Automatic Detection of Sentence Fragments

Chak Yan Yeung and John Lee
Halliday Centre for Intelligent Applications of Language Studies
Department of Linguistics and Translation
City University of Hong Kong
chak.yeung@my.cityu.edu.hk
jsylee@cityu.edu.hk

Abstract

We present and evaluate a method for automatically detecting sentence fragments in English texts written by non-native speakers. Our method combines syntactic parse tree patterns and parts-of-speech information produced by a tagger to detect this phenomenon. When evaluated on a corpus of authentic learner texts, our best model achieved a precision of 0.84 and a recall of 0.62, a statistically significant improvement over baselines using non-parse features, as well as a popular grammar checker.

1 Introduction

It is challenging to detect and correct sentence-level grammatical errors because it involves automatic syntactic analysis on noisy, learner sentences. Indeed, none of the teams achieved any recall for comma splices in the most recent CoNLL shared task (Ng et al., 2014). Sentence fragments fared hardly better: of the thirteen teams, two scored a recall of 0.25 for correction and another scored 0.2; the rest did not achieve any recall.

Although parser performance degrades on learner text (Foster, 2007), parsers can still be useful for identifying grammatical errors if they produce consistent patterns that indicate these errors. We show that parse tree patterns, automatically derived from training data, significantly improve system performance on detecting sentence fragments.

The rest of the paper is organized as follows. The next section defines the types of sentence fragments treated in this paper. Section 3 reviews related work. Section 4 describes the features used in our model. Section 5 discusses the datasets and section 6 analyzes the experiment results. Our best model significantly outperforms baselines that do not consider syntactic information and a widely used grammar checker.

2 Sentence Fragment

Every English sentence must have a main or independent clause. Most linguists require a clause to contain a subject and a finite verb (Hunt, 1965; Polio, 1997); otherwise, it is considered a sentence fragment. Following Bram (1995), we classify sentence fragments into the following four categories:

No Subject. Fragments that lack a subject,[1] such as "According to the board, is $100."

No finite verb. Fragments that lack a finite verb. These may be a nonfinite verb phrase, or a noun phrase, such as "Mrs. Kern in a show."

No subject and finite verb. Fragments lacking both a subject and a finite verb; a typical example is a prepositional phrase, such as "Up through the ranks."

Subordinate clause. These fragments consist of a stand-alone subordinate clause; the clause typically begins with a relative pronoun or a subordinating conjunction, such as "While they take pains to hide their assets."

3 Related Work

Using parse tree patterns to judge the grammaticality of a sentence is not new. Wong and Dras (2011) exploited probabilistic context-free grammar (PCFG) rules as features for native language identification. In addition to production rules, Post (2011) incorporated parse fragment features computed from derivations of tree substitution grammars. Heilman et al. (2014) used the parse scores and syntactic features to classify the comprehensibility of learner text, though they made no attempt to correct the errors.

In current grammatical error correction systems, parser output is used mainly to locate

[1] Our evaluation data distinguishes between imperatives and fragments. Our automatic classifier, however, makes no such attempt because it would require analysis of the context and significant real-world knowledge.

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), pages 599–603,
Beijing, China, July 26-31, 2015.

relevant information involved in long-distance grammatical constructions (Tetreault et al., 2010; Yoshimoto et al., 2013; Zhang and Wang, 2014). To the best of our knowledge, the only previous work that used distinctive parse patterns to detect specific grammatical errors was concerned with comma splices. Lee et al. (2014) manually identified distinctive production rules which, when used as features in a CRF, significantly improved the precision and recall in locating comma splices in learner text. Our method will similarly leverage parse tree patterns, but with the goal of detecting sentence fragment errors. More importantly, our approach is fully automatic, and can thus potentially be broadly applied on other syntax-related learner errors.

Many commercial systems, such as the Criterion Online Writing Service (Burstein et al., 2004), Grammarly[2], and WhiteSmoke[3], give feedback about sentence fragments. To the best of our knowledge, these systems do not explicitly consider parse tree patterns. The grammar checker embedded in Microsoft Word also gives feedback about sentence fragments, and will serve as one of our baselines.

Aside from the CoNLL-2014 shared task (see Section 1), the only other reported evaluation on detecting or correcting sentence fragments has been performed on Microsoft ESL Assistant and the NTNU Grammar Checker (Chen, 2009). Neither tool detected any of the sentence fragments in the test set.

4 Fragment Detection

We cast the problem of sentence fragment detection as a multiclass classification task. Given a sentence, the system would mark it either as false, if it is not a fragment, or as one of the four fragment categories described in Section 2. Rather than a binary decision on whether a sentence is a fragment, this categorisation provides more useful feedback to the learner, since each of the four fragment categories requires its own correction strategy.

4.1 Models

Baseline Models. We trained three baseline models with features that incorporate an increasing amount of information about sentence structure. The first baseline model was trained on the word trigrams of the sentences, the second model on part-of-speech unigrams, and the third on part-of-speech trigrams. All of these features can be obtained without syntactic parsing. To reduce the number of features, we filtered out the word trigrams that occur less than twenty times and the POS trigrams that occur less than a hundred times in the training data.

Parse Models. Our approach uses parse tree patterns as features. Although any arbitrary subtree structure can potentially serve as a feature, the children of the root of the tree tend to be most salient. These nodes usually denote the syntactic constituents of the sentence, and so often reveal differences between well-formed sentences and fragments. Consider the sentence "While Peter was a good boy.", shown in the parse tree in Figure 1. The child of the root of the tree is SBAR. When the subordinating conjunction "while" is removed to yield a well-formed sentence, the children nodes change accordingly into the expected NP and VP. In contrast, the POS tags, used in the baseline models, tend to remain the same.

We use the label of the root and the trigrams of its children nodes as features, similar to Sjöbergh (2005) and Lin et al. (2011). We also extend our patterns to grandchildren in some cases. When analyzing an ill-formed sentence, the parser can sometimes group words into constituents to which they do not belong, such as forming a VP that does not contain a verb. For example, the phrase "up the hill" was analyzed as a VP in the fragment "A new challenger up the hill" when in fact the sentence is missing a verb. To take into account such misanalyses, we also include the POS tag of the first child of all NP, VP, PP, ADVP, and ADJP as features. The first child is chosen because it often exposes the parsing error, as is the case with the preposition "up" in the purported VP "up the hill" in the above example.

We trained two models for experiments: the "Parse" model used the parser's POS tags and the "Parse + Tag" model used the tags produced by the POS tagger, which was trained on local features and tends to be less affected by ill-formed sentence structures. For example, in the sentence "Certainly was not true.", the word "certainly" was tagged as a plural noun by the parser while the tagger correctly identified it as an adverb. The NP construction in the fragment was encoded as "NP-

[2]www.grammarly.com
[3]www.whitesmoke.com

NNP" in the "Parse" model and "NP-RB" in the "Parse + Tag" model. To reduce the number of features, we filtered out the node trigrams that occur less than ten times in the training data.

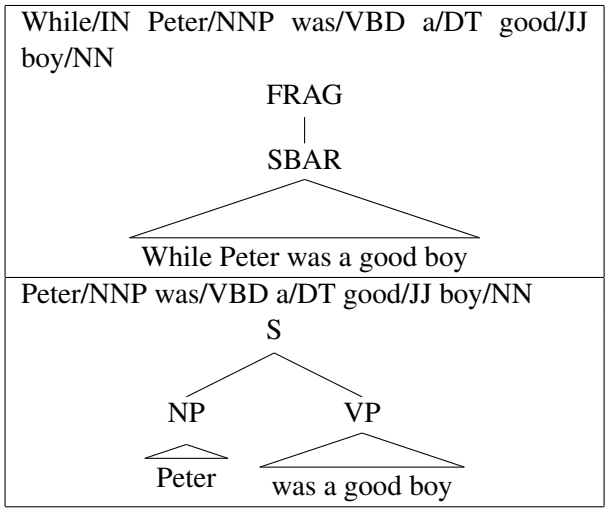

Figure 1: The POS-tagged words and parse trees of the fragment "While Peter was a good boy." and the well-formed sentence "Peter was a good boy.".

5 Data

5.1 Training Data

We automatically produced training data from the New York Times portion of the AQUAINT Corpus of English News Text (Graff, 2002). Similar to Foster and Andersen (2009), we artificially generate fragments that correspond to the four categories (Section 2) by removing different components from well-formed English sentences. For the "no subject" category, the NP immediately under the topmost S was removed. For the "no finite verb" category, we removed the finite verb in the VP immediately under the topmost S. For the "no subject and finite verb" category, we removed both the NP and the finite verb in the VP immediately under the topmost S. For the "subordinate clause" category, we looked for any SBAR in the sentence that is preceded by a comma and consists of an IN child followed by an S. The words under the SBAR are extracted as the fragment. Using this method, we created a total of 60,000 fragments, with 15,000 sentences in each category. Together with the original sentences, our training data consists of 120,000 sentences, half of which are fragments.

5.2 Evaluation Data

Fragment was among the 28 error types introduced in the CoNLL-2014 shared task (Ng et al., 2014), but the test set used in the task only contained 16 such errors and is too small for our purpose. Instead, we evaluated our system on the NUCLE corpus (Dahlmeier et al., 2013), which was used as the training data in the shared task. The error label "SFrag" in the NUCLE corpus was used for sentence fragments in a wider sense than the four categories defined by Bram (1995) (see Section 2). For example, "SFrag" also labels sentences with stylistic issues, such as those beginning with "therefore" or "hence", and sentences that, though well-formed, should be merged with its neighbor, such as "In Singapore, we can see that this problem is occurring. This is so as there is a huge discrepancy in the education levels.".

We asked two human annotators to classify the fragments into the different categories described in Section 2. The kappa was 0.84. Most of the disagreements involved sentences that contain a semi-colon which, when replaced with a comma, would become well-formed. One annotator flagged these cases as fragments while the other did not, considering them to be punctuation errors. Another source of disagreements was whether a sentence should be considered an imperative.

Among the 249 sentences marked as fragments, 86 were classified as one of the Bram (1995) categories by at least one of the annotators. Most of the fragments belong to categories "no finite verb" and "subordinate clause", accounting for 43.0% and 31.4% of the cases respectively. The categories "no subject and finite verb" and "no subject" both account for 12.8% of the cases. We left all errors in the sentences in place so as to reflect our models' performance on authentic learner data.

6 Results

We obtained the POS tags and parse trees of the sentences in our datasets with the Stanford POS tagger (Toutanova et al., 2003) and the Stanford parser (Manning et al., 2014). We used the logistic regression implementation in scikit-learn (Pedregosa et al., 2011) for the maximum entropy models in our experiments. In addition to the three baseline models described in Section 4.1, we computed a fourth baseline using the grammar

checker in Microsoft Word 2013 by configuring the checker to capture "Fragments and Run-ons" and "Fragment - stylistic suggestions".

6.1 Fragment detection

We first evaluated the systems' ability to detect fragments. The fragment categories are disregarded in this evaluation and the system's result is considered correct even if its output category does not match the one marked by the annotators. We adopted the metric used in the CoNLL-2014 shared task, $F_{0.5}$, which emphasizes precision twice as much as recall because it is important to minimize false alarms for language learners[4].

The results are shown in Table 1. The "Parse" model achieved a precision of 0.82, a recall of 0.57 and an $F_{0.5}$ of 0.75. Using the POS tags produced by the POS tagger instead of the ones produced by the parser, the "Parse + Tag" model achieved a precision of 0.84, a recall of 0.62 and an $F_{0.5}$ of 0.78, improving upon the results of the "Parse" model and significantly outperforming all four baselines[5].

Most of the false negatives are in the "no finite verb" category and many of them involve fragments with subordinate clauses, such as "The increased of longevity as the elderly are leading longer lives.". In order to create parse trees that fit those of complete sentences, the parser tended to interpret the verbs in the subordinate clauses (e.g., "are" in the above example) as the fragments' main verbs, causing the errors. For false positives, the errors were caused mostly by the presence of introductory phrases. The parse trees of these sentences usually contain a PP or an ADVP immediately under the root, which is a pattern shared by fragments. The system also flagged some imperative sentences as fragments.

6.2 Fragment classification

For the fragments that the system has correctly identified, we evaluated their classification accuracy[6]. Table 2 shows the confusion matrix of the system's results.

The largest source of error is the system wrongly classifying 'no finite verb" and "subordinate clause" fragments as "no subject and finite verb". Most of these involve fragments that begin with a prepositional phrase, such as "for example", followed by a comma. The annotators treated the prepositional phrase as introductory phrase and focused on the segment after the comma. In contrast, based on the parser output, the system often treated the entire fragment as a PP, which should then belong to "no subject and finite verb". It can be argued that both interpretations are valid. For instance, the fragment "For example, apples and oranges" can be corrected as "For example, apples and oranges are fruits" or, alternatively, "I love fruits, for example, apples and oranges".

System	$P/R/F_{0.5}$
Word Trigrams	0.20/0.03/0.09
POS Tags	0.56/0.33/0.47
POS Trigrams	0.55/0.42/0.52
MS Word	0.80/0.15/0.43
Parse	0.82/0.57/0.75
Parse + Tag	0.84/0.62/0.78

Table 1: System precision, recall and F-measure for fragment detection.

→ Expected ↓ System	S	V	SV	C
S	[6]	4	1	1
V	0	[12]	2	0
SV	0	5	[2]	11
C	0	0	0	[9]

Table 2: The confusion matrix of the system for classifying the detected sentence fragments into the categories no subject (S), no finite verb (V), no subject and finite verb (SV) and subordinate clause (C).

7 Conclusion

We have presented a data-driven method for automatically detecting sentence fragments. We have shown that our method, which uses syntactic parse tree patterns and POS tagger output, significantly improves accuracy in detecting fragments in English learner texts.

Acknowledgments

This work was supported in part by a Strategic Research Grant (#7008166) from City University of

[4] $F_{0.5}$ is calculated by $F_{0.5} = (1 + 0.5^2)$ x R x P / (R + 0.5^2 x P) for recall R and precision P.

[5] At $p \leq 0.002$ by McNemar's test.

[6] The grammar checker in Microsoft Word is excluded from this evaluation because it does not provide any correction suggestions for fragments.

Hong Kong.

References

Barli Bram. 1995. *Write Well, Improving Writing Skills*. Kanisius.

Jill Burstein, Martin Chodorow, and Claudia Leacock. 2004. Automated essay evaluation: The Criterion online writing service. *AI Magazine*, 25(3):27.

Hao-Jan Howard Chen. 2009. Evaluating two web-based grammar checkers-Microsoft ESL Assistant and NTNU statistical grammar checker. *Computational Linguistics and Chinese Language Processing*, 14(2):161–180.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The NUS Corpus of Learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

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. 2007. Treebanks gone bad. *International Journal of Document Analysis and Recognition (IJDAR)*, 10(3-4):129–145.

David Graff. 2002. The AQUAINT corpus of English news text. *Linguistic Data Consortium, Philadelphia*.

Michael Heilman, Joel Tetreault, Aoife Cahill, Nitin Madnani, Melissa Lopez, and Matthew Mulholland. 2014. Predicting grammaticality on an ordinal scale. In *Proceedings of ACL-2014*.

Kellogg W Hunt. 1965. Grammatical structures written at three grade levels. NCTE research report no. 3.

John Lee, Chak Yan Yeung, and Martin Chodorow. 2014. Automatic detection of comma splices. In *Proceedings of PACLIC-2014*.

Nay Yee Lin, Khin Mar Soe, and Ni Lar Thein. 2011. Developing a chunk-based grammar checker for translated English sentences. In *Proceedings of PACLIC-2011*, pages 245–254.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830.

Charlene G Polio. 1997. Measures of linguistic accuracy in second language writing research. *Language learning*, 47(1):101–143.

Matt Post. 2011. Judging grammaticality with tree substitution grammar derivations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 217–222. Association for Computational Linguistics.

Jonas Sjöbergh. 2005. Chunking: an unsupervised method to find errors in text. In *Proceedings of the 15th NODALIDA conference*, pages 180–185.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of ACL-2010*, pages 353–358. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.

Sze-Meng Jojo Wong and Mark Dras. 2011. Exploiting parse structures for native language identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610. Association for Computational Linguistics.

Ippei Yoshimoto, Tomoya Kose, Kensuke Mitsuzawa, Keisuke Sakaguchi, Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, and Yuji Matsumoto. 2013. NAIST at 2013 CoNLL grammatical error correction shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, volume 26.

Longkai Zhang and Houfeng Wang. 2014. Go climb a dependency tree and correct the grammatical errors. In *Proceedings of EMNLP-2014*.