# The Effect of Multiple Grammatical Errors on Processing Non-Native Writing

**Courtney Napoles**
Johns Hopkins University
courtneyn@jhu.edu

**Aoife Cahill** **Nitin Madnani**
Educational Testing Service
{acahill,nmadnani}@ets.org

## Abstract

In this work, we estimate the deterioration of NLP processing given an estimate of the amount and nature of grammatical errors in a text. From a corpus of essays written by English-language learners, we extract ungrammatical sentences, controlling the number and types of errors in each sentence. We focus on six categories of errors that are commonly made by English-language learners, and consider sentences containing one or more of these errors. To evaluate the effect of grammatical errors, we measure the deterioration of ungrammatical dependency parses using the labeled F-score, an adaptation of the labeled attachment score. We find notable differences between the influence of individual error types on the dependency parse, as well as interactions between multiple errors.

## 1 Introduction

With the large number of English-language learners and the prevalence of informal web text, noisy text containing grammatical errors is widespread. However, the majority of NLP tools are developed and trained over clean, grammatical text and the performance of these tools may be negatively affected when processing errorful text. One possible workaround is to adapt tools for noisy text, e.g. (Foster et al., 2008; Cahill et al., 2014). However, it is often preferable to use tools trained on clean text, mainly because of the resources necessary for training and the limited availability of large-scale annotated corpora, but also because tools should work correctly in the presence of well-formed text.

Our goal is to measure the performance degradation of an automatic NLP task based on an estimate of grammatical errors in a text. For example, if we are processing student responses within an NLP application, and the responses contain a mix of native and non-native texts, it would be useful to be able to estimate the difference in performance (if any) of the NLP application on both types of texts.

We choose dependency parsing as our prototypic task because it is often one of the first complex downstream tasks in NLP pipelines. We will consider six common grammatical errors made by non-native speakers of English and systematically control the number and types of errors present in a sentence. As errors are introduced to a sentence, the degradation of the dependency parse is measured by the decrease in the F-score over dependency relations.

In this work, we will show that

- increasing the number of errors in a sentence decreases the accuracy of the dependency parse (Section 4.1);
- the distance between errors does not affect the accuracy (Section 4.2);
- some types of grammatical errors have a greater impact, alone or in combination with other errors (Section 4.3).

While these findings may seem self-evident, they have not previously been quantified on a large corpus of naturally occurring errors. Our analysis will serve as the first step to understanding what happens to a NLP pipeline when confronted with grammatical errors.



*Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–11,
San Diego, California, June 16, 2016. 

## 2 Data

Previous research concerning grammatical errors has artificially generated errors over clean text, such as Foster et al. (2008) and Felice and Yuan (2014), among others. While this is one approach for building a large-scale corpus of grammatical and ungrammatical sentence pairs, we use text with naturally occurring errors so that our analysis covers the types of errors typically seen in non-native writing.

As the source of our data, we use the training section of the NUS Corpus of Learner English (NUCLE),[1] which is a large corpus of essays written by non-native English speakers (Dahlmeier et al., 2013). The NUCLE corpus has been annotated with corrections to the grammatical errors, and each error has been labeled with one of 28 error categories.

We will only consider the following common errors types, which constitute more than 50% of the 44 thousand corrections in NUCLE:

- Article or determiner [*Det*]
- Mechanical (punctuation, capitalization, and spelling) [*Mec*]
- Noun number [*Noun*]
- Preposition [*Prep*]
- Word form [*Wform*]
- Verb tense and verb form [*Verb*]

While other error coding schemes specify the nature of the error (whether the text is unnecessary, missing, or needs to be replaced) in addition to the word class (Nicholls, 2004), the NUCLE error categories do not make that distinction. Therefore we automatically labeled each error with an additional tag for the *operation* of the correction, depending on whether it was *missing* a token, had an *unnecessary* token, or needed to *replace* a token. We labeled all noun, verb, and word form errors as *replacements*, and automatically detected the label of article, mechanical, and preposition errors by comparing the tokens in the original and corrected spans of text. If the correction had fewer unique tokens than the original text, it was labeled *unnecessary*. If the correction had more unique tokens, it was labeled *missing*. Otherwise the operation was labeled a *replacement*. To verify the validity of this algorithm, we reviewed the 100 most frequent error–correction pairs labeled with each operation, which encompasses 69% of the errors in the corpus.[2]

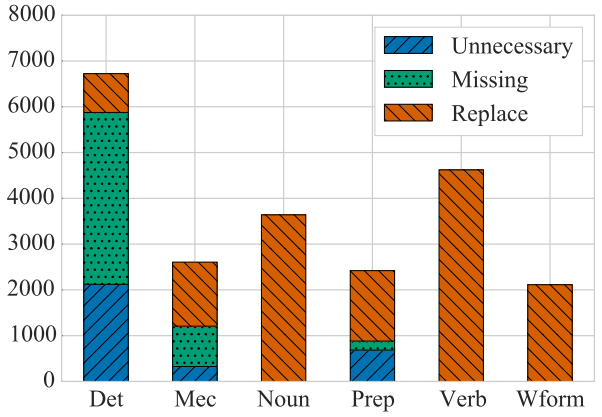


Figure 1: The number of corrections by error type and operation that we used in this study.

To compile our corpus of sentences, we selected all of the corrections from NUCLE addressing one of the six error types above. We skipped corrections that spanned multiple sentences or the entire length of a sentence, as well as corrections that addressed punctuation spacing, since those errors would likely be addressed during tokenization.[3]

We identified 14,531 NUCLE sentences containing errors subject to these criteria. We applied the corrections of all other types of errors and, in the rest of our analysis, we will use the term *errors* to refer only to errors of the types outlined above. On average, each of these sentence has 26.4 tokens and 1.5 errors, with each error spanning 1.2 tokens and the correction 1.5 tokens. In total, there are 22,123 errors, and Figure 1 shows the total number of corrections by error type and operation. Because of the small number of naturally occurring sentences with exactly 1, 2, 3, or 4 errors (Table 1), we chose to generate new sentences with varying numbers of errors from the original ungrammatical sentences.

For each of the NUCLE sentences, we generated ungrammatical sentences with $n$ errors by systematically selecting $n$ corrections to ignore, applying all of the other corrections. We generated sentences

[1] Version 3.2

[2] Many error–correction pairs are very frequent: for example, inserting or deleting *the* accounts for 3,851 of the errors and inserting or deleting a plural *s* 2,804.

[3] NLTK was used for sentence and token segmentation (`http://nltk.org`).

| # errors | NUCLE sentences | Generated sentences | Exactly $n$ errors |
|---|---|---|---|
| 1 | 14,531 | 22,123 | 9,474 |
| 2 | 5,030 | 11,561 | 3,341 |
| 3 | 572 | 5,085 | 0 |
| 4 | 570 | 3,577 | 362 |

Table 1: The number of NUCLE sentences containing at least $n$ errors, the number of sentences with $n$ errors that were generated from them, and the number of NUCLE sentences with exactly $n$ errors.

with $n = 1$ to 4 errors, when there were at least $n$ corrections to the original sentence. For example, a NUCLE sentence with 6 annotated corrections would yield the following number of ungrammatical sentences: 6 sentences with one error, $\binom{6}{2} = 15$ sentences with two errors, $\binom{6}{3} = 20$ sentences with three errors, and so on. The number of original NUCLE sentences and generated sentences with each number of errors is shown in Table 1. We also generated a grammatical sentence with all of the corrections applied for comparison.

We parsed each sentence with the ZPar constituent parser (Zhang and Clark, 2011) and generated dependency parses from the ZPar output using the Stanford Dependency Parser[4] and the universal dependencies representation (De Marneffe et al., 2014). We make the over-confident assumption that the automatic analyses in our pipeline (tokenization, parsing, and error-type labeling) are all correct.

Our analysis also depends on the quality of the NUCLE annotations. When correcting ungrammatical text, annotators are faced with the decisions of whether a text needs to be corrected and, if so, how to edit it. Previous work has found low inter-annotator agreement for the basic task of judging whether a sentence is grammatical ($0.16 \leq \kappa \leq 0.40$) (Rozovskaya and Roth, 2010). The NUCLE corpus is no different, with the three NUCLE annotators having moderate agreement on how to correct a span of text ($\kappa = 0.48$) and only fair agreement for identifying what span of text needs to be corrected ($\kappa = 0.39$) (Dahlmeier et al., 2013). Low inter-annotator agreement is not necessarily an indication of the quality of the annotations, since it could also be attributed to the diversity of appropriate corrections that have been made. We assume that the annotations are correct and complete, meaning that the spans and labels of annotations are correct and that all of the grammatical errors are annotated. We further assume that the annotations only fix grammatical errors, instead of providing a stylistic alternatives to grammatical text.

[4]Using the EnglishGrammaticalStructure class with the flags `-nonCollapsed -keepPunct`.

## 3 Metric: Labeled F-score

To measure the effect of grammatical errors on the performance of the dependency parser, we compare the dependencies identified in the corrected sentence to those from the ungrammatical sentence.

The labeled attachment score (LAS) is a commonly used method for evaluating dependency parsers (Nivre et al., 2004). The LAS calculates the accuracy of the dependency triples from the candidate dependency graph with respect to those of the gold standard, where each triple represents one relation, consisting of the head, dependent, and type of relation. The LAS assumes that the surface forms of the sentences are identical but only the relations have changed. In this work, we require a method that accommodates unaligned tokens, which occur when an error involves deleting or inserting tokens and unequal surface forms (replacement errors).

There are some metrics that compare the parses of unequal sentences, including SParseval (Roark et al., 2006) and TEDeval (Tsarfaty et al., 2011), however neither of these metrics operate over dependencies. We chose to evaluate dependencies because dependency-based evaluation has been shown to be more closely related to the linguistic intuition of good parses compared to two other tree-based evaluations (Rehbein and van Genabith, 2007).

Since we cannot calculate the LAS over sentences of unequal lengths, we instead measure the $F_1$-score of the dependency relations. So that substitutions (such as morphological changes) are not severely penalized, we represent tokens with their index instead of the surface form. First, we align the tokens in the grammatical and ungrammatical sentences and assign an index to each token such that the aligned tokens in each sentence share the same index. Because reordering is uncommon in the NUCLE corrections, we use dynamic programming to find the lowest-

cost alignment between a sentence pair, where the cost for insertions and deletions is 1, and substitutions receive a cost proportionate to the Levenshtein edit distance between the tokens (to award "partial credit" for inflections).

We calculate the Labeled F-score (LF) over dependency relations of the form <head index, dependent index, relation>. This evaluation metric can be used for comparing the dependency parses of aligned sentences with unequal lengths or tokens.[5] A variant of the LAS, the Unlabeled Attachment Score, is calculated over pairs of heads and dependents without the relation. We considered the corresponding unlabeled F-score and, since there was no meaningful difference between that and the labeled F-score, we chose to use labeled relations for greater specificity.

In the subsequent analysis, we will focus on the difference in LF before and after an error is introduced to a sentence. We will refer to the LF of a sentence with $n$ errors as $\text{LF}_n$. The LF of a sentence identical to the correct sentence is 100, therefore $\text{LF}_0$ is always 100. The decrease in LF of an ungrammatical sentence with $n$ errors from the correct parse is $\text{LF}_0 - \text{LF}_n = 100 - \text{LF}_n$, where a higher value indicates a larger divergence from the correct dependency parse.

## 4 Analysis

Our analysis will be broken down by different characteristics of ungrammatical sentences and quantifying their effect on the LF. Specifically, we will examine increasing numbers of errors in a sentence, the distance between errors, individual error types, and adding more errors to an already ungrammatical sentence.

### 4.1 Number of errors

The first step of our analysis is to verify our hypothesis that the absolute LF decrease ($\text{LF}_0 - \text{LF}_n$) increases as the number of errors in a sentence increases from $n = 1$ to $n = 4$. Pearson's correlation reveals a weak correlation between the LF decrease and number of errors (Figure 2). Since this analysis will be considering sentences generated with only a subset of errors from the original sentence, we will verify the validity of this data by comparing the LF decrease of the generated sentences to the LF decrease of sentences that originally had exactly $n$ errors. Since the LF decreases of the generated and original sentences are very similar, we presume that the generated sentences exhibit similar properties as the original sentences with the same number of errors. We further compared the distribution of sentences with each error type as the number of errors per sentence changes, and find that the distribution is fairly constant. The distribution of sentences with one error is shown in Figure 3. We will next investigate whether the LF decrease is due to interaction between errors or if there is an additive effect.

[5] Available for download at `https://github.com/cnap/ungrammatical-dependencies`.

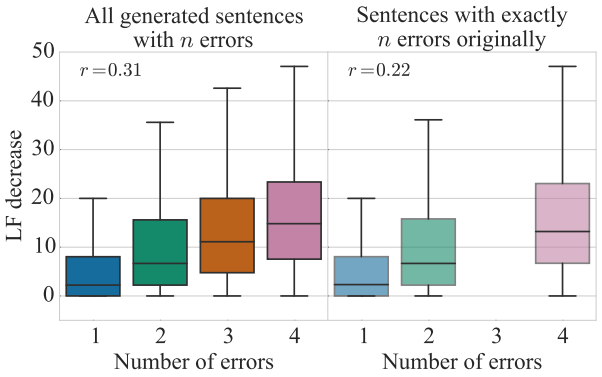


Figure 2: Mean absolute decrease in LF by the number of errors in a sentence ($100 - \text{LF}_n$).

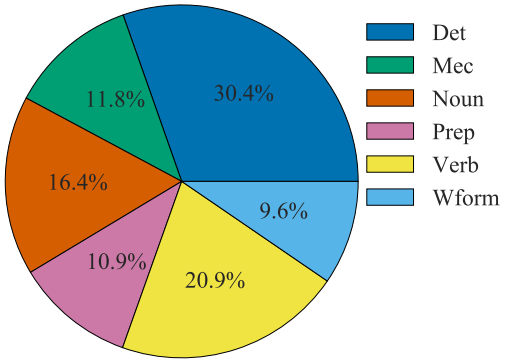


Figure 3: The distribution of error types in sentences with one error. The distribution is virtually identical ($\pm 2$ percentage points) in sentences with 2–4 errors.

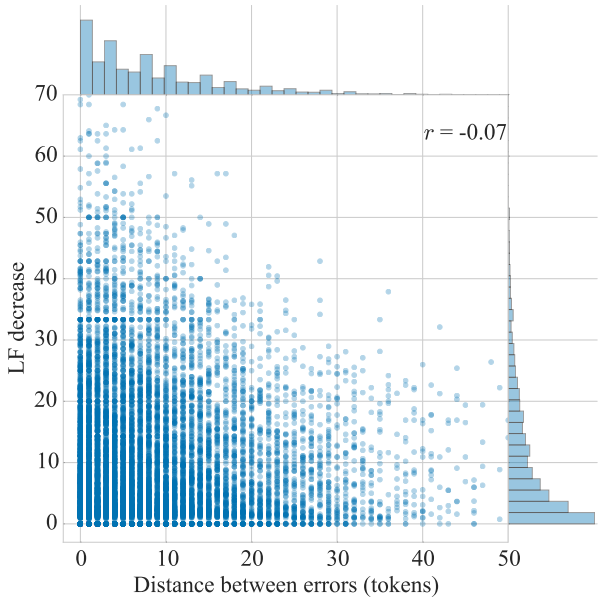


Figure 4: Distance between two errors and the decrease in LF.

### 4.2 Distance between errors

To determine whether the distance between errors is a factor in dependency performance, we took sentences with only two errors and counted the number of tokens between the errors (Figure 4). Surprisingly, there is no relationship between the distance separating errors and the dependency parse accuracy. We hypothesized that errors near each other would either interact and cause the parser to misinterpret more of the sentence, or conversely that they would disrupt the interpretation of only one clause and not greatly effect the LF. However, neither of these were evident based on the very weak negative correlation. For sentences with more than two errors, we calculated the mean, minimum, and maximum distances between all errors in each sentence, and found a weak to very weak negative correlation between those measures and the LF decrease ($-0.15 \leq r \leq -0.04$).

### 4.3 Error type and operation

Next, we considered specific error types and their operation—whether they were missing, unnecessary, or needed replacement. To isolate the impact of individual error types on the LF, we calculated the mean LF decrease ($100 - \text{LF}_1$) by error and operation over sentences with only one error (Figure 5). The mean values by error type are shown in Figure 6, column 1.

Two trends are immediately visible: there is a clear difference between error types and, except for determiner errors, missing and unnecessary errors have a greater impact on the dependency parse than replacements. Nouns and prepositions needing replacement have the lowest impact on the LF, with $100 - \text{LF}_1 < 4$. This could be because the part of speech tag for these substitutions does not often change (or only change NN to NNS in the case of nouns), which would therefore not greatly affect a dependency parser's interpretation of the sentence, but this hypothesis needs to be verified in future work. A prepositional phrase and noun phrase would likely still be found headed by that word. Verb replacements exhibit more than twice the decrease in LF than nouns and prepositions. Unlike noun and preposition replacements, replacing a verb tends to elicit greater structural changes, since some verbs can be interpreted as nouns or past participles and gerunds could be interpreted as modifying nouns, etc. (Lee and Seneff, 2008).

Determiner errors also have a low impact on LF and there is practically no difference by the operation of the correction. This can be explained because determiners occur at the beginning of noun phrases, and so deleting, inserting, or replacing a determiner would typically affect one child of the noun phrase and not the overall structure. However, mechanical errors and missing or unnecessary prepositions have a great impact on the LF, with $\text{LF}_1$ at least $10\%$ lower than $\text{LF}_0$. Inserting or deleting these types of words could greatly alter the structure of a sentence. For example, inserting a missing preposition would introduce a new prepositional phrase and the subsequent noun phrase would attach to that phrase. Regarding *Mec* errors, inserting commas can drastically change the structure by breaking apart constituents, and removing commas can cause constituents to become siblings.

### 4.4 Adding errors to ungrammatical sentences

We have seen the mean LF decrease in sentences with one error, over different error types. Next, we examine what happens to the dependency parse when an error is added to a sentence that is already ungrammatical. We calculated the LF of sen-

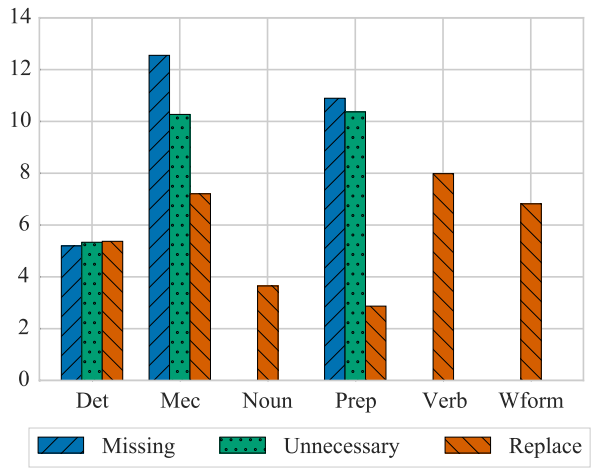


Figure 5: The mean decrease in LF ($100 - \text{LF}_1$) for sentences with one error, by error type.

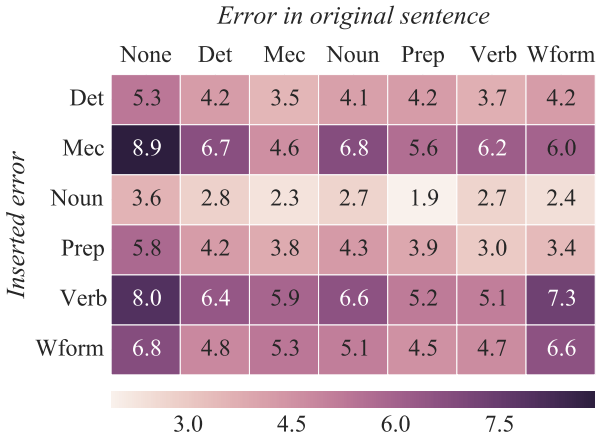


Figure 6: Mean decrease in LF ($\text{LF}_1 - \text{LF}_2$) for sentences when introducing an error (row) into a sentence that already has an error of the type in the column. The *None* column contains the mean decrease when introducing a new error to a grammatical sentence ($100 - \text{LF}_1$).

tences with one error ($\text{LF}_1$), introduced a second error into that sentence, and calculated the decrease in LF ($\text{LF}_1 - \text{LF}_2$). We controlled for the types of errors both present in the original sentence and introduced to the sentence, not differentiating the operation of the error for ease of interpretation. The mean differences by error types are in Figure 6.

Each column indicates what type of error was present in the original sentence (or the *first* error), with *None* indicating the original sentence was grammatically correct and had no errors. Each row represents the type of error that was added to the sentence (the *second* error). Note that this does not indicate the left–right order of the errors. This analysis considers all combinations of errors: for example, given a sentence with two determiner errors *A* and *B*, we calculate the LF decrease after inserting error *A* into the sentence that already had error *B* and vice versa.

Generally, with respect to the error type, the relative magnitude of change caused by adding the second error (column 2) is similar to adding that type of error to a sentence with no errors (column 1). However, introducing the second error always has a lower mean LF decrease than introducing the first error into a sentence, suggesting that each added error is less disruptive to the dependency parse as the number of errors increase.

To verify this, we added an error to sentences with 0 to 3 errors and calculated the LF change ($\text{LF}_n - \text{LF}_{n+1}$) each time a new error was introduced. Figure 7 shows the mean LF decrease after adding an error of a given type to a sentence that already had 0, 1, 2, or 3 errors.

Based on Figure 7, it appears that the LF decrease may converge for some error types, specifically determiner, preposition, verb, and noun errors. However, the LF decreases at a fairly constant rate for mechanical and word form errors, suggesting that ungrammatical sentences become increasingly uninterpretable as these types of errors are introduced. Further research is needed to make definitive claims about what happens as a sentence gets increasingly errorful.

## 5 Qualifying LF decrease

In the previous analysis, the range of LF decreases are from 1 to around 10, suggesting that approximately 1% to 10% of the dependency parse was changed due to errors. However, this begs the question of what a LF decrease of 1, 5, or 10 actually means for a pair of sentences. Is the ungrammatical sentence garbled after the LF decrease reaches a certain level? How different are the dependencies found in a sentence with a LF decrease of 1 versus 10? To illustrate these differences, we selected an example sentence and calculated the LF decrease

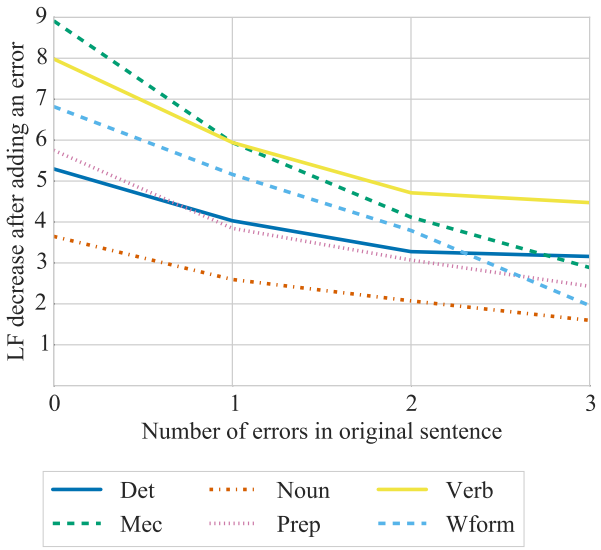


Figure 7: Mean decrease in LF ($\text{LF}_n - \text{LF}_{n+1}$) when an error of a given type is added to a sentence that already has $n$ errors.

and dependency graph as more errors were added (Table 2, Figure 8, and Figure 9).

Notice that the largest decrease in LF occurs after the first and second errors are introduced (10 and 13 points, respectively). The introductions of these errors result in structural changes of the graph, as does the fourth error, which results in a lesser LF decrease of 5. In contrast, the third error, a missing determiner, causes a lesser decrease of about 2, since the graph structure is not affected by this insertion.

Considering the LF decrease as the percent of a sentence that is changed, for a sentence with 26 tokens (the mean length of sentences in our dataset), a LF decrease of 5 corresponds to a change in 1.3 of the tokens, while a decrease of 10 corresponds to a change in 2.6 tokens. Lower LF decreases ($< 5$ or so) generally indicate the insertion or deletion of a token that does not affect the graph structure, or changing the label of a dependency relation. On the other hand, greater decreases likely reflect a structural change in the dependency graph of the ungrammatical sentence, which affects more relations than those containing the ungrammatical tokens.

## 6 Related work

There is a modest body of work focused on improving parser performance of ungrammatical sentences. Unlike our experiments, most previous work has used small (around 1,000 sentences) or artificially generated corpora of ungrammatical/grammatical sentence pairs.

The most closely related works compared the structure of constituent parses of ungrammatical to corrected sentences: with naturally occurring errors, Foster (2004) and Kaljahi et al. (2015) and evaluate parses of ungrammatical text based on the constituent parse and Geertzen et al. (2013) evaluate performance over dependencies. Cahill (2015) examines the parser performance using artificially generated errors, and Foster (2007) analyzes the parses of both natural and artificial errors. In Wagner and Foster (2009), the authors compared the parse probabilities of naturally occurring and artificially generated ungrammatical sentences to the probabilities of the corrected sentences. They found that the natural ungrammatical sentences had a lower reduction in parse probability than artificial sentences, suggesting that artificial errors are not interchangeable with spontaneous errors. This analysis suggests the importance of using naturally occurring errors, which is why we chose to generate sentences from the spontaneous NUCLE errors.

Several studies have attempted to improve the accuracy of parsing ungrammatical text. Some approaches include self-training (Foster et al., 2011; Cahill et al., 2014), retraining (Foster et al., 2008), and transforming the input and training text to be more similar (Foster, 2010). Other work with ungrammatical learner text includes Caines and Buttery (2014), which identifies the need to improve parsing of spoken learner English, and Tetreault et al. (2010), which analyzes the accuracy of prepositional phrase attachment in the presence of preposition errors.

## 7 Conclusion and future work

The performance of NLP tools over ungrammatical text is little understood. Given the expense of annotating a grammatical-error corpus, previous studies have used either small annotated corpora or generated artificial grammatical errors in clean text.

This study represents the first large-scale analysis of the effect of grammatical errors on a NLP task. We have used a large, annotated corpus of grammat-

ical errors to generate more than 44,000 sentences with up to four errors in each sentence. The ungrammatical sentences contain an increasing number of naturally occurring errors, facilitating the comparison of parser performance as more errors are introduced to a sentence. This is the first step toward a larger goal of providing a confidence score of parser accuracy based on an estimate of how ungrammatical a text may be. While many of our findings may seem obvious, they have previously not been quantified on a large corpus of naturally occurring grammatical errors. In the future, these results should be verified over a selection of manually corrected dependency parses.

Future work includes predicting the LF decrease based on an estimate of the number and types of errors in a sentence. As yet, we have only measured change by the LF decrease over all dependency relations. The decrease can also be measured over individual dependency relations to get a clearer idea of which relations are affected by specific error types. We will also investigate the effect of grammatical errors on other NLP tasks.

We chose the NUCLE corpus because it is the largest annotated corpus of learner English (1.2 million tokens). However, this analysis is relies on the idiosyncrasies of this particular corpus, such as the typical sentence length and complexity. The essays were written by students at the National University of Singapore, who do not have a wide variety of native languages. The types and frequency of errors differ depending on the native language of the student (Rozovskaya and Roth, 2010), which may bias the analysis herein. The available corpora that contain a broader representation of native languages are much smaller than the NUCLE corpus: the Cambridge Learner Corpus–First Certificate in English has 420 thousand tokens (Yannakoudakis et al., 2011), and the corpus annotated by (Rozovskaya and Roth, 2010) contains only 63 thousand words.

One limitation to our method for generating ungrammatical sentences is that relatively few sentences are the source of ungrammatical sentences with four errors. Even though we drew sentences from a large corpus, only 570 sentences had at least four errors (of the types we were considering), compared to 14,500 sentences with at least one error. Future work examining the effect of multiple errors would need to consider a more diverse set of sentences with more instances of at least four errors, since there could be peculiarities or noise in the original annotations, which would be amplified in generated sentences.

## Acknowledgments

We would like to thank Martin Chodorow and Jennifer Foster for their valuable insight while developing this research, and Beata Beigman Klebanov, Brian Riordan, Su-Youn Yoon, and the BEA reviewers for their helpful feedback. This material is based upon work partially supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1232825.

## 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.

Aoife Cahill. 2015. Parsing learner text: To shoehorn or not to shoehorn. In *Proceedings of The 9th Linguistic Annotation Workshop*, pages 144–147, Denver, Colorado, USA, June. Association for Computational Linguistics.

Andrew Caines and Paula Buttery. 2014. The effect of disfluencies and learner errors on the parsing of spoken learner language. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 74–81, Dublin, Ireland, August. Dublin City University.

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, Atlanta, Georgia, June. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of Language Resources and Evaluation Conference (LREC)*, volume 14, pages 4585–4592.

Mariano Felice and Zheng Yuan. 2014. Generating artificial errors for grammatical error correction. In *Proceedings of the Student Research Workshop at the 14th*

| Num. errors | Inserted error type | LF decrease | Sentence |
|---|---|---|---|
| 0 | n/a | n/a | One of the factors that determines and shapes technological innovation the most is the country 's economic status . |
| 1 | Verb | 10.0 | One of the factors that ***determined*** and shapes technological innovation the most is the country 's economic status . |
| 2 | Mec | 13.1 | One of the factors that determined and shapes technological innovation the most is the ***country*** economic status . |
| 3 | Det | 1.9 | One of the factors that determined and shapes ***the*** technological innovation the most is the country economic status . |
| 4 | Verb | 5.0 | One of the factors that determined and ***shaped*** the technological innovation the most is the country economic status . |

Table 2: An example of a sentence with 4 errors added and the LF decrease ($\text{LF}_{n-1} - \text{LF}_n$) after adding each subsequent error to the previous sentence. Changed text is shown in bold italics.

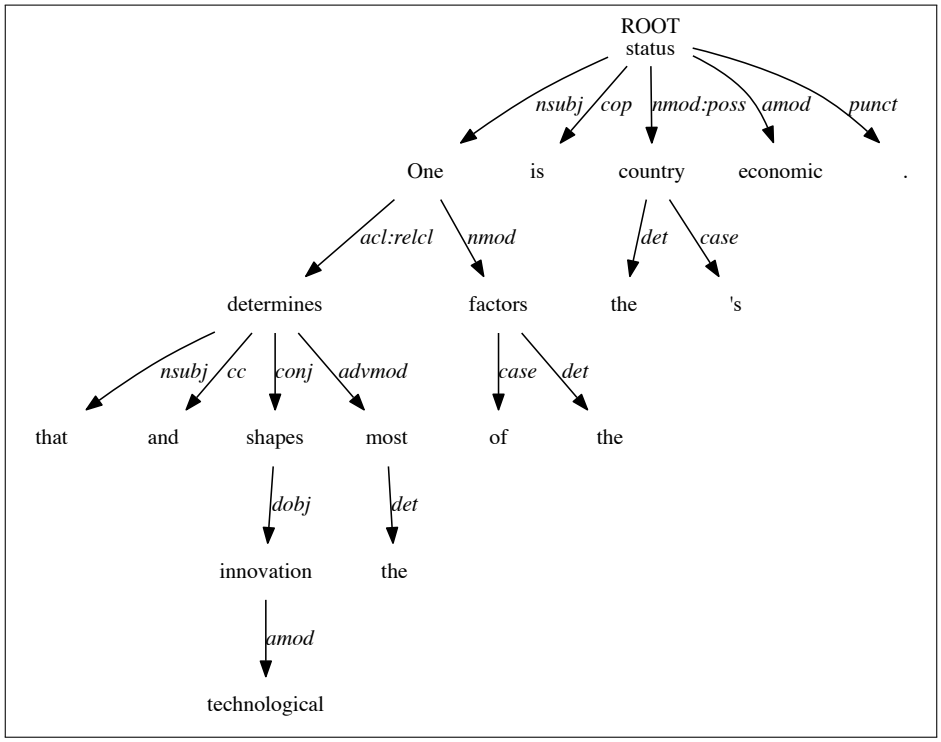


Figure 8: Dependency graph of the correct sentence in Table 2.

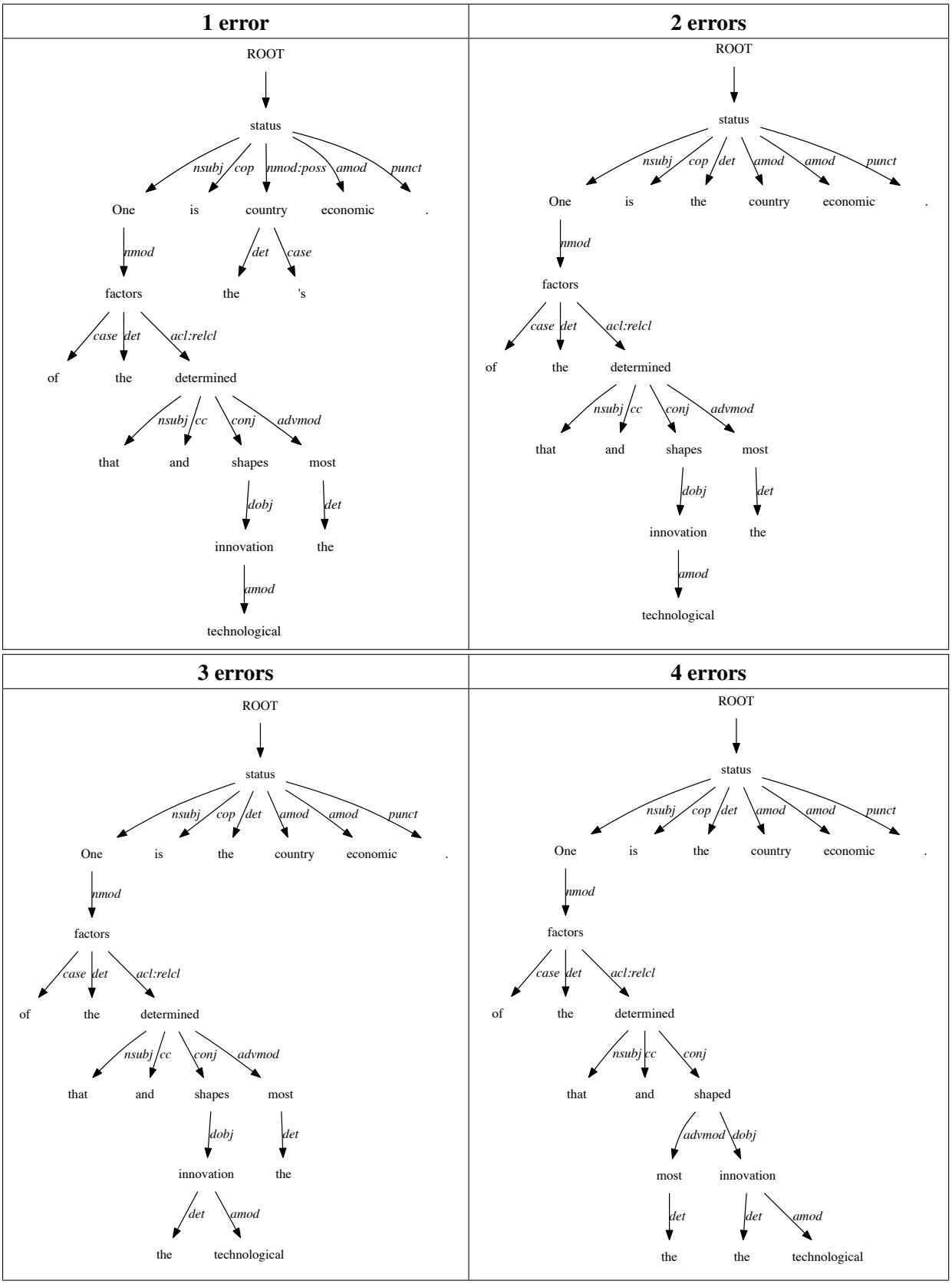


Figure 9: The dependency graphs of the sentence in Table 2 and Figure 8 after each error is introduced.

*Conference of the European Chapter of the Association for Computational Linguistics*, pages 116–126, Gothenburg, Sweden, April. Association for Computational Linguistics.

Jennifer Foster, Joachim Wagner, and Josef Van Genabith. 2008. Adapting a WSJ-trained parser to grammatically noisy text. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 221–224. Association for Computational Linguistics.

Jennifer Foster, Özlem Çetinolu, Joachim Wagner, and Josef van Genabith. 2011. Comparing the use of edited and unedited text in parser self-training. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 215–219, Dublin, Ireland, October. Association for Computational Linguistics.

Jennifer Foster. 2004. Parsing ungrammatical input: an evaluation procedure. In *Proceedings of Language Resources and Evaluation Conference (LREC)*.

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

Jennifer Foster. 2010. "cba to check the spelling": Investigating parser performance on discussion forum posts. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 381–384, Los Angeles, California, 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. Somerville, MA: Cascadilla Proceedings Project*.

Rasoul Kaljahi, Jennifer Foster, Johann Roturier, Corentin Ribeyre, Teresa Lynn, and Joseph Le Roux. 2015. Foreebank: Syntactic analysis of customer support forums. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

John Lee and Stephanie Seneff. 2008. Correcting misuse of verb forms. In *Proceedings of ACL-08: HLT*, pages 174–182, Columbus, Ohio, June. Association for Computational Linguistics.

Diane Nicholls. 2004. The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 Conference*, pages 572–581.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 49–56, Boston, Massachusetts, USA, May 6–May 7. Association for Computational Linguistics.

Ines Rehbein and Josef van Genabith. 2007. Evaluating evaluation measures. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA)*, pages 372–379.

Brian Roark, Mary Harper, Eugene Charniak, Bonnie Dorr, Mark Johnson, Jeremy G Kahn, Yang Liu, Mari Ostendorf, John Hale, Anna Krasnyanskaya, et al. 2006. SParseval: Evaluation metrics for parsing speech. In *Proceedings of Language Resources and Evaluation Conference (LREC)*.

Alla Rozovskaya and Dan Roth. 2010. Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 28–36. Association for Computational Linguistics.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics on Human Language Technologies: Short Papers*, pages 353–358, Uppsala, Sweden, July. Association for Computational Linguistics.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2011. Evaluating dependency parsing: Robust and heuristics-free cross-annotation evaluation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 385–396, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Joachim Wagner and Jennifer Foster. 2009. The effect of correcting grammatical errors on parse probabilities. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 176–179. Association for Computational Linguistics.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 180–189. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.