

Automated Writing Support Using Deep Linguistic Parsers

Luis Morgado da Costa[♣], Roger V P Winder[♡], Shu Yun Li[♡],
Benedict Christopher Lin,[♡] Joseph Mackinnon[♡], Francis Bond[♣]

♣ Linguistics and Multilingual Studies

♡ Language and Communication Centre

School of Humanities

Nanyang Technological University

lmorgado.dacosta@gmail.com

Abstract

This paper introduces a new web system that integrates English Grammatical Error Detection (GED) and course-specific stylistic guidelines to automatically review and provide feedback on student assignments. The system is being developed as a pedagogical tool for English Scientific Writing. It uses both general NLP methods and high precision parsers to check student assignments before they are submitted for grading. Instead of generalized error detection, our system aims to identify, with high precision, specific classes of problems that are known to be common among engineering students. Rather than correct the errors, our system generates constructive feedback to help students identify and correct them on their own. A preliminary evaluation of the system's *in-class* performance has shown measurable improvements in the quality of student assignments.

Keywords: technology-enhanced learning, blended learning, immediate feedback, grammar engineering, tertiary teaching

1. Introduction

Automated Grammar Error Detection (GED) and Correction (GEC) are tasks that have attracted some attention within the NLP community. This is especially true for English, where a myriad of shared-tasks periodically compare and attest the impact of the latest available technology. Some recent efforts in organizing shared-tasks within these topics include: the 2011 Helping Our Own (Dale and Kilgarriff, 2011, HOO) shared-task on GEC; the more focused 2012 HOO shared-task on Preposition and Determiner Error Correction (Dale et al., 2012); the 2013 and 2014 CoNLL shared-tasks on English GEC (Ng et al., 2013; Ng et al., 2014); the 2016 shared-task on Automated Evaluation of Scientific Writing, focusing on error detection (Daudaravicius et al., 2016, AESW); and, most recently, the 2019 shared-task on English GEC (Bryant et al., 2019).

Most of these tasks aim to test the ability to perform generalized GEC, and often include a large variety of common errors attested by learner corpora. While we agree that these efforts are of utmost importance for the field, this paper is best aligned with the spirit of the shared-task on AESW – where the focus is on the detection of issues (i.e. not strictly *errors*), and the goal is to assist authors in writing better academic papers.

As Daudaravicius (2015) rightfully describes, the task of assisting academic writing includes monitoring *language quality* in dimensions that go well beyond grammatical issues. These might include a variety of stylistic appropriateness checks that include, for example, word and sentence ambiguity, voice, word-choice, academic and discipline-specific terminology, etc.

As such, the system we describe here is not a generalized error detection system. Rather, it is a targeted contribution, focusing specifically on the needs of our engineering students in their English Technical Writing. Since the main goal of our system was for it to be used as a pedagogical tool, we did not aim to perform coverage on a wide range of error classes.

Instead, we started by focusing on error classes that would be helpful for the students and that we knew we could detect well. In addition, the checks that our system performs go beyond grammaticality, and include many stylistic checks recommended by our team of lecturers. While many of these checks would most likely be useful beyond our classroom, a few are likely to be specific to either our students' language backgrounds, or to our lecturers' sensitivity to certain aspects of style.

Our system differs from most other existing systems in another very important way. The fact that we wanted to use this system as a pedagogical tool had a direct impact on the granularity with which errors were considered. We want to be able to provide constructive feedback that is informative enough to allow students to understand the problem, to explore possible solutions and to decide on the best corrections on their own.

For example, instead of a generalized message '*something is wrong with the use of determiners in this phrase.*', whenever possible, we aim to provide specific feedback for specific classes of errors. For example, our system is able to differentiate problems related with determiners that include, among others: the omission of an article for single countable nouns; the use of indefinite articles with uncountable nouns; and the use of the wrong form of the indefinite article 'a/an'. This idea, which relates to the type of feedback we offer students, will be further discussed in Section 5.3.

One final aspect where our system differs significantly from most other recent systems is the fact that we choose to perform GED using linguistically inspired computational parsers (i.e. instead of statistical learning over large sets of labeled data). The reasons for this are many, and include both the precision and granularity of this GED. This choice also allows independence from these datasets, which are often skewed and impose some restrictions in the way GED can be learned. In addition, as it will be made clear in Section 2, the use of computational parsers enhanced with *mal-*

rules allows each ungrammatical sentence to be ambiguated to many possible intended meanings. Although this is not fully explored in the current version of this system, we hope to make use of this to further improve the pedagogical reach of this system in future versions.

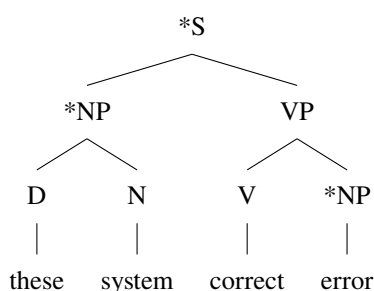
The rest of this paper will be structured as follows: Section 2 explains how we perform GED using computational parsers; Section 3 discusses the motivation behind the development of this system; Section 4 discusses the importance of learner corpora in our system’s design; Section 5 describes the system’s building blocks and usage flow; Section 6 provides an account of our preliminary evaluation experiment; and Section 7 concludes with a small discussion and pointers to future work.

2. Grammatical Error Detection using Computational Parsers

Using symbolic parsers, such as computational grammars, for GED or GEC has both advantages and disadvantages. The main disadvantage is, most definitely, coverage. Symbolic parsers take a long time to develop before being able to compete against statistical parses on coverage. When coverage is not an issue, however, symbolic parsers are often able to provide much higher quality and richer structure to language. Our system takes advantage of this benefit to perform error detection and select feedback based on a concept known as *mal-rules*.

The concept of *mal-rule* was first proposed by Schneider and McCoy (1998). These rules are used to extend descriptive grammars in order to allow specific ungrammatical phenomena, while reconstructing structures that were violated. Although the design of *mal-rules* is time consuming, they can enable fine-tuned error distinctions that statistical parsers would have a hard time dealing with. Consider example (1), below:

- (1) * *These system correct error.*



A descriptive grammar of English should reject (1) as a proper sentence. However, the decision of how to correct this sentence is not simple. Without context, at least four corrections (2 to 5) should be considered – but other options could also be considered. From a pedagogical point of view, each of these corrections should elicit different kinds of corrective feedback.

- (2) *These systems correct the error.*
 (3) *These systems correct errors.*
 (4) *This system corrects the error.*

- (5) *This system corrects errors.*

While dealing this ambiguity might seem daunting for some statistical systems, a few *mal-rules* would allow this sentence to be parsed while reconstructing all of the meanings shown above. This would require: a *mal-rule* that allows disagreement between nouns and determiners, so that both *this system* and *these systems* can be built; a *mal-rule* to either add a quantifier or change *error* into plural; and, finally, a rule that would allow subject-predicate disagreement, so that both *this system* and *these systems* can be taken as subjects.

Mal-rules can act on syntactic structures and on individual lexical items. A system using *mal-rules* can perform both error detection and error correction (since ungrammatical structures can be reconstructed). From a pedagogical point of view, each *mal-rule* activation can be converted into a constructive feedback message (e.g. *‘the subject and the verb of your sentence do not agree in number’*). Because of this, these rules are often named in a special way, or kept in different layers of the parser, such that grammatical and ungrammatical constructions can be differentiated.

Within implemented grammars, *mal-rules* can be selectively available for parsing but not for generation (Bender et al., 2004), or to allow one type of error but not others. If these implemented grammars produce a semantic representation, as is the case for the grammar used in this project, these *mal-rules* can be further designed to reconstruct the semantics of ungrammatical sentences in a way that allows the generation of corrected counterparts (Bender et al., 2004). Often, as shown above, the same ungrammatical sentence can trigger multiple parses, each reconstructing different semantics, so as to mimic different possible intended meanings behind an ungrammatical input. Morgado da Costa et al. (2016) provide a fuller account of how *mal-rules* and semantic reconstruction can be used in Computer Assisted Language Learning.

2.1. English Resource Grammar

The system presented in this paper uses, at its core, the English Resource Grammar (Copestake and Flickinger, 2000; Flickinger, 2000, ERG) as the main parser. The ERG is a symbolic grammar with a very large lexicon and wide coverage of syntactic phenomena. It has, in fact, been used by a different team in the 2016 AESW task (Flickinger et al., 2016). The team that used the ERG ranked second in the probabilistic estimation track, and fourth in the boolean decision track.

Of special interest for our system is the fact that this grammar has had substantial work to allow it to parse and identify both ungrammatical and *stylistically deprecated* sentences (Bender et al., 2004; Flickinger and Yu, 2013; Suppes et al., 2014). This is made available through a variety of methods, including *mal-rules* and the ability to define parsing strictness using root conditions of the tree (e.g. disable parsing of sentence fragments). In addition, the ERG provides a deep linguistic representation for each sentence (with detailed syntactic and semantic information), as opposed to a shallower representation such as an ngram language model. The information contained in this representation, such as verbal

mood, can be very useful for designing extra-grammatical checks.

3. Motivation

The motivation behind the development of our system is to assist undergraduate engineering students in a mandatory course on English Scientific Writing at Nanyang Technological University, in Singapore. This system was primarily conceived as a pedagogical tool, with the goal of alleviating some of the challenges our tutors have while coaching a cohort of over 2,000 undergraduate engineering students per year. These challenges revolve around correcting and providing timely, high quality feedback on student assignments, so students can learn from the feedback and iteratively improve their assignments throughout the duration of the course. Unfortunately, the size of the student cohort and the number of available tutors makes this a very difficult task. Without a system like the one we are presenting here, students rarely receive any feedback on their assignments before their grade is final, leaving them with little incentive to improve from the tutors' feedback after the course is over. Following this pedagogical mindset, the system's objective is not to correct errors. Instead, our system was designed to identify issues and provide constructive feedback that prompts students to consider whether corrections are needed. This allows students to have a more meaningful participation in the error correction process, learning while actively identifying errors and choosing from multiple ways which are often available to correct different classes of problems. In other words, the goal of this system is to provide immediate feedback to students on possible errors in syntax, style and lexis, encouraging independent critical thinking and inviting students to explore and evaluate possible solutions and decide on the best correction on their own.

4. Learner Corpora

Learner Corpora are made by the collection and analysis of language learner data, especially the process of labelling problems in written or spoken language (i.e. describing issues with a set of tags) (Granger, 2003). Learner Corpora are an essential component of multiple lines of research, including Second Language Teaching and Learning, Applied Linguistics and, perhaps not surprisingly, also to GED and GEC. The types of errors language learners make, as well as the frequency with which each error occur, are implicitly encoded in the labeling process – which is the necessary training data over which statistical GED and GEC systems learn.

Even though our system did not require statistical learning over labeled data, its development was profoundly inspired by previous work done on English Learner Corpora. Deserving notable mention are the NTU Corpus of Learner English (Winder et al., 2017), the NUS Corpus of Learner English (Dahlmeier et al., 2013) and the Cambridge Learner Corpus (Nicholls, 2003).

The process of selecting which checks to include in our system was a combination of firsthand experience of our tutors, data driven analysis based on the Learner Corpora mentioned above, and ease of implementation given the available tools. Fortunately, as mentioned above, the ERG al-

ready had the ability to detect many classes of common errors made by learners.

5. System Architecture

The system is fully developed on top of existing open-source platforms. At its core, it is a web system developed using Python and Flask, fully open-source, and scalable.

5.1. Submission

Submission to the system is done online, using any modern browser. Access to the system is done through a typical sign-in method, and is currently only available to students at our institution. After the signing in, upload instructions are presented to the student. For the time being, only documents with the type *docx* are accepted. This was a design decision based on the lower quality of text extraction available for other formats. Since sentence boundaries are extremely important in GED and GEC, extracting text from document formats like PDF would hurt the system's performance immensely. *docx* type documents are open, non-proprietary, and can be produced from most modern word processors (including Microsoft Word, Pages, Google Documents, LibreOffice Writer, etc).

Before submission, students are prompted with a choice of whether or not to release their assignments under a Creative Commons 0 license.¹ The way this prompt is shown and phrased follows the approved IRB protocols for this project. With this consent form, for each document uploaded, students make an informed decision about releasing their assignments for further research. The preference for a liberal license was to safeguard students' privacy. Using a license that required attribution would contradict students' right to anonymity, which was deemed more important. During this process, students are also informed that choosing not to release their assignment will not adversely affect them in any way.

5.2. Parsing and Checking

Once submitted, the *docx* files are converted to *xml* using Python's Mammoth² library. Sentence segmentation is done using the Python Natural Language Toolkit (Bird et al., 2009). Each sentence is then parsed using the ERG. This process uses PyDelphin³ which presents an easy-to-use API for, among other things, the efficient parser ACE.⁴ The current design of the system uses two parsers in parallel: one uses the standard ERG grammar which is designed to parse only sentences considered 'proper' English; the other uses a version of the ERG enhanced with *mal-rules* which is capable of parsing many kinds of ungrammatical sentences. For each sentence, the system first uses the standard ERG parser to try to produce an analysis. If this succeeds, then the sentence is considered grammatical. When the standard ERG parser fails, the ERG enhanced with *mal-rules* is used. If this second step is able to extract an analysis for the sentence, this means that we can determine which *mal-rules* were used to parse that sentence, and find out what

¹<https://creativecommons.org/public-domain/cc0/>

²<https://pypi.org/project/mammoth/>

³<https://github.com/delph-in/pydelphin>

⁴<http://moin.delph-in.net/AceTop>

was wrong with it. In the case more than one *mal-rule* was necessary to parse the sentence, the system interprets it as multiple errors in the same sentence (the discussion of (1), above, provides an example of this).

For the time being, our system uses only the top parse produced by *mal-rules* – while, in fact, multiple parses with different combinations of *mal-rules* are often available. This means that from the multiple available ways of correcting an ungrammatical sentence, we are only using the one the system considers most probable. We believe that, for our pedagogical use, this was useful enough. Students are the ones ultimately responsible for exploring and choosing the best way to address the problem. Even if the suggested correction does not carry the desired meaning, the system is still useful since it flags a likely problematic sentence that needs to be reviewed.

For some sentences neither of the two parsers is able to parse it, meaning that the sentence is likely ungrammatical but that there are no *mal-rules* specifically available to detect the errors the sentence contains. More rarely, it is a grammatical sentence but the ERG cannot parse it. Our system interprets this case as a generic *unspecified* error. This generic error might sometimes be useful, if the error was due to a typographical error, or if the student is able to determine what it wrong with it on their own. Our main goal, however, is to cluster these generic errors into meaningful categories and develop more *mal-rules* that are able to identify these classes of errors currently beyond the reach of our system.

After going through the parsing steps, all sentences also go through a collection of other checks written in Python. These include, for example, checking the sentence length, repeated words, and proper capitalization of words. Each of these checks is able to add to the number of problems identified in a given sentence, making it not at all unusual for a sentence to have more than one problem associated with it. In total, the system performs around 70 checks, using a mix of *mal-rules* and stylistic checks written in Python.

However, the process of checking errors for each sentence can be quite resource hungry, especially due to the parsing step. Each parser uses up to 2Gb of RAM memory per sentence (depending on sentence length and ambiguity). Because of this, and in order to allow multiple concurrent users, the system is currently designed to parse sentences in a serial fashion. This, of course, means that the students have to wait a couple of minutes before receiving feedback on their assignment. This design choice could easily be changed at a later stage, if the system were to be deployed in a large scalable server.

5.3. Feedback

Each check our system performs is tied to a feedback message, designed by our team of English Scientific Writing tutors. The feedback was written to help students understand the problem and to guide them through possible ways they can resolve it. Here are some examples of these feedback messages:

- *This sentence may have a verb which does not **agree** in person (e.g. 'I', 'you', 's/he') and number (singular/plural) with*

its subject: {{placeholder}}. Please check the sentence and ensure that the verb agrees with its subject.

- *You may be using 'a' (an indefinite article) before something that cannot be counted and does not have a plural form (an uncountable noun such as 'research'): {{placeholder}}. Please check your sentence for uncountable nouns and remove any 'a' that comes before them.*
- *This sentence may contain **subjective or informal words or expressions**: {{placeholder}}. You may want to replace these words and expressions with more formal and objective alternatives.*
- *This sentence is much **longer than the average sentence**. It may be difficult for readers to read the sentence and understand it after reading it once. There is also a higher risk of making grammar mistakes in such a long sentence. You may want to consider breaking up the sentence to make it easier for the reader to follow the text.*
- *You have used **'there'** in this sentence. Please check if it should be **'their'** instead and make the change if necessary.*

The feedback messages often include a placeholder that points the students to the word or words in the vicinity of the errors detected. These messages are also explicitly non-committal, as there might be multiple ways to correct an ungrammatical sentence, and because the system's performance is not perfect. As was mentioned above, we believe that many of these potential problems require the judgment of students as to whether they need to be addressed or not. A good example for this is a sentence flagged to be 'overly long'. Such sentences should, in principle, be checked (and possibly rewritten) but since the check is merely making a decision based on the number of words in the sentence, this may not be a serious problem for some sentences.

Finally, Figure 1 shows the end result produced by the system. The student's submitted document is converted to HTML (preserving images and most of the original styling of the document), and sentences that were considered problematic are highlighted in either red or yellow. These two colors are used to mark different levels of confidence and severity of the identified problems. For example, using overly casual or overly formal words or expressions triggers a yellow warning (since this is mostly a question of style and, in most contexts, not a very serious problem). In contrast, the lack of agreement between subject and predicate triggers a red level warning – most definitely requiring the student to change the sentence in some way.

5.4. Automatically tagged corpus

A useful side-product of this system is an automatically-tagged learner corpus, produced from users' submissions. As mentioned in Section 5.1, each user is invited to release their submissions under a public domain license. From our experience, between 40% and 50% of our students choose to release their assignments – which are then automatically tagged and stored by our system. And while we understand that this learner corpus is biased by the ability of our system to detect only certain classes of errors, we believe this will be a useful dataset for the research community. We are currently in the process of anonymizing these assignments, and plan to release an automatically tagged learner corpus from this data.

Anti Food Deserta

Background:
 In the recent years, Singapore's food wastage rate shoots rapidly. In all domains of food production and consumption, wastage of food has grown significantly. Article by food waste republic shows that in 2011, each Singaporean can generate 130kg of food waste per year, out of 0.68 million tons of food waste being thrown each year, only 10% could be recycled. [1].

Problem:
 This phenomenon raised a problem which maybe much neglected in Singapore, the awareness of wasting food is dropping. One big wastage source would be from the cosmetic filtering of food. In this sentence, you have used the passive verb phrase 'is/are + verb-ed': on in markets to filter food and this generates majority of food wastage. Also the awareness is falling, espec is. Please check whether you intended to use 'has/have + verb-ed' instead; I feel the urgency to address this problem as its negative effect is now impacting the current society and might even have a longer lasting negative effect in the future. Thus Anti deserta, meaning anti waste in Latin, is been used as the title.

Solution:
 To control the growth of this issue, I propose to setup a convenient way to distribute the wasted edible food. It requires combined effort from organizations, such as Disabled People's Association (DPA), fighting for the welfare of the elderly, disabled and the poor. Often, they are facing problem of food shortage. Instead of wasting food by discarding them, a more reasonable way would be distributing them to these needy groups. These organizations can set up centers all around Singapore to ease the collection of excess food. On the other hand, an app will be created for the convenience of notifying these centers for food collection. This provides a direct interaction between the food owners and the centers, curbing the food wasted from cosmetic filtering as now there is a better way for producers to make use of them. At such it would reduce the amount of edible food wasted daily and instead, put into good use to aid the needy. This can be beneficial especially within short term, while lowering the food wastage, it also provided the buffer time for other possible longer term solution to take effect.

Benefits:
 The immediate benefited ones would be the needy groups, directly solving their food shortage. This also cuts burden on NEA's pilot project reported in Channel News Asia [3], in the effort to reduce inedible food wastage.

Implementation:

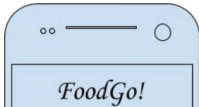


Figure 1: Online Error Detection System - Feedback Example

Rank	Label	Freq.
1	No Parse (unspecified problem)	14,834
2	Use of instructions/commands	4,032
3	Overly long sentences	3,727
4	Singular nouns without specifiers (e.g. article or determiner)	3,443
5	Use of first or second person singular pronouns	2,485
6	Repeated words	1,238
7	Use of informal words or expressions	942
8	Use of verbs that do not agree with their subjects	909
9	Use of questions	894
10	Use of contractions	372
11	Omission of the definite article 'the'	169
12	Use of indefinite articles with uncountable nouns	150
13	Use of comma splicing	126
14	Incorrect use of the verb form 'are'	126
15	Missing, inappropriate or unnecessary modal	125
16	Use of singular nouns with plural determiners	118
17	Incorrect use of the verb form 'is'	71
18	Use of plural with mass nouns	61
19	Use of formal or archaic words or expressions	39
20	Use of the wrong form of the indefinite article 'a/an'	29

Table 1: Frequency of the top 20 classes of errors detected by the system

6. System Evaluation

The main goal of the system is to improve student writing. We therefore tested it with a cohort of students attending a course entitled *Engineering Communication I*, taught at our university.

Since this system was not designed to compete with generalized GEC and GED systems, we decided it was not our first priority to test its performance on the publicly available datasets shared-tasks often use. In addition, (Flickinger and Yu, 2013) have shown that at least some of these datasets are biased towards certain classes of errors statistical systems are able to catch, and precision parsers such as the ERG are able to find many instances of errors not included in the original test-sets.

Our experiment was set up to measure if the system could

produce a positive impact on students' language use. One of the written assignments for this course invites students to write a technical proposal that describes an engineering solution to a real life problem, using a fixed document structure consisting of: background, problem, solution, benefits, implementation, budget and conclusion. The assignment was written in pairs, and had a limit of 800 words.

Students had a previously established deadline for submitting this assignment and, until the date of submission, were unaware of the existence of this system. After turning in their assignments, the full cohort of 1,855 students received an email informing them of the existence of a new system designed to give them feedback on probable grammatical and stylistic problems in their writing. Students were given one extra week to use this system and edit their assignments. There was no limit to the number of times an assignment

could be uploaded, and students were encouraged to use the system as much as they wanted, with the goal of improving their assignments.

A second deadline was set up for exactly one week after the first deadline, and even though using the system was optional, all assignments needed to be turned in a second time. This was done to encourage (though not force) the use of the system. At the end of this extra week, the system had received 2,581 submissions, from 798 pairs (i.e. roughly 86% participation). In this experiment, about 55% of the submissions agreed to release their assignment for future research.

In total, the system identified 34,141 problems spread through all 2,581 submissions. Table 1 shows the top 20 classes of errors detected by the system, with their respective frequencies. It is, nevertheless, important to note that the distribution was somewhat asymmetrical – both concerning the number of submissions per pair, and the number of errors detected in each assignment. Some pairs submitted the same assignment as many as 36 times while the average number of submissions was between 3 and 4. Also, even though the average number of problems found in each assignment was just over 13, some submissions had over 100 problems identified. This asymmetry reflects a fairly mixed sample of students enrolled in this course – many of them with native or near-native levels of English proficiency, and the rest distributed along lower levels of proficiency.

It is also important to note that Table 1 shows the sum frequencies for all submissions, which may include some repetition given that assignments could be submitted multiple times. We made this decision for a simple reason: as students were encouraged to correct their mistakes between submissions, different submissions of the same assignment were expected to include different errors. These might include failed attempts to correct a specific error, or new errors introduced by solving certain problems relating to style (e.g. long sentences, pronoun usage, etc.). The main goal of Table 1 is to provide a broad comparison between different classes of errors. The question concerning the best practices when converting our data into a Learner Corpus (mentioned in Section 5.4) is still under consideration.

Another aspect that needs to be addressed is the frequency of the error class “No Parse (unspecified problem)” – as it was, by far, the most frequent error label reported by our system. After conducting a focused error analysis on this label, we found that around 62% of these errors were false positives. The first version of our system (used in this experiment) was unable to detect the document structure. As such, the system was used in portions of the document that it should have ignored: e.g., references, mid-sentence citations, section headings, legends (etc.). As the development of the system continues, this will soon stop being a problem as we are now working to properly ignore these structures. Students were made aware of this problem, and were told to ignore such errors when it happened in structures that should not be considered full sentences. The remainder of this category ($\approx 38\%$) were errors our system was not yet able to detect. As this first version of our system focused only on errors known to be frequent among our students, we were not expecting to be able to correctly diagnose all

errors. As we keep improving the system, this error label will slowly decrease. Concerning the decision to highlight this error to students, we believe that even though it might not be very informative, it can still be helpful to students who can identify and correct the error by themselves. In case students are not able to figure out how to solve these undiagnosed problems on their own, they can ask for the help of a peer or a tutor.

6.1. Classroom Evaluation

In order to determine if the system had a positive impact on student language use, we carried out a paired-blind review of pre- and post-system submissions. We randomly chose 108 assignments from the pool of those where (i) the students used the system and (ii) there were at least six fewer errors in the final submission compared to the original submission. This allowed us to examine submissions where the students had taken advantage of the system.

A group of four experienced tutors evaluated both pre- and post-system submissions of these 108 assignments. Three assignments were unanimously removed from this sample, because the contents of the assignments differed too much, and were not deemed comparable – all other 105 assignments were deemed comparable, differing only in style or sentence structure, but not in content. Each of the remaining 105 assignments were given to two different tutors for comparison. Each assignment had two unidentified versions – an original version (i.e., the first submission to the system), and a post-system version of the same assignment (i.e., the last submission to the system). As grading was too subjective, tutors were asked, whenever possible, to choose the *best* document, without knowing which document corresponded to which version. If choosing the best document was not possible (i.e. both documents showed similar amount of problems), this meant that using the system had no measurable impact on the student’s writing. When the post-system version of the document was selected, this was interpreted as a positive impact on the student’s writing. Finally, the system was deemed to have a negative impact on the student’s writing whenever the original document was selected instead.

This experiment determined that using this system had positive impact on students’ language about 84% of the time (the system showed negative results in less than 2% of cases, and had no perceived impact about 14% of the time). These numbers are promising, especially considering that the system is still in an early proof-of-concept stage, and that much can still be done to improve its performance.

However, we would like to make some notes on the way our results can be interpreted. One could argue that just by extending the deadline for a week would grant students the ability to improve their assignments – whether the system existed or not – and that’s not false. All students had the opportunity to revise their assignments, and all students had to turn-in their assignments a second time. However, as mentioned above, our evaluation selected only assignments from students who actively engaged the system during this extra week. This was done by a quantifiable decrease in detected errors between the first and the last submission to the system. Because of this, we feel confident that the improve-

ment on the quality of writing was at least in part due to the use of our system.

Another important point to take into consideration is that we do not claim our system will have a positive impact on 84% of all of its users. We are aware that our experiment selected a subset between the cases where improvement was possible (i.e., errors were detected by the system) and action was taken (i.e., these errors were addressed by the students). The numbers we report are, therefore, measuring only the possible impact that our system can have in the right context. As a pedagogical tool, it is expected that this system will be more useful to struggling students – and this is what our results seem to corroborate.

In addition to the quantitative results, a small survey included as part of the end of the course survey revealed that the majority of students were not only very aware of the benefits that the system provided (see Fig. 2), but that students were also interested in using this system in other courses and assignments (see Fig. 3). In summary, this shows that the system was well received by the students, in addition to improving their language use. This survey also contained a few other open ended questions concerning possible improvements to the system and a free comments section that have not yet been analysed with methodological detail.

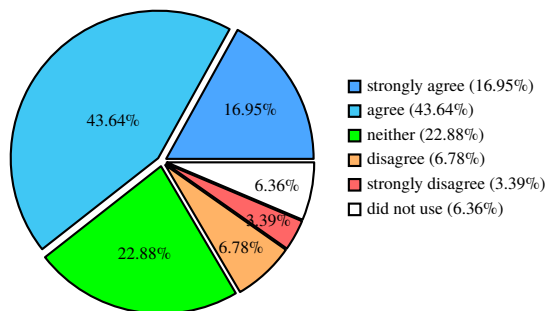


Figure 2: ‘I found the online error detection tool useful.’ (n=236)

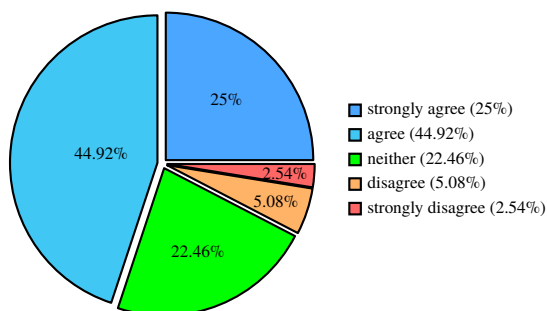


Figure 3: ‘I would like to use the online error detection tool for other courses and assignments.’ (n=236)

7. Discussion and Future Work

This paper describes the release of a system that bridges English Grammatical Error Detection (GED) and course-specific stylistic guidelines to automatically review and pro-

vide feedback on student assignments. This system is part of a larger ongoing project that is exploring how computational parsers and *mal-rules* can be used in various pedagogical settings. Even though this system is still very much in development, it has already been tested in a classroom setting, and proven to be useful in assisting students to improve their Scientific Writing.

The next stages of development for this system include adding support for other input formats, namely \LaTeX – which is widely used in Academic Writing, specially in the fields of engineering. There are multiple tools available for extracting text from \LaTeX , and this addition would make our system more valuable outside the classroom.

Another area that we would like to dedicate some effort to is in the improvement of how we deal with the overall document structure. Academic writing has some elements that are distinct from other genres of writing (e.g. section headers, mathematical expressions, citations, captions and references). In order to provide the best possible feedback, it is important to identify these different elements. It is not appropriate, for example, to perform grammatical checks on bibliographic references (although our prototype did). For this reason, we would like to dedicate some time to improving this aspect of the system – providing only feedback that is relevant to specific elements, and ignoring elements that are outside the scope of our system.

Following the discussion introduced in Section 2, in parallel with the development of more *mal-rules* to increase the type of errors our system can detect, we would also like to take more advantage of the ambiguity generated from *mal-rules* to offer multiple options to correct the same sentence. As mentioned above, it is often the case that multiple reconstructed parses are available when we parse an ungrammatical sentence. Thus far, we have chosen one of the available parses to be the source of the feedback (i.e. assuming this parse was the intended meaning of the sentence). However, this is not strictly necessary. We would like to explore the ability to provide different sets of feedback for the same sentence, allowing the users to see more than one way of correcting the same problematic sentence. Towards this end, we have started to build a treebank with the *mal-rule* enhanced grammar, and will soon be able to train a parse ranking model using *mal-rules*, and rank multiple possible corrections of the same sentence in order of likelihood.

Finally, we are currently performing a detailed analysis of the appropriateness of each error check and feedback message elicited by it. Despite the fact that students showed interest in using this system outside of class, many students voiced their concerns about the clarity of the feedback messages. We are aware that some feedback messages currently use linguistic terminology that might not be clear to every student (especially when their native language is not English). Because of this, we would like to keep improving the feedback provided by our system so that it is more accessible to all students.

Release Notes

The first version of this system has been released on GitHub⁵ under the MIT license.⁶ The automatically collected corpus will also be released under a Creative Commons Attribution 4.0 International License (CC BY 4.0).⁷

8. Acknowledgements

The work presented in this paper has adhered to all IRB protocols requested by our home institution. This research is supported by Nanyang Technological University's EdEx Teaching and Learning Grant administered through the Teaching, Learning and Pedagogy Division (TLPD) and the MOE TRF grant *Syntactic Well-Formedness Diagnosis and Error-Based Coaching in Computer Assisted Language Learning using Machine Translation Technology*.

9. Bibliographical References

- Bender, E. M., Flickinger, D., Oepen, S., Walsh, A., and Baldwin, T. (2004). Arboretum: Using a precision grammar for grammar checking in CALL. In *INTIL/ICALL Symposium 2004*.
- Bird, S., Klein, E., and Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.
- Bryant, C., Felice, M., Andersen, Ø. E., and Briscoe, T. (2019). The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75.
- Copestake, A. and Flickinger, D. (2000). An open source grammar development environment and broad-coverage English grammar using HPSG. In *In proceedings of LREC 2000*, pages 591–600.
- Dahlmeier, D., Ng, H. T., and Wu, S. M. (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.
- Dale, R. and Kilgarriff, A. (2011). Helping our own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 242–249. Association for Computational Linguistics.
- Dale, R., Anisimoff, I., and Narroway, G. (2012). HOO 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 54–62. Association for Computational Linguistics.
- Daudaravicius, V., Banchs, R. E., Volodina, E., and Napoles, C. (2016). A report on the automatic evaluation of scientific writing shared task. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 53–62.
- Daudaravicius, V. (2015). Automated evaluation of scientific writing: AESW shared task proposal. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 56–63.
- Flickinger, D. and Yu, J. (2013). Toward more precision in correction of grammatical errors. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 68–73.
- Flickinger, D., Goodman, M., and Packard, W. (2016). UW-stanford system description for AESW 2016 shared task on grammatical error detection. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 105–111.
- Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(01):15–28.
- Granger, S. (2003). The International Corpus of Learner English: a new resource for foreign language learning and teaching and second language acquisition research. *TESOL Quarterly*, 37(3):538–546.
- Morgado da Costa, L., Bond, F., and Xiaoling, H. (2016). Syntactic well-formedness diagnosis and error-based coaching in computer assisted language learning using machine translation. In *Proceedings of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA2016)*, pages 107–116, Osaka, Japan. The COLING 2016 Organizing Committee.
- Ng, H. T., Wu, S. M., Wu, Y., Hadiwinoto, C., and Tetreault, J. (2013). The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria, August. Association for Computational Linguistics.
- Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C. (2014). The CoNLL-2014 shared task on grammatical error correction. In *CoNLL Shared Task*, pages 1–14.
- Nicholls, D. (2003). The Cambridge learner corpus: Error coding and analysis for lexicography and elt. In *Proceedings of the Corpus Linguistics 2003 conference*, volume 16, pages 572–581.
- Schneider, D. and McCoy, K. F. (1998). Recognizing syntactic errors in the writing of second language learners. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 2, COLING '98*, pages 1198–1204, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Suppes, P., Liang, T., Macken, E. E., and Flickinger, D. P. (2014). Positive technological and negative pre-test-score effects in a four-year assessment of low socioeconomic status k-8 student learning in computer-based math and language arts courses. *Computers & Education*, 71:23–32.
- Winder, R. V. P., MacKinnon, J., Li, S. Y., Lin, B., Heah, C., Morgado da Costa, L., Kuribayashi, T., and Bond, F. (2017). NTUCLE: Developing a corpus of learner English to provide writing support for engineering students.

⁵<https://github.com/lmorgadodacosta/LCC-AssignmentChecker>

⁶<https://opensource.org/licenses/MIT>

⁷<https://creativecommons.org/licenses/by/4.0/>

In *Proceedings of the 4th Workshop on NLP Techniques for Educational Applications (NLPTEA 2017)*, Taipei, Taiwan. (IJCNLP 2017 Workshop).