we extract the complete set of reranking features of Charniak and Johnson (2005), and just the local ones (C&J local).[3]
- **Frontier size** ($\mathcal{F}_n, \mathcal{F}_n^l$). Instances of this feature class count the number of TSG fragments having frontier size $n, 1 \leq n \leq 9$.[4] Instances of $\mathcal{F}_n^l$ count only lexical items for $0 \leq n \leq 5$.

4.2 Results

Table 2 contains the classification results. The first block of models all perform at chance. We experimented with SVM classifiers instead of maximum entropy, and the only real change across all the models was for these first five models, which saw classification rise to 55 to 60%.

On the BLLIP dataset, the C&J feature sets perform the best, even when the set of features is restricted to local ones. However, as shown in Table 3, this performance comes at a cost of using ten times as many features. The classifiers with TSG features outperform all the other models.

The (near)-perfect performance of the TSG models on the Treebank is a result of the large number of features relative to the size of the training data:

[2]The sampler was run with the default settings for 1,000 iterations, and a grammar of 192,667 fragments was then extracted from counts taken from every 10th iteration between iterations 500 and 1,000, inclusive. Code was obtained from `http://github.com/mjpost/dptsg`.

[3]*Local* features can be computed in a bottom-up manner. See Huang (2008, §3.2) for more detail.

[4]A fragment's frontier is the number of terminals and nonterminals among its leaves, also known its *rank*. For example, the fragment in Figure 1 has a frontier size of 5.

feature set	Treebank	BLLIP
length (l)	50.0	46.4
3-gram score (S_3)	50.0	50.1
PCFG score (S_P)	49.5	50.0
TSG score (S_T)	49.5	49.7
Charniak score (S_C)	50.0	50.0
$l + S_3$	61.0	64.3
$l + S_P$	75.6	70.4
$l + S_T$	82.4	76.2
$l + S_C$	76.3	69.1
$l + R_2$	62.4	70.6
$l + R_3$	61.3	70.7
$l + R_P$	60.4	85.0
$l + R_T$	99.4	89.3
l + C&J (local)	89.1	92.5
l + C&J	88.6	**93.0**
$l + R_T + \mathcal{F}_* + \mathcal{F}^l_*$	**100.0**	89.9

Table 2: Classification accuracy.

feature set	Treebank	BLLIP
$l + R_3$	18K	122K
$l + R_P$	15K	**11K**
$l + R_T$	**14K**	60K
l + C&J (local)	24K	607K
l + C&J	58K	959K
$l + R_T + \mathcal{F}_*$	14K	60K

Table 3: Model size.

the positive and negative data really do evince different fragments, and there are enough such features relative to the size of the training data that very high weights can be placed on them. Manual examination of feature weights bears this out. Despite having more features available, the Charniak & Johnson feature set has significantly lower accuracy on the Treebank data, which suggests that the TSG features are more strongly associated with a particular (positive or negative) outcome.

For comparison, Cherry and Quirk (2008) report a classification accuracy of 81.42 on BLLIP. We exclude it from the table because a direct comparison is not possible, since we did not have access to the split on the BLLIP used in their experiments, but only repeated the process they described to generate it.

5 Analysis

Table 4 lists the highest-weighted TSG features associated with each outcome, taken from the BLLIP model in the last row of Table 2. The learned weights accord with the intuitions presented in Section 3. Ungrammatical sentences use smaller, abstract (unlexicalized) rules, whereas grammatical sentences use higher rank rules and are more lexicalized. Looking at the fragments themselves, we see that sensible patterns such as balanced parenthetical expressions or verb predicate-argument structures are associated with grammaticality, while many of the ungrammatical fragments contain unbalanced quotations and unlikely configurations.

Table 5 contains the most probable depth-one rules for each outcome. The unary rules associated with ungrammatical sentences show some interesting patterns. For example, the rule NP → DT occurs 2,344 times in the training portion of the Treebank. Most of these occurrences are in subject settings over articles that aren't required to modify a noun, such as *that*, *some*, *this*, and *all*. However, in the BLLIP n-gram data, this rule is used over the definite article *the* 465 times – the second-most common use. Yet this rule occurs only nine times in the Treebank where the grammar was learned. The small fragment size, together with the coarseness of the nonterminal, permit the fragment to be used in distributional settings where it should not be licensed. This suggests some complementarity between fragment learning and work in using nonterminal refinements (Johnson, 1998; Petrov et al., 2006).

6 Related work

Past approaches using parsers as language models in discriminative settings have seen varying degrees of success. Och et al. (2004) found that the score of a bilexicalized parser was not useful in distinguishing machine translation (MT) output from human reference translations. Cherry and Quirk (2008) addressed this problem by using a latent SVM to adjust the CFG rule weights such that the parser score was a much more useful discriminator between grammatical text and n-gram samples. Mutton et al. (2007) also addressed this problem by combining scores from different parsers using an SVM and showed an improved metric of fluency.

grammatical	ungrammatical
(VP VBD (NP CD) PP)	$\mathcal{F}_0^l$
(S (NP PRP) VP)	(NP (NP CD) PP)
(S NP (VP TO VP))	(TOP (NP NP NP .))
$\mathcal{F}_2^l$	$\mathcal{F}_5$
(NP NP (VP VBG NP))	(S (NP (NNP UNK-CAPS-NUM)))
(SBAR (S (NP PRP) VP))	(TOP (S NP VP (. .)))
(SBAR (IN that) S)	(TOP (PP IN NP .))
(TOP (S NP (VP (VBD said) NP SBAR) .))	(TOP (S " NP VP (. .)))
(NP (NP DT JJ NN) PP)	(TOP (S PP NP VP .))
(NP (NP NNP NNP) , NP ,)	(TOP (NP NP PP .))
(TOP (S NP (ADVP (RB also)) VP .))	$\mathcal{F}_4$
(VP (VB be) VP)	(NP (DT that) NN)
(NP (NP NNS) PP)	(TOP (S NP VP . "))
(NP NP , (SBAR WHNP (S VP)) ,)	(TOP (NP NP , NP .))
(TOP (S SBAR , NP VP .))	(QP CD (CD million))
(ADJP (QP $ CD (CD million)))	(NP NP (CC and) NP)
(SBAR (IN that) (S NP VP))	(PP (IN In) NP)
$\mathcal{F}_8$	(QP $ CD (CD million))

Table 4: Highest-weighted TSG features.

Outside of MT, Foster and Vogel (2004) argued for parsers that do not assume the grammaticality of their input. Sun et al. (2007) used a set of templates to extract labeled sequential part-of-speech patterns together with some other linguistic features) which were then used in an SVM setting to classify sentences in Japanese and Chinese learners' English corpora. Wagner et al. (2009) and Foster and Andersen (2009) attempt finer-grained, more realistic (and thus more difficult) classifications against ungrammatical text modeled on the sorts of mistakes made by language learners using parser probabilities. More recently, some researchers have shown that using features of parse trees (such as the rules used) is fruitful (Wong and Dras, 2010; Post, 2010).

grammatical	ungrammatical
(WHNP CD)	(NN UNK-CAPS)
(NP JJ NNS)	(S VP)
(PRT RP)	(S NP)
(WHNP WP NN)	(TOP FRAG)
(SBAR WHNP S)	(NP DT JJ)
(WHNP WDT NN)	(NP DT)

Table 5: Highest-weighted depth-one rules.

7 Summary

Parsers were designed to discriminate among structures, whereas language models discriminate among strings. Small fragments, abstract rules, independence assumptions, and errors or peculiarities in the training corpus allow probable structures to be produced over ungrammatical text when using models that were optimized for parser accuracy.

The experiments in this paper demonstrate the utility of tree-substitution grammars in discriminating between grammatical and ungrammatical sentences. Features are derived from the identities of the fragments used in the derivations above a sequence of words; particular fragments are associated with each outcome, and simple statistics computed over those fragments are also useful. The most complicated aspect of using TSGs is grammar learning, for which there are publicly available tools.

Looking forward, we believe there is significant potential for TSGs in more subtle discriminative tasks, for example, in discriminating between finer grained and more realistic grammatical errors (Foster and Vogel, 2004; Wagner et al., 2009), or in discriminating among translation candidates in a machine translation framework. In another line of potential work, it could prove useful to incorporate into the grammar learning procedure some knowledge of the sorts of fragments and features shown here to be helpful for discriminating grammatical and ungrammatical text.

References

Mohit Bansal and Dan Klein. 2010. Simple, accurate parsing with an all-fragments grammar. In *Proc. ACL*, Uppsala, Sweden, July.

Rens Bod. 1993. Using an annotated corpus as a stochastic grammar. In *Proc. ACL*, Columbus, Ohio, USA.

Rens Bod. 2001. What is the minimal set of fragments that achieves maximal parse accuracy? In *Proc. ACL*, Toulouse, France, July.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. ACL*, Ann Arbor, Michigan, USA, June.

Eugene Charniak. 1996. Tree-bank grammars. In *Proc. of the National Conference on Artificial Intelligence*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL*, Seattle, Washington, USA, April–May.

Colin Cherry and Chris Quirk. 2008. Discriminative, syntactic language modeling through latent svms. In *Proc. AMTA*, Waikiki, Hawaii, USA, October.

Trevor Cohn, Sharon. Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proc. NAACL*, Boulder, Colorado, USA, June.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Jennifer Foster and Øistein E. Andersen. 2009. Generrate: generating errors for use in grammatical error detection. In *Proceedings of the fourth workshop on innovative use of nlp for building educational applications*, pages 82–90. Association for Computational Linguistics.

Jennifer Foster and Carl Vogel. 2004. Good reasons for noting bad grammar: Constructing a corpus of ungrammatical language. In *Pre-Proceedings of the International Conference on Linguistic Evidence: Empirical, Theoretical and Computational Perspectives*.

Joshua Goodman. 1996. Efficient algorithms for parsing the DOP model. In *Proc. EMNLP*, Philadelphia, Pennsylvania, USA, May.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Columbus, Ohio, June.

Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages: Beyond Words*, volume 3, pages 71–122.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):330.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. Gleu: Automatic evaluation of sentence-level fluency. In *Proc. ACL*, volume 45, page 344.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, et al. 2004. A smorgasbord of features for statistical machine translation. In *Proc. NAACL*.

Daisuke Okanohara and Jun'ichi Tsujii. 2007. A discriminative language model with pseudo-negative samples. In *Proc. ACL*, Prague, Czech Republic, June.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. COLING/ACL*, Sydney, Australia, July.

Matt Post and Daniel Gildea. 2009a. Bayesian learning of a tree substitution grammar. In *Proc. ACL (short paper track)*, Suntec, Singapore, August.

Matt Post and Daniel Gildea. 2009b. Language modeling with tree substitution grammars. In *NIPS workshop on Grammar Induction, Representation of Language, and Language Learning*, Whistler, British Columbia.

Matt Post. 2010. *Syntax-based Language Models for Statistical Machine Translation*. Ph.D. thesis, University of Rochester.

Remko Scha. 1990. Taaltheorie en taaltechnologie; competence en performance. In R. de Kort and G.L.J. Leerdam, editors, *Computertoepassingen in de neerlandistiek*, pages 7–22, Almere, the Netherlands.

Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proc. International Conference on Spoken Language Processing*.

Ghihua Sun, Xiaohua Liu, Gao Cong, Ming Zhou, Zhongyang Xiong, John Lee, and Chin-Yew Lin. 2007. Detecting erroneous sentences using automatically mined sequential patterns. In *Proc. ACL*, volume 45.

Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2009. Judging grammaticality: Experiments in sentence classification. *CALICO Journal*, 26(3):474–490.

Sze-Meng Jojo Wong and Mark Dras. 2010. Parser features for sentence grammaticality classification. In *Proc. Australasian Language Technology Association Workshop*, Melbourne, Australia, December.

Andreas Zollmann and Khalil Sima'an. 2005. A consistent and efficient estimator for Data-Oriented Parsing. *Journal of Automata, Languages and Combinatorics*, 10(2/3):367–388.

Willem Zuidema. 2007. Parsimonious Data-Oriented Parsing. In *Proc. EMNLP*, Prague, Czech Republic, June.