Parsing Learner Text: to Shoehorn or not to Shoehorn

Aoife Cahill
Educational Testing Service
660 Rosedale Rd
Princeton, NJ 08541, USA
acahill@ets.org

1 Introduction

The texts written by language learners can be considered a type of non-canonical text. Language learners tend to make errors when writing in a second language and in this regard, can be seen to violate the canonical rules of a language. The kinds of errors that learners may make include: spelling, grammatical, vocabulary, collocation. The extent and degree to which learners make errors will depend on their proficiency level and this is a factor that should be taken into account when thinking about non-native writing. Highly proficient speakers will make very few errors, and given just a small sample of text it may not even be possible to identify that they are language learners. However, at the same time, the kinds of errors that even highly-proficient language learners make are often very different from the kinds of errors that a native speaker will make. A non-native speaker is likely to have the most trouble with collocations and lexical choice, whereas a native speaker will be less likely to have difficulty here (Leacock et al., 2014).

Our discussions here will focus on the syntactic analysis of English learner data. In particular, we are primarily considering learners at a low to mid-level proficiency. The kinds of mechanical and grammatical errors that these learners make are likely to cause the most difficulty for syntactic analyzers. Syntactic analysis is a key component of attempting to understand the meaning of a text. Therefore, syntactic analysis of learner text is an important step in many applications. The kinds of applications that need to analyze learner text include automated systems that detect and correct grammatical errors, systems that automatically grade texts, native language identification systems, feedback systems, etc.

2 Parsing Learner Text

Geertzen et al. (2013) parse a corpus of 1,000 learner sentences with the Stanford parser and examine the kinds of errors made by the parser. They find that in general the parser is able to recover syntactic dependency relations with high accuracy. In addition, there is only a small amount of variation across proficiency levels. They found that the parser can compensate well for morphological mistakes, but has more difficulty with more complex errors.

Although in this work we are only considering English data, it is worth pointing out some recent related work on German. Ott and Ziai (2010) apply an out-of-the-box German dependency parser to learner text and analyze the impact on down-stream semantic interpretation. They find that core functions such as subject and object can generally be reliably detected, but that when there are key elements (e.g. main verbs) missing from the sentence that the parses are less reliable. They also found that less-severe grammatical errors such as agreement did not tend to cause problems for the parser. Krivanek and Meurers (2011) compare a hand-crafted parser to a statistical parser on German data and find that the parsers are better at detecting complementary dependencies.

The highest-performing NLP tools have all been trained to perform well on well-edited text (often in the newspaper domain). There are two main problems when applying these tools to learner text which

Proceedings of LAW IX - The 9th Linguistic Annotation Workshop, pages 144–147,
Denver, Colorado, June 5, 2015. ©2015 Association for Computational Linguistics

may contain many errors. The first is that the state-of-the-art tools are robust to noise and will almost always find some analysis. Depending on the kinds of grammatical errors in the learner text, this analysis can be seriously flawed. The second issue is that often, due to the errors, a traditional linguistic analysis of learner text is not possible or appropriate.

One way of looking at the problem of training statistical NLP tools for learner texts is that learner text is of a different domain to the domain for which the NLP tools were designed. Many unsupervised approaches to domain adaptation have been proposed in the literature, which may be applicable in this scenario. Self-training (McClosky et al., 2006) is one very common and straightforward technique for improving NLP tool performance on text from a new domain. Cahill et al. (2014) showed that it was possible to improve the performance of a baseline constituency parser on learner text by applying self-training.

Another approach to adapting NLP tools to learner text is to train them directly on annotated data. The SALLE project (Syntactically Annotating Learner Language of English) at Indiana University is working towards developing a set of guidelines for annotating syntactic properties (in the form of dependencies) of texts written by learners of English (Ragheb and Dickinson, 2012; Ragheb and Dickinson, 2014). Their goal is to provide accurate syntactic dependency analyses for learner text given the morphological realizations of tokens, and they do not attempt to connect directly to the intended meanings. They plan to release a manually annotated dataset, and are also planning to work on bootstrapping approaches to semi-automatically annotate data.

3 Parsing and Grammaticality

Heilman et al. (2014) argue that grammaticality judgments for sentences should be made on an ordinal scale rather than the binary scale that is often used when talking about grammaticality. They propose a four-point scale where 1 is incomprehensible and 4 is native-sounding.[1] Viewing grammaticality in this way, it is likely that the performance of a syntactic parser will be more or less impacted by the severity of the grammatical error.

In order to briefly test whether different error types impact syntactic parsing to different degrees, we carry out a preliminary experiment with some artificially generated errors. We consider 6 errors that are typical of those made by language learners. These six error types were selected because they can easily be simulated, we do not make any claims about the relative "severity" of these errors here. In general, these errors would not lead to severe difficulties in interpretation for most people, however there are some cases where these errors could lead to ambiguity in interpretation. At the same time, we would predict that some of these errors would cause problems for state of the art parsers (e.g. missing determiner/preposition). We expect tolerance for grammatical errors to differ considerably between parsers and native speakers. The six errors we consider are:

1. missing determiner
2. missing preposition
3. missing pronoun
4. noun number error (plural instead of singular)
5. verb form error (present tense conjugation)
6. incorrect position of adverb

We use the parsed version of WSJ section 23 as our gold standard test corpus and use the GenERRate tool (Foster and Andersen, 2009) to artificially introduce these 6 errors into this well-formed text. The GenERRate tool allows the user to define operations that are applied to well-formed text in order to yield ill-formed text. For example, the operation to introduce a "missing determiner" error is `delete DT`. GenERRate also allows the user to specify the proportion of each error type in the output text. In our experiments, we choose a proportion of 0.03. This means for this error for example, that 3% of the determiners in the original corpus would be deleted.[2]

For each error, we process section 23 to get a version of the text containing that error. We then parse

[1] Non-word spelling errors are ignored in that scheme.

[2] Future work would include experimentation with varying this rate.

	Labeled Bracketing		
	Precision	Recall	F-Score
original	90.23	89.82	90.03
Verb form	89.73	89.24	89.48
Noun number	89.52	89.39	89.45
missing PRP	82.10	79.94	81.01
missing DT	75.49	74.65	75.07
Adverb	71.63	71.41	71.52
missing IN	73.68	68.49	70.99

Table 1: The effect on parser performance on ungrammatical text as measured by labeled constituents.

the original text as well as each modified version of section 23 with ZPar (Zhang and Clark, 2011). We evaluate the output of the parser using SParseval (Roark et al., 2006). This is necessary because the tokens in the gold standard are no longer necessarily in the parser output and standard evaluation software such as `evalb` cannot be applied. The labeled bracket constituency results are given in Table 1. The results show a large difference in parser performance across the 6 error types.

Confusing singular and plural nouns, or confusing the form of the verb lead to only very minor changes in overall constituency structure compared to parsing the original text by Zpar. This is in some ways not that unexpected, since these kinds of errors (at least in the manner they were artificially introduced) only affect the part-of-speech tag of the word. Missing determiners and prepositions lead to large drops in performance. This is expected, since without these key function words, the parser will have difficulty building up NP and PP constituents. Interestingly, the missing pronoun errors do not lead to as dramatic a drop in performance. This may be because pronouns alone form complete NP constituents and their absence will have less of an impact on the construction of the surrounding constituents.

Another important factor to consider is the evaluation metric. Evaluation metrics and annotation schemes can often mask true differences and accentuate other differences by over-counting. Rehbein and van Genabith (2007) compare three different parser evaluation metrics and show that a dependency-based evaluation is best suited to measuring the linguistic information encoded in parse trees. Unfortunately, SParseval does not take the alignment into account when computing dependency scores and so we are unable to report those scores for our experiments at this time.[3]

4 Discussion

Annotating learner text with syntactic analysis, either manually or automatically is problematic for a number of reasons. As shown above, the automatic annotation of texts that contain grammatical errors can have a large impact on parser performance, depending on the kind of error. In the examples above, only one error per sentence was ever introduced.[4] In reality, learner errors interact and can be difficult to disentangle. At the same time, these errors were artificially introduced into relatively long and complex English sentences that a language learner would not necessarily be able to produce. In Geertzen et al. (2013) the naturally occurring errors in their corpus did not seem to cause the parser too much trouble.

Current research has two main approaches: (1) training parsers to produce more accurate trees based on the Penn Treebank style annotation guidelines (e.g. Cahill et al. (2014)) or (2) adapting the underlying annotation schemes to better capture the fact that there may be errors in the text (e.g. Ragheb and Dickinson (2014)). The two approaches have different strengths. The first will produce the kinds of annotated trees that other NLP tools are used to getting as input. Therefore these kinds of trees fit nicely into an already existing NLP pipeline. The second will produce the kinds of annotated trees that will ultimately be more informative when it comes to developing learner-specific applications. Both approaches also have different weaknesses. The Penn Treebank style trees alone cannot provide any insight into potential errors in the sentence, and developing the tools that generate these trees such that they work well on learner text requires more work. On the other hand, a new annotation scheme requires a significant amount of manual effort in order to an-

[3] The dependency scores reported by SParseval will overly-penalize errors involving a change in surface form, as in the noun-number error.

[4] Although the algorithm GenERRate employs to insert errors according to a defined frequency would in theory allow for multiple errors per sentence, we did not see any instances of this in our data.

notate enough data to be able to train a new statistical parser.[5]

Given the encouraging results of Geertzen et al. (2013) and Cahill et al. (2014), the approach of shoehorning existing annotation schemes to fit learner data is the most practical for large-scale applications currently.

References

Aoife Cahill, Binod Gyawali, and James Bruno. 2014. Self-training for parsing learner text. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 66–73, Dublin, Ireland, August. Dublin City University.

Jennifer Foster and Oistein 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, Boulder, Colorado, June. Association for Computational Linguistics.

Jeroen Geertzen, Theodora Alexopoulou, and Anna Korhonen. 2013. Automatic linguistic annotation of large scale L2 databases: the EF-Cambridge Open Language Database (EFCAMDAT). In *Proceedings of the 31st Second Language Research Forum.*

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting Grammaticality on an Ordinal Scale. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 174–180, Baltimore, Maryland, June. Association for Computational Linguistics.

Rafiq Abdul Khader, Tracy Holloway King, and Miriam Butt. 2004. Deep call grammars: The lfgot experiment.

Julia Krivanek and Detmar Meurers. 2011. Comparing rule-based and datadriven dependency parsing of learner language. In *Proceedings of the Int. Conference on Dependency Linguistics (Depling 2011)*, pages 310–317.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. Automated Grammatical Error Detection for Language Learners, Second Edition. *Synthesis Lectures on Human Language Technologies*, 7(1):1–170.

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

Wolfgang Menzel and Ingo Schröder. 1999. Error diagnosis for language learning systems. *ReCALL*, 11:20–30.

Niels Ott and Ramon Ziai. 2010. Evaluating dependency parsing performance on German learner language. In *Proceedings of the Ninth Workshop on Treebanks and Linguistic Theories (TLT-9)*, pages 175–186.

Marwa Ragheb and Markus Dickinson. 2012. Defining Syntax for Learner Language Annotation. In *Proceedings of COLING 2012: Posters*, pages 965–974, Mumbai, India, December. The COLING 2012 Organizing Committee.

Marwa Ragheb and Markus Dickinson. 2014. Developing a Corpus of Syntactically-Annotated Learner Language for English. In *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 137–148, Tübingen, Germany.

Ines Rehbein and Josef van Genabith. 2007. Evaluating Evaluation Measures. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA-2007)*, Tartu, Estonia.

Brian Roark, Mary Harper, Eugene Charniak, Bonnie Dorr, Mark Johnson, Jeremy G Kahn, Yang Liu, Mari Ostendorf, John Hale, Anna Krasnyanskaya, Matthew Lease, Izhak Shafran, Matthew Snover, Robin Stewart, and Lisa Yung. 2006. Sparseval: Evaluation metrics for parsing speech. In *Proceedings of LREC.*

Anne Vandeventer Faltin. 2003. *Syntactic Error Diagnosis in the context of Computer Assisted Language Learning*. Ph.D. thesis, Université de Genève.

Yue Zhang and Stephen Clark. 2011. Syntactic Processing Using the Generalized Perceptron and Beam Search. *Computational Linguistics*, 37(1):105–151.

[5] There has also been work on extending non-statistical hand-crafted grammars to return structures that indicate the location of grammatical errors (Menzel and Schröder, 1999; Vandeventer Faltin, 2003; Khader et al., 2004).