NON-NATIVE TEXT ANALYSIS WITH SYNTACTIC DIFF, A GENERAL COMPARATIVE TEXT MINING FRAMEWORK

BY

SEAN ALEXANDER MASSUNG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2015

Urbana, Illinois

Advisor:

Professor ChengXiang Zhai

ABSTRACT

Non-native speakers of English far outnumber native speakers; English is the main language of books, newspapers, airports, air-traffic control, international business, academic conferences, science, technology, diplomacy, sports, international competitions, pop music, and advertising [1]. Online education in the form of MOOCs (massive online open courses) is also primarily in English—even teaching English. This creates enormous amounts of text written by non-native speakers, which in turn generates a need for grammar correction and analysis. Even aside from MOOCs, the number of English learners only in Asia alone is in the tens of millions.

In response to this powerful motivation, we describe SYNTACTICDIFF, a novel edit-based method for transforming sequences of words given a reference corpus. These transformations can be used directly or can be employed as features to represent text data in a wide variety of text mining scenarios. As case studies, we apply SYNTACTICDIFF to four quite different tasks in non-native text analysis and show its benefit in each case.

In the first task, we use weighted word edits with likelihood scoring for grammatical error correction. Our method is compared against systems in a grammar correction shared task, and we find that SYNTACTICDIFF edits perform comparably while being much more general than the other methods. The second task is native language identification: a classification problem predicting the native language of a student writer based on English essays. We represent documents as vectors of edits, and show that a combination of unigram words and SYNTACTICDIFF edits outperforms each representation individually. The third task is fluency scoring, in which we see if the manually categorized fluency levels of English students can be modeled by SYNTACTICDIFF features. In the fourth task, we create clusters of student essays with similar errors via topic modeling, and find that the interpretability is significantly higher than an n-gram words approach.

SYNTACTICDIFF is highly customizable and able to capture syntactic differences from a reference corpus at the sentence, document, and subcorpus levels. This enables both a rich translation method and feature representation for many text mining tasks that deal with word usage and syntax beyond bag-of-words. In particular, this thesis focuses on non-native text analysis applications, though SYNTACTICDIFF is not at all limited to that domain.

ACKNOWLEDGMENTS

I am most grateful to my advisor, Professor ChengXiang Zhai. He is an excellent mentor, and without his help and supportive guidance, my work during my last year of undergraduate study and two years of graduate school would not be what it is today. It is with his generous time and wise insights that I was able to complete the work that appears in this thesis.

I would also like to thank Professor Julia Hockenmaier and Dr. Cinda Heeren for their supportive roles in my research and academic life. I've also received very useful comments about my papers and presentations from Professor Jiawei Han, Professor Dan Roth, and Professor Hari Sundaram.

My colleagues in graduate school have also generously shared motivation, encouragement, and fruitful discussion. Special thanks to Jason Cho, Chase Geigle, and Urvashi Khandelwal, among many others.

Finally, I'd like to extend my gratitude towards both my current and future families, and especially to my fiancée, Kai Nemoto.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

By the year 2020, the British Council [1] estimates that there will be two billion English language learners. Some learn in the classroom; some learn online. Some may even learn through their phone or in an online class. Regardless of the medium, computational tools to enhance this educational experience will be valuable. Automatic scoring of essays—not only for grammar, but also fluency—would contribute greatly to second-language learners' understanding. User personalization for online services (including search engines and social networks) would benefit from improved user profiling. More relevant books or news articles could be recommended if the user's background and competency of English were known.

Due to these many motivating examples, research in *non-native text analysis* has prospered. This field encompasses any textual task that deals with words written in a language other than the writer's native tongue. We call the native language L1 and the second, learned language L2. Throughout this survey, we will usually assume that L2 is English, though most (but not all) techniques discussed in this survey are general and could function with any pair of L1 and L2.

In this thesis, we start by providing a brief survey of existing work on non-native text analysis. First, we discuss non-native grammatical error correction—finding and modifying text to fix errors or to make it sound more fluent. Second, an introduction to native language identification: determining the native language of an author based on text in the second language. Then, we take a brief look at two emerging fields: native fluency scoring and text simplification for non-native speakers. This concludes the literature review component of this thesis, and opens up discussion on our novel method, SYNTACTICDIFF.

We propose a method based on comparative text mining that is able to be adapted for use in all of the described non-native text analysis scenarios. However, the reader should note that SYNTACTICDIFF is a quite general method, and

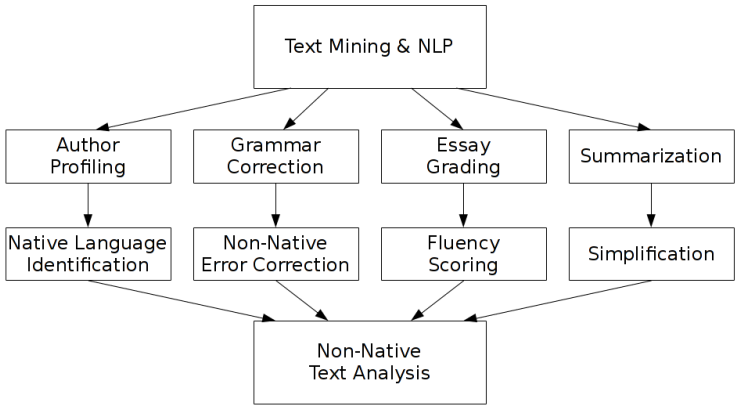

Figure 1.1: Dichotomy of non-native text analysis as part of the general text mining and NLP domain.

we simply decided to chose non-native text analysis as the avenue of introduction and explanation.

1.1 Scope

The current work on non-native text analysis generally falls into the four categories mentioned above:

1. **Native language identification** (NLI): classifying L1 based on text written in L2. Techniques can be categorized into **feature-based** (using a classifier) or **likelihood-based** (using a probabilistic model).

2. **Non-native Grammatical Error Correction** (GEC): detecting and correcting grammatical errors in L2 text. Techniques can be categorized into **targeted** (correcting specific errors) or **general** (correcting all errors).

3. **Fluency Scoring**: given L2 text, how close to native does it appear?

4. **Text simplification**: providing a better experience for users interacting with text in their L2. Techniques are much more varied in this field.

These four areas can be regarded as special cases of four more general categories of text mining and NLP as shown in Fig 1.1.

An excellent overview of grammatical error correction with a focus on non-native learners can be found in [2]. This short book is a concise collection on the topic and consists of many recent advances since 2010. If the reader wishes to delve into more detail in this subtopic, we suggest referencing their work, whereas this thesis features a broader outline and is thus not able to go into as much depth in one particular area.

Spelling correction described in [3] is a potentially relevant task, though we choose to focus on the grammar correction aspect instead. This is not to say that spelling features don't play a role in (*e.g.*) NLI; rather, the method of spelling correction is not relevant to non-native text analysis. Correcting speech is also an important issue: besides the obvious speech recognition challenges in computer science and electrical engineering, speakers tend to voice their words much differently than when writing. Therefore, we do not include any investigation into these works in this thesis.

Shared tasks provide a common goal and dataset to a wide array of researchers. This enables quick and accurate comparison of different methods, while simultaneously increasing interest and producing exposure for the problem at hand. Thus, we encourage the reader to explore two relevant shared tasks that deal with non-native writers of English. The first is a shared task in native language identification, held in 2013 [4]. The second is a task in grammatical error correction, using text written by non-native English speakers [5].

Author profiling, authorship attribution, and plagiarism detection at PAN 2014 with [6] are also related and we consider them related works since there is no specific component of non-native analysis. Shared tasks have even started to evolve for L2s besides English. For example, the Techniques for Educational Applications Workshop had a shared task on GEC for Chinese as a foreign language organized by [7].

1.2 Organization

We now review the structure of this thesis. Chapter 2 outlines the field of non-native text analysis which consists of non-native grammar correction, native language identification, fluency scoring, and summarization. Then, chapter 3 outlines SYNTACTICDIFF our proposed general comparative text mining framework. We explain in detail the motivation and mathematical implemen-

tation. Chapter 4 uses the SYNTACTICDIFF framework to perform analysis by running experiments on the four non-native text mining areas discussed previously. Next, chapter 5 is a discussion on the generality of SYNTACTICDIFF as well as suggestions for improvements via future work. Finally, chapter 6 completes the thesis in summary and conclusions.

CHAPTER 2

NON-NATIVE TEXT ANALYSIS

This chapter outlines the main problems in the quickly emerging field of non-native text analysis. In each section, we give an overview of methods used to solve each problem.

We hope this gives the reader some context in which to appreciate the generality of SYNTACTICDIFF, as each approach below is specifically tailored towards one task.

2.1 Non-Native Grammar Correction

Grammatical error correction takes as input a potentially malformed (ungrammatical) sentence in L2 and performs some transformation of the text that results in a more fluent output. Some examples of common error categories taken from a shared task in grammatical error correction are below:

- **Article or determiner**: "*In late nineteenth century, there was a severe air crash happening at Miami international airport.*" Correction: replace *late* with *the late*.
- **Preposition**: "*Also tracking people is very dangerous if it has been controlled by bad men in a not good purpose.* Correction: replace *in* with *for*.
- **Noun number**: "*I think such powerful device shall not be made easily available.*" Correction: replace *device* with *devices*.
- **Verb form**: "*However, it is an achievement as it is an indication that our society is progressed well and people are living in better conditions.*" Correction: replace *progressed* with *progressing*.

- **Subject-verb agreement**: "*People still prefers to bear the risk and allow their pets to have maximum freedom.*" Correction: replace *prefers* with *prefer*.

Correcting machine translated text is a related issue, but we do not discuss it here; instead, please see [8] or [9]. Fig 2.1 compares the different methods discussed in this section.

Some grammar correction methods are targeted towards a very specific subset of errors, often categorized by the corpus; others attempt to solve more general errors concerning word sense or collocations. Evaluation for grammar correction is much more varied and unstandardized in comparison to the configurations from NLI as we will see in the next section. Which non-native corpus is used also dictates the types of errors that can be corrected.

Lee and Seneff [10] train a trigram language model on a lattice of alternatives, where "alternatives" are prepositions, articles, and auxiliaries that may or may not occur between words in the original text. For example, the sentence *I want flight Monday* can be corrected by inserting two tokens as such: *I want a flight on Monday*. Their algorithm first strips all such alternatives from the original sentence. So far, this is not much different from the article and preposition corrections. However, they additionally change each remaining word in the input sentence to be a set of related words to the base form: $want \rightarrow \{want, wants, wanted, wanting\}$. Their language model then outputs the k-best candidates. Next, these candidates are given to a PCFG and reranked. The final output is the top-ranked sentence from the PCFG. Across all experiments, they found that reranking the language model candidates significantly increased the F measure.

Brockett, Dolan, and Gamon [11] used statistical machine translation to translate non-native speech into native speech. They first identified common errors in a Chinese learner's English corpus, and used regular expressions to convert target English from Reuters articles into ungrammatical English. This approach is very similar to one by Rozovskaya and Roth [12], which is an error insertion method; instead of articles, it uses generic grammatical errors such as *I knew many informations about Christmas*. It is unclear how extensive or comprehensive the regular expressions were to introduce grammatical errors since no examples are given or referenced. Additionally, the regular expressions uniformly distribute errors throughout the source language, which is not

Paper	Method	Target
Lee and Seneff 2006	LM with PCFG scoring*	articles, prepositions, and word forms
Brockett et al. 2006	machine translation*	specific injected errors
West et al. 2011	bilingual random walk+	word sense
Dahlmeier and Ng 2011a	machine translation+	collocation errors
Dahlmeier and Ng 2011b	structure optimization*	articles/prepositions

Figure 2.1: Comparison of GEC strategies. * indicates targeted approaches and + indicates general approaches.

how errors naturally occur. Despite this, they did find that their processed training data was able to be used in the MT system to successfully correct errors. This shows that—given a source model—statistical machine translation may be used to correct grammatical errors.

West, Park, and Levy [13] use bilingual random walks between L1 and L2 word senses. For example, on one side of a bipartite graph are L1 words. There are connections from a word $w \in$ L1 to a word $w' \in$ L2 if a w could be translated into w'. w could be the English word *head*, and be translated into a physical head, head of an organization, or the verb *to head*. This model was used to correct non-native sounding phrases such as *entire stranger* to the more natural *complete stranger*. This bipartite graph was combined with a language model to correct non-native sentences. In these experiments, the native language was Korean. Evaluation was performed with Amazon Mechanical Turk [1] where workers chose between the corrected sentence and the original sentence. Results were not strongly positive, since sometimes the corrected errors changed the meaning of the sentence or made it ungrammatical. In future work the authors suggest using a richer probabilistic model such as a PCFG.

Dahlmeier and Ng [14] use the NUCLE corpus to find and correct collocation errors via machine translation. Here, a collocation is a phrase commonly used by native speakers. The authors propose that when a writer mentally translates from L1 to L2, some unnatural phrases result due to word choice. They give an example, "*I like to look movies*" that might be written by a native Chinese speaker since *watch* and *look* are very similar in the L1. It would be possible to correct this to the more grammatical "*I like to look at movies*", but it still doesn't

[1] `https://www.mturk.com/mturk/welcome`

sound natural. Instead, *look* is replaced by *watch*, resulting in the more fluent collocation *watch movies*. For their experiments, they assume the unnatural collocations have already been identified; this mimics a system where a user may ask for improvement suggestions for a snippet of writing. They train a statistical machine translation model on a parallel Chinese-English corpus to correct collocation errors in the NUCLE corpus. A log-linear model was used to score the candidate phrases which allows additional spelling, homophone, and synonym features to be incorporated. They evaluated their method as a retrieval task, where they returned the top k suggestions to fix each collocation error. Two native-English speakers judged results from five hundred corrections with good rater agreement. Finally, they performed an analysis of errors and found that the main reason top-ranked phrases were not correct was due to out-of-vocabulary words.

Dahlmeier and Ng [15] also introduce an alternating structure optimization (ASO) approach to GEC. In short, ASO is able to leverage a common structure between multiple related problems; see [16] for a more detailed description. In this case, the related problems are selection (find features from native text) and correction (fix the errors in non-native text). Targets were article and preposition errors, again using the NUCLE corpus. It was shown that ASO significantly outperformed a simple linear classifier as well as two unnamed commercial grammar checkers. Features included part-of-speech tags, hypernyms from WordNet, named entities, and shallow parsing tags.

In conclusion, we saw a variety of techniques for correcting grammatical errors, which can be categorized into targeted *vs* general strategies. Targeted strategies focus on errors of only specific types (such as the shared task) which general correctors try to improve the overall fluency of the L2 text.

2.2 Native Language Identification

We now transition from non-native grammar correction to native language identification. NLI usually relies on classification, but also consists of other components that are able to capture a deeper syntactic meaning (such as dependencies or language modeling).

NLI is usually the first step in any second language error correction or author profiling system. Identifying the native language of an anonymous text was

first popularized by Koppel et al. [17]. Brooke and Hirst [18] do an extensive survey of NLI feature efficacy, and develop a robust model that works well when used across corpora. NLI tasks are most commonly evaluated solely on a small learner corpus usually consisting of student essays. It was previously thought that lexical features would be biased or overfit towards essay topics, but a cross-corpus evaluation showed that this was not the case [18].

For an in-depth discussion of the related task of authorship attribution, we recommend the reader consult [19]. Many techniques common to authorship attribution and author profiling are also relevant to NLI.

The International Corpus of Learner English (ICLE) [20] was an early popular dataset to evaluate NLI tasks, especially using a subset of five European languages partitioned by Koppel et al. A table has been created (Fig 2.2) to portray the summary described in [18]. In addition to the results displayed in the table, Wong and Dras also attempted to perform dimensionality reduction with LDA by [21] as feature generation; however, this was not a successful method.

Techniques for NLI can be categorized into two methods: feature-based and likelihood-based. Feature-based methods rely on informative features derived from the text and are fed to standard machine learning algorithms (usually SVM or MaxEnt). Likelihood-based methods learn a probabilistic model (usually a grammar or language model) for each L1 and assign a label based on the maximum likelihood model. We now move onto specific techniques applied in NLI.

The first feature-based method used by Tsur and Rappoport [22] found that incredibly simple top two hundred frequent bigram character features fed to SVM led to 66% accuracy on the five native languages. They claimed that word choice of non-native speakers is influenced by the phonology of their native language (as evidenced by the effectiveness of the character features). This is compared to a unigram words baseline which achieved only 47% accuracy. They finally hypothesized that using a spoken-language corpus would achieve even stronger results favoring character bigrams since conscious effort put into speaking words is much less than writing them.

NLI has also been approached through contrastive analysis from Wong and Dras [23]: the idea that errors in text are influenced by the native language of the author. They investigated three error types as features: subject-verb disagreement, noun-number disagreement, and determiner misuse. These error

Paper	Method	Accuracy
(Tsur and Rappoport 2007)	character *n*-grams*	66%
(Wong and Dras 2009)	syntactic errors*	74%
(Wong and Dras 2011)	syntactic rules*	80%
(Wong et al. 2012)	adaptor grammars*+	76%
(Swanson and Charniak 2012)	tree substitution grammars*	78%

Figure 2.2: Summary of NLI results listed in Brooke and Hirst for the ICLE corpus; accuracies added to chart. * indicates feature-based methods and + indicates likelihood-based methods.

types are then used as "stylistic markers" for NLI features with an SVM classifier. To find these errors in text, they used an open source grammar checker[2], as opposed to professionally edited text. Interestingly, ANOVA showed that the features had a measurable effect, but after combining their contrastive features with existing methods, they were not able to significantly increase the classification accuracy from Koppel et al.

The previous authors, Wong and Dras [24], follow their work on contrastive analysis, attempting to amend its shortcomings. Instead of error types, they use two different features obtained from grammatical parse trees: horizontal slices (production rules) and parse rerankings. They claim these are the first pure syntactic features used in NLI. For the production rules, they immediately applied information gain dimensionality reduction. The reranking features are those contained in the Charniak parser[3] and Stanford Parser[4] trained on the Wall Street Journal. Unlike the previous two attempts, the authors found MaxEnt to outperform SVM as the classifier. Additionally, five-fold cross validation was performed (as opposed to ten-fold), which means the accuracies can't be precisely compared with previous work. In any event, they report a final accuracy of 80%, which was the highest reported as of 2012.

Wong, Dras, and Johnson [25] explore the last author's—Mark Johnson's—adaptor grammars [26] to generate features. Simply, adaptor grammars are a non-parametric extension to PCFGs (probabilistic context free grammars). They can learn arbitrary-length word sequences (collocations); for example, *gradient descent* and *cost function* were learned under a machine learning topic.

[2] `http://queequeg.sourceforge.net/index-e.html`
[3] `http://cs.brown.edu/~ec/`
[4] `http://nlp.stanford.edu/software/lex-parser.shtml`

These adaptor grammars are used in two ways: in the first, collocations are used as features in a MaxEnt classifier. In the second, the grammar is trained on each class (representing native language). At test time, the most probable grammar to have generated the text is selected. For both tasks, the authors use five-fold cross validation on seven native languages. In the feature-based classification, they achieved 76%; in the language model-based classification, they achieved only 50%, a performance similar to the unigram word baseline [22].

Swanson and Charniak [27] made use of tree substitution grammars [28] (TSGs). Various tree induction methods are compared to generate features, and five-fold cross validation on seven native languages is performed. All TSG features outperformed the CFG baseline (at 73%). The highest TSG induction method was Bayesian induction at 78%.

Massung et al. [29] also make use of grammatical parse tree features, but mainly focus on their structural aspects as opposed to the syntactic category labels. In one classification task, they found these features to work well in determining the nationality of student essay writers from the CEEAUS dataset. These structural parse tree features may be applicable in other tree-based objects such as adaptor grammars, tree substitution grammars, and dependency parses, but this has yet to be explored.

Although not directly tackling NLI, authorship attribution from Kim et al. [30] does use grammatical parse tree features similarly to the above papers. They defined a new tree-based feature, k-embedded-edge (*ee*) subtrees: subtrees that share a set of k ancestor-descendant subtrees. Therefore, a 0-*ee* subtree would be one arbitrarily-sized subtree, and a 1-*ee* subtree would be one subtree and one descendant subtree anywhere in the parse tree. This creates an exponential number of potential patterns, and the authors define frequent pattern mining algorithms to prune the number of *ee* tree features. As with the last paper, this approach would be feasible for other tree structures, but is also unexplored.

In summary, Fig 2.2 lists the comparable accuracies from experiments run on the ICLE subset of five European languages. In general though, accuracies between 70% to 80% are standard for a wide variety of techniques and corpora.

The following two sections represent work that is less developed and cohesive than the previous two sections. The first deals with tools to evaluate and score L2 text; the second shows how L1 text can be made more approachable for its learners. For each topic we briefly touch on some relevant studies,

and leave the reader with some open questions and promising areas for future work.

2.3 Fluency Scoring

Which of the two following sentences sounds more natural?

1. *"If there are unexpected expenses, material for their lesson, for example, they may not be able to pay money to it only with monthly allowance."*
2. *"If there are unexpected expenses—school materials, for example—they may not be able to afford them with only a monthly allowance."*

Most readers would probably agree that the second sentence sounds much more fluent than the first, even though the first sentence has only very minor grammatical errors. Fluency scoring is thus related to GEC in this way. However, as evidenced above, a lack of grammatical errors does not necessarily mean that a sentence sounds native.

The ETS corpus [31] was designed to aid in NLI research, but it also has scores in the range $[1, 5]$ for each essay; these scores represent how well the student answered the essay question, and is *not* a measure of fluency. According to the essay scoring rubric[5], even essays with a perfect score may have "occasional language errors" that do not result in "imprecise presentation of content". Similarly, a score of 1 could mean either the response is not relevant to the prompt or the level of English is too low to understand. Thus while the level of English fluency may be correlated with the essay score, the score is strictly a result of how well the prompt was answered.

The e-rater system [32] was used to grade non-native essays from the ETS dataset. It used features such as "grammar, usage, mechanics, style, organization, length, word length, vocabulary, and correlation between prompt and essay vocabulary", but does not explain in detail how some features are derived. They found that e-rater had the same reliability of two human raters' scores, but the correlation between two human raters was about 0.60, or a weighted kappa rater agreement score of 0.44 (the low end of moderate agreement).

[5] `https://www.ets.org/Media/Tests/TOEFL/pdf/Writing_Rubrics.pdf`

Powers et al. [33] attempted to undermine the first version of e-rater's scoring abilities by taking advantage of its scoring system. They found it was easy to make e-rater give a higher-than-deserved score, but much harder to make it give a lower-than-deserved score. Even so, these scores are still based on answering the prompt, and not on English fluency.

Many non-native English datasets contain partitions of essays based on language level, but with such a small number of partitions (usually three), it is difficult to have a meaningful analysis of fluency. Besides, since most of the partitions are based on class level, they don't take into account which students are better L2 writers than their peers in the same class. To our knowledge, no serious study to measure purely fluency has been performed. This is most likely hindered by lack of reliable data as described previously.

One suggestion to tackle this problem is by Yasudao et al. [34]. They designed a machine translation evaluation metric specifically for measuring English fluency. Its main focus is communication skill, and less on vocabulary and grammar (which could be measured by other systems). They found a good correlation between their metric and a standard English fluency test. Like many machine translation systems, their algorithm works in a specific topical domain, namely sentences containing travel expressions and vocabulary. It seems there is no future work based on their research.

In conclusion, we have found that neither grammatical errors nor essay grades can be used to determine L2 fluency. What can be done? We have copora partitioned on approximate fluency, but these partitions are unlikely accurate enough, and certainly not fine-grained enough. Current work in essay grading is usually based on lexical features and still has a focus on content rather than style. See the list of classic systems [35] to get an idea of what features are used. Future work in fluency scoring would include reliable ways to specifically measure L2 fluency A special-purpose dataset for this task or extensive human annotation would likely be necessary.

2.4 Simplification and Summarization

Consider the following two sentences:

1. *"The main bar at King's is far older, and is the site of more informal meetings*

between students. The bar has been traditionally painted a socialist red, including a depiction of a hammer and sickle."

2. *"King's main bar is older. The bar is traditionally painted a socialist red, including a picture of a hammer and sickle."*

The first sentence is longer and uses a slightly larger vocabulary (*depiction* instead of *picture*). As a non-native speaker of English, it is likely that the second sentence is easier to understand, or would at least take less time to comprehend.

Summarization, simplification, and readability go hand in hand to help a non-native speaker understand text. Unlike NLI, GEC, and even fluency scoring, most algorithms operate solely on well-formed, native L2 passages. Simplification can be seen as an easy-to-understand summary of a more difficult text; simplification essentially "translates" one sentence to another, in efforts to make the result have a better readability. It is a form of monolingual machine translation when using a parallel corpus of advanced and simple language. For a detailed description of general text simplification, we direct the reader to [36].

Unfortunately, not much work has been done in text simplification *specifically* for non-native speakers. A typical use case is simplifying medical texts so the common reader can make sense of them (*e.g.* see [37]). Other use cases could be helping younger readers or users with learning disabilities.

Summarization can be thought of as a form of simplification. The second sentence is essentially a summarized version of the first. That is, it's shorter, easier to read, and contains all the necessary information required to understand the sentence. The amount of summarization determines the level of simplification and vice versa. In this section, we will usually use the word "simplification" due to the English-as-a-second-language applications. Non-native text summarization where text is both the input and output is its most common form, though this is not always necessarily the case. We will see an example of another case in our exploration of SYNTACTICDIFF.

Wikipedia and Simple Wikipedia[6] are common parallel corpora for this task. In fact, the first example sentence in this section is from Wikipedia and the second is from Simple Wikipedia. Both Wubben et al. [38] and Zhu et al. [39] use

[6] `http://simple.wikipedia.org/wiki/Main_Page`

them as corpora for sentence simplification via monolingual machine translation. The former uses non-native speakers to judge sentences from their system, but the system itself doesn't take into account the users' native language when forming the simplifications. The latter defines sentence splitting, deletion, reordering, and substitution operations on complex parse trees in order to simplify them into more understandable sentences. They evaluate with standard readability measures as well as perplexity from an English language model.

Lappas and Vlachos [40] show how to rank documents in a search engine to favor both relevance and readability for non-native speakers. The readability score is determined based on the user's native language, although this is not automatically detected. Each document is then assigned a $(relevance, readability)$ pair at query-time, and it can be imagined that documents are plotted in this 2D space. A document is said to dominate another document if it is more understandable and more relevant. In the 2D document space, documents that are not dominated by any other document are on the "skyline" (or perimeter) of the space. These are the documents that are browsed by the user. They evaluated their search engine based on the number of documents a user viewed before satisfaction, and found that taking readability into account decreased the number of documents that needed to be examined.

As the text simplification field continues to evolve, we hope to see more simplification tasks specifically aimed at helping second-language learners. The "teddy bear principle" states that language learners tend to stick with a relatively small set of learned syntactic patterns when speaking or writing in L2. Depending on the L1, a sentence simplification task could translate the complex sentences into a format more comfortable to the user. [41] analyze changes made to professionally abridged versions of newspaper articles to determine common translations. These common modifications could be incorporated in a monolingual translation model.

Another relatively unaddressed question is whether simplification is better than an alternative means to understanding (*e.g.*, elaboration). The thesis by Maxwell [42] considers this question and asserts that elaboration is more beneficial based on reading comprehension scores of Korean high school students studying English. She claims that simplification often results in unnatural-sounding phrases that do not resemble authentic L1 text. This is still an open problem that has not been approached with computational techniques.

CHAPTER 3

SYNTACTIC DIFF

This chapter motivates and defines this thesis' main contribution, SYNTACTICDIFF. We start with some background and the issue of an adequate text representation. Next, we discuss language models, a natural language processing concept heavily used by SYNTACTICDIFF. Then, the last two sections outline the contribution in more mathematical and algorithmic detail.

3.1 A Generic Comparative Text Mining Framework

Text representation plays a crucial role in information retrieval, text mining, and virtually all other text-related applications. The most popular text representation used in many applications is the simplest bag-of-words representation, which tends to work reasonably well for many content-processing tasks despite its simplicity. One reason for its popularity is its robustness—it is very general and can be applied to any natural language text. However, such a simple representation is clearly insufficient; for example, it cannot distinguish different orders of words. Improvement over bag-of-words representation has thus been attempted, including n-grams or phrase-based representations, and mixed representations based on part-of-speech tags and words.

However, virtually all the existing work on text representation has assumed that the representation of a text object such as a document would be derived based on *solely the document itself*. Unfortunately, such an "independent representation" strategy is insufficient for many tasks, particularly those that require discrimination that goes beyond pure content analysis.

For example, to support learning a second language at scale in Massive Open Online Courses (MOOCs), it is often necessary to cluster student essays based on their grammar mistakes to enable "batch grading" of a whole cluster together [43]. Since all the students may have been asked to write about similar

topics, a content-based representation would clearly not work well. To effectively cluster text documents for this application, we would need a representation of each document based on how far it deviates from some reference text data (*e.g.* writing by native speakers). A comparative analysis of a document with a reference text would be necessary in this case, allowing for the discovery of many subtle differences in the document from comparable native writing. Such a comparative analysis can reveal frequent article errors or incorrect verb form uses, among others. We can use the set of all such mistakes to represent the document in which they occurred, allowing us to cluster essays where similar mistakes are made.

Consider an authorship attribution variant with the goal of identifying the native language of a document's author, which was a shared task in 2013 [4]. In nature, this is a text categorization problem, so it is common to apply a supervised learning approach. As in the case of clustering, text representation plays a critical role here. Since different authors may have written about the same topic, pure content-based representations again would not work well. Instead, we would need to represent a document based on features that can characterize and distinguish the writing styles. Once again, comparative analysis of the document with a reference corpus of writings by native speakers on similar topics can be very useful for generating more discriminative features to characterize style differences; since writers speaking different native languages tend to have somewhat different writing styles, such features derived from comparative analysis of text are likely much more effective than ordinary content-based features for this categorization task.

In both examples above, we see a clear need for deriving a representation of a text object based on comparative analysis involving another reference text; such a comparative analysis approach to text representation has not been studied in any existing work. In this paper, we conduct the first study of such a new strategy for generating text representation via comparative analysis of text data. Specifically, we propose SYNTACTICDIFF, a novel edit-based method for transforming sequences of words given a reference corpus (model) and use these transformations directly as features or to derive useful features based on them for improved text representation. In addition, the proposed transformation method can be used directly to solve many interesting application problems involving text transformation or comparative analysis of text such as grammatical error correction.

The basic idea of SYNTACTICDIFF is to define three basic (and therefore general) edit operations: `insert` a word, `remove` a word, and `substitute` one word for another. These edits are used to transform a given sentence. With a source sentence S and a reference text collection R, we can ask the following question: what's the minimum set of edits that we have to apply to S in order to transform it into a sentence in R? This question is interesting because the "minimum set of edits" can be used to measure the deviation of S from sentences in R; what is most interesting is that this "measure" is not a numerical one, but a set of edits that can be features for text representation.

For example, suppose S is a sentence possibly with grammatical errors written by a non-native speaker, and R is a set of sentences written by native speakers on similar topics which includes a very similar sentence to S with no grammatical error. The minimum set of edits would be very meaningful because they are precisely the *corrections* we must make in order to correct the grammatical errors in S (making it look just like the one written by a native). Thus, we can represent the original sentence S with a minimum set of edits, instead of with the words or other content-based features derived from S. Such a transformation-based representation would be much more effective than content-based representation for generating clusters of sentences that share similar grammatical errors, a task useful for "batch grading" as discussed before.

However, there is one caveat here: what if there is no sentence in R that is very similar to S? We solve this problem by relaxing the requirement of transforming S to a sentence in R and simply requiring the new sentence S^*, resulted from applying a set of edits to S, to "look like" sentences in R. Formally, this can be quantified by estimating an n-gram language model θ based on R, and maximizing the probability of observing S^* from this language model, *i.e.*, seeking S^* that would maximize $P(S^*|\theta)$, or equivalently, minimizing the perplexity of S^* according to θ. This is a very general and robust strategy, as it allows us to compute the minimum set of edits (subject to some constraints on the edits, such as the maximum number of edits allowed) for any sentence S with respect to any reference text data R. This is similar to likelihood-based methods, but these methods are not created with such rigorously defined operations.

The obtained minimum set of edits can then be used as features to represent text in a context-sensitive way (R as context), which can be used as either an alternative or supplement to the existing content-based representation. By

varying the constraints on the edits in interesting ways (*e.g.* restricting the words to be inserted or deleted to only function words or varying R), we can naturally obtain many interesting variations of text representation that are not possible to generate by any existing methods.

It is easy to see that when restricted to insertion of function words and substitutions involving only lexical transformations, such an edit-based transformation method can be directly useful for grammatical error correction. However, it is important to note that the proposed method can have many other interesting applications also besides generating interesting features for representing text. For instance, the method can also be used for performing comparative analysis of opposite opinions about an issue in a debate. This can reveal the differences between the opinions since the edits that have to be applied to transform one group of opinions to the other (or vice versa) can potentially reveal the details of their differences. Furthermore, when comparing an article with a reference collection with only deletion edits allowed, we would obtain a set of deletion edits that represent the main topic of the article, since deleting words that are frequent in the article but not frequent in the reference collection is encouraged to make the article conform to the language model induced by the reference collection (those topical words likely have smaller probabilities in the reference collection, thus deleting them in the original article helps increase the likelihood).

In summary, SYNTACTICDIFF is a general text analysis framework for transforming (modifying) text with respect to a reference corpus using various edits; the goal is to transform a text object into another so as to better match the reference corpus. Aside from modifying single sentences, it can also be used to make syntactic comparisons between two bodies of text as well as using edits performed on a collection of sentences as features for text representation. We hope to be able to transform, compare, summarize, and induce features from text. The proposed definition will give us both the power and flexibility to solve these tasks.

Using the generic framework of SYNTACTICDIFF, chapter 4 proposes general methods for applying it to four different tasks and show that SYNTACTICDIFF is beneficial in each case. In the first task, we use weighted word edits with likelihood scoring for grammatical error correction. The method is compared against systems in an grammar correction shared task, and we find that SYNTACTICDIFF edits perform comparably while being much more general than

the other methods. The second task is native language identification: a classification problem predicting the native language of a student writer based on English essays. We represent documents as vectors of edits, and show that a combination of unigram words and SYNTACTICDIFF edits outperforms each representation individually. Then, the third task attempts to classify student essays based on their grade level in English fluency. We find that the edit differences between the student essays and a reference English corpus are able to capture a fluency signal. In the last task, we create clusters of student essays with similar errors via topic modeling, and find that the interpretability is significantly higher than an n-gram words approach.

3.2 Reference Language Models

The reference corpus provides guidance for how we transform a given text object through a reference language model estimated based on the reference corpus. Specifically, we would seek transformations to convert the original text object into a new one that would have a higher probability according to the reference language model.

Without loss of generality, we make use of an n-gram language model. An n-gram language model assigns probability to a sequence of m words, where each word is conditioned on the previous $n-1$ words. Thus, for a language model θ:

$$P_\theta(w_1, w_2, \ldots, w_m) \approx \prod_{i=1}^{m} P(w_i | w_{i-n+1}, \ldots, w_{i-1}). \tag{3.1}$$

In practice, we reserve probability mass for unseen events by smoothing our language model. A simple form of smoothing used by the SYNTACTICDIFF language model is linear interpolation. An example of this smoothing for a 3-gram language model is

$$\begin{aligned} P_\theta(w_i | w_{i-2}, w_{i-1}) = \lambda_3 P(w_i | w_{i-2}, w_{i-1}) \\ + \lambda_2 P(w_i | w_{i-1}) \\ + \lambda_1 P(w_i), \end{aligned} \tag{3.2}$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$ in order to ensure a valid probability distribution.

Perplexity is a measure for language model evaluation. It can be used to test

the likelihood of a sequence given a language model θ.

$$Perp(w_1, w_2, \ldots, w_m) = \left(\prod_{i=1}^{m} \frac{1}{P_\theta(w_i | w_{i-n+1}, \ldots, w_{i-1})} \right)^{\frac{1}{n}} \tag{3.3}$$

A lower perplexity (or cross-entropy) means that the sequence was more likely to have been generated by θ. We use perplexity per word as a normalized form of scoring for candidate sentences in SYNTACTICDIFF. For a more rigorous and detailed introduction to language models and their related concepts, please consult Jurafsky and Martin [44].

3.3 Transformation Edits

We define three basic edit operations on sentences:

1. **Insert** the word w after position j in sentence S: $insert(S, j, w)$. The inserted word is drawn from a set of words V^{INS}.
2. **Remove** the word at position j in S: $remove(S, j)$.
3. **Substitute** the word at position j in S with w: $substitute(S, j, w)$. The substituted word is drawn from a set of words potentially dependent on w_j: $V^{SUB}(w_j)$.

These three edit functions are used to incrementally transform the original sentence into multiple candidate sentences. The candidate sentences are scored based on perplexity using the reference language model, and the sentence with the lowest perplexity per word becomes the output. Setting V^{INS} to only insert non-content words and setting $V^{SUB}(w)$ to replace words with similar words or inflected forms of the word allows `insert` and `delete` to preserve the original meaning of the sentence, though this is not a requirement. It's possible that the two sets are defined to capture some other grammatical meaning as a particular task demands.

For an index j, there are candidates generated from each edit function for a total of $|V^{INS}|+1+|V^{SUB}(w_j)|$ edits in addition to the original sentence, which is also regarded as a candidate. Each iteration of SYNTACTICDIFF only performs the edit functions on one index. The index j is chosen by the least likely n-gram from the current sentence $S = w_1, w_2, \ldots, w_m$ (which is most promising

Algorithm 1 The SYNTACTICDIFF algorithm

procedure SYNTACTICDIFF(S)
  $candidates \leftarrow \{\}$
  Initialize V^{INS}
  Initialize $V^{SUB}(w) \forall w \in V$
  SYNTACTICDIFF(S, 0)
  return best candidate from $candidates$
end procedure

Algorithm 2 The recursive SYNTACTICDIFF algorithm

procedure SYNTACTICDIFF(S, $depth$)
  return if $depth = k$
  $j \leftarrow \arg\max_{i \in [0,m]} \{Perp(w_i, w_{i+1}, \ldots, w_{i+n-1} \in S)\}$
  for $w \in V^{INS}$ **do**
    $S' \leftarrow insert(S, j, w)$
    $candidates.add(S')$
    SYNTACTICDIFF($S', depth + 1$)
  end for
  $S' \leftarrow remove(S, j)$
  $candidates.add(S')$
  SYNTACTICDIFF($S', depth + 1$)
  for $w \in V^{SUB}(w_j)$ **do**
    $S' \leftarrow substitute(S, j, w)$
    $candidates.add(S')$
    SYNTACTICDIFF($S', depth + 1$)
  end for
end procedure

for increasing the likelihood and lowering the perplexity). The index of this n-gram is given by

$$j = \arg\max_{i \in [0,m]} \{Perp(w_i, w_{i+1}, \ldots, w_{i+n-1})\}. \tag{3.4}$$

Next, we need to choose k, the number of iterations to perform. Each iteration operates on all candidate sentences, so for iteration one, only one sentence is operated on. In the second iteration, all new candidates are operated on. Generally, we choose $k \in [1, 5]$ in order to preserve the main content of the original sentence. The full algorithm for SYNTACTICDIFF is given in Alg 1 and Alg 2. Initially, we learn an n-gram language model θ from a reference corpus and pick a maximum depth k. Not shown in the pseudocode are checks

to ensure edits aren't recomputed for duplicated sentences, since the same candidate sentence may be generated in different branches of the algorithm. This is a simple dynamic programming optimization.

3.4 Weighted Edits

Until now, each candidate sentence is scored equally based on minimizing perplexity per word, regardless of the number or type of edits. This gives the simple scoring function

$$S^* = \underset{S \in candidates}{\arg\min} \{Perp(S)\}. \tag{3.5}$$

However, we can improve the scoring function to capture some meaning in each edit:

$$S^* = \underset{S \in candidates}{\arg\min} \{\alpha \cdot Perp(S) + (1-\alpha) \cdot W_S\}, \tag{3.6}$$

where W_S is the edit weight (or edit penalty) of S and $\alpha \in [0,1]$. α controls the tradeoff between lowering perplexity and lowering penalty; for simplicity, in this first study of SYNTACTICDIFF, we simply set $\alpha = 0.5$, though obviously it is also interesting to further study how to optimize α in the future work. The edit penalty of S can be determined as the average penalty over all edits performed on S. Each penalty edit weight can be on $[0,1]$.

In this paper, we define four penalties, though the framework is general and any type of penalty may be defined using information from the current sentence or reference corpus. We define: an `insert` penalty, a `remove` penalty, a `substitute` penalty. We also have a base penalty incurred if *any* edit is performed, penalizing sentences with many edits.

If we set all penalties to zero, we arrive at the original SYNTACTICDIFF formulation; thus, weighted SYNTACTICDIFF is a generalization of the previous description. Furthermore, these penalties can be further refined to vary according to the specific words inserted, deleted, or substituted, and optimized based on specific needs of an application. Since only the scoring function to find S^* changes for weighted edits, the SYNTACTICDIFF algorithm remains unchanged from Alg 1 and Alg 2.

CHAPTER 4

APPLICATIONS AND EXPERIMENTS

The proposed SYNTACTICDIFF can be potentially useful for a wide range of interesting applications as we will further discuss in chapter 5. As specific case studies, in this chapter, we apply it to four different and representative text mining tasks related to non-native text analysis in a MOOC or any other online learning scenario. Please note though, that SYNTACTICDIFF could be used in virtually any text mining environment.

First, we show that SYNTACTICDIFF can be used to *search for a transformation* of a sentence with grammatical errors into one with no errors by using native writing as a reference corpus, thus performing *grammatical error correction* as monolingual translation. This application could be a tool that students use to correct or grade their own writing.

Second, we show that the edits found by SYNTACTICDIFF can be used *as features* to improve text representation for the *classification task* of native language identification, for which pure content-based features tend not to be very effective. Once a student's native language is known, that information could be used as a fluency score with a confidence level. Additionally, knowing the native language of a student would enable course material to be specifically targeted towards that demographic, or to combat "patriotic grading" [45].

Third, we show that edits based on a gold standard English referenece corpus may serve *as a fluency measure* in non-native essay grading. This allows graders to understand exactly why language may seem unnatural or unusual, and is displayed in the *fluency scoring task*.

Fourth, we show that the edits found by SYNTACTICDIFF for each sentence can be used *as new tokens* to replace the original text for topical analysis using topic models. When applied to student essays, this would allow course instructors to find groups of similar essays that share common errors. These clusters can be viewed as a form of *summary of the corpus* and can be used to form teams, pair complementary students, or allow batch grading.

Since our goal is to demonstrate the benefit of SYNTACTICDIFF in a variety of different tasks, we do not attempt to optimize the performance for any of these tasks and thus do not report detailed results for parameter variations.

All experiments and algorithms (including SYNTACTICDIFF) are open source and freely available online as part of the toolkit METAA[1]. The NUCLE corpus[2] (for grammar correction) and the ICNALE corpus[3] (for summarization, fluency scoring, and classification) are also freely available online. All experiments were run on a laptop computer with an eight-threaded processor and eight gigabytes of memory.

4.1 Non-Native Grammar Correction

Using the edits directly on each sentence can be seen as a form of monolingual translation. We use the NUCLE corpus [46] to investigate SYNTACTICDIFF's performance correcting grammatical errors. It is evaluated with precision, recall, and F_1 score using the same framework and testing and training data as the CoNLL-2013 Shared Task in Grammatical Error Correction [47].

Experimental Setup

We used the 1,036 training data sentences to do parameter selection on the four different edit penalties and maximum step size. Since the runtime of SYNTACTICDIFF is quite fast on the NUCLE corpus training data, we easily applied grid search on the weights and k (the maximum number of edits), optimizing the F_1 score. The n-gram value was fixed at $n = 3$, a standard value for sentence fluency scoring purposes. As the reference corpus, we used approximately 50,000 sentences from the Wall Street Journal that are part of the Penn Treebank, since this text is a staple of well-formed English.

The selected edit weights from the training data were 0.0 for substitute and base penalties, 0.07 for insert, and 0.30 for remove. This shows that the default SYNTACTICDIFF needs to remove fewer words to get better performance, while inserting slightly less. The selected value of k was 3.

[1] `http://meta-toolkit.github.io/meta/`
[2] `http://www.comp.nus.edu.sg/ nlp/corpora.html`
[3] `http://language.sakura.ne.jp/icnale/download.html`

Weight	LM	No-op	P	R	F_1	t
No	No	0.0%	2.96	4.49	3.57	120s
No	Yes	0.8%	3.22	4.47	3.74	**11s**
Yes	No	25.5%	18.78	**19.40**	19.09	123s
Yes	Yes	57.4%	**35.20**	17.55	**23.42**	**11s**

Figure 4.1: Grammar correction task: the table shows whether edit weights are used, whether insertions are done based on perplexity, how many final candidate sentences are unchanged (no-ops), precision, recall, F_1 score, and runtime in seconds. This system would place 7^{th} in the CoNLL shared task.

We set V^{INS} to be a short list of function words, since the omission of these is a common error. We considered using the Porter2 stemmer[4] for $V^{SUB}(w)$. This means `substitute` can exchange words that share the same stem. After some brief trials, we found the Porter2 stemmer was providing too many unrelated word substitutions. For example, the root *gener* can come from *generate*, *generic*, and *generous*, which are completely unrelated. These substitutions (*e.g.* $generate \rightarrow generous$) would change the meaning of the original sentence, so we modified the stemmer to only include step 0, 1a, and 1c. Essentially, these steps reduce plural and possessive forms into the same root.

We tested with the designated 345 testing data sentences and used the evaluation scripts from the shared task. Given a candidate sentence S, the predicted corrected form is a new sentence S^* that has the lowest perplexity (see section 3.2).

Results

Fig 4.1 shows the results of SYNTACTICDIFF used for grammatical error correction. We also included results without the edit positions selected by the language model and results without the tuned edit weights. Without edit points selected, edits are performed at every position in the sentence, generating many more candidates. Without edit weights, each particular type of edit is treated equally, and there is no distinction between many or few edits in scoring.

Lack of weighted edits and language model insertion is similar to Lee and

[4] `http://snowball.tartarus.org`

Seneff [10]. Of course, the language model is still used to score the candidates in all cases. As seen in Fig 4.1, the intelligent edit points greatly increase run time and the learned edit weights contribute significantly to the performance.

Some sentences in the NUCLE corpus are free of errors, so the correct annotation for these is a no-op. The true no-op rate in the testing data is 36.2%; all other sentences had at least one correction. A system with 100% no-ops received a precision and recall of zero using the CoNLL scoring script. We included the percent of no-ops in the table to compare how zealous each configuration was in suggesting changes. When no edit weights are used, virtually every sentence was modified in some way; consequently, having edit weights ensures that the top-ranked candidate sentence is fluent enough despite having edits.

For a more direct comparison, we can look at the results from the CoNLL shared task where the teams were judged by F_1 score. SYNTACTICDIFF's score of 23.42 would place it in seventh overall, beating out 65% (eleven) of the other teams. Not only does our method place fairly in the shared task standings, but SYNTACTICDIFF is a much more general system than its competitors. The other systems specifically targeted five error types: article/determiner, preposition, noun number, verb form, and subject-verb agreement. The standard system first classified errors into one of the five types. Then, a specific module was run on each error type in order to produce candidates. Finally, the set of candidates were scored, and results from each of the five modules was combined into the final corrected sentence.

SYNTACTICDIFF has no concept of different error types, and doesn't rely on classifiers to select particular modules to run. Thus, it is a much more general solution than required for the shared task.

4.2 Native Language Identification

We use the ICNALE native language identification corpus [48] to test the effectiveness of using the SYNTACTICDIFF edits as features to represent text for classifying English essays based on the native language of the author. This corpus contains 5,600 total essays on two prompts. We hypothesize that the SYNTACTICDIFF features capture the grammatical differences in writing styles of the eleven different native backgrounds.

Features	$\|V\|$	DL_{avg}	F_1	$Acc.$
Unigram words	9,021	129	80.1	81.8
SYNTACTICDIFF	12,279	56	73.1	75.4
Combined	21,300	185	**84.5***	**85.9***

Figure 4.2: Classification task: comparison between the three methods on the ICNALE essays. Displayed are vocabulary size, average document length, F_1 score, and accuracy. *Combined results are significantly higher with $p < 0.001$.

Experimental Setup

The same bag-of-edits representation as the summarization task is used as input for a classifier to predict the native language of the student essay writer. The Wall Street Journal sentences from the Penn Treebank are used for the reference language model as they were for the monolingual translation task.

As a baseline, we use a standard unigram words feature representation with stemming and stop word removal. Additionally, we combine the unigram words representation with the SYNTACTICDIFF features to see if the performance increases compared to using only one method.

The ICNALE corpus is split in half, based on whether the essay is a smoking essay or a part-time job essay. We use the part-time job subcorpus as a development set to do parameter selection on n and k, for the n-gram language model and maximum number of edits respectively. Once the parameters ($k = 5, n = 5$) were chosen, we evaluated with five-fold cross validation on the smoking testing set. Each fold of the cross validation is used to do an unpaired t-test for statistical significance. For both development and testing, we use the default SVM classifier that is part of the META toolkit. The unigram words baseline and feature combination are also part of the same toolkit.

Since adding edit weights will always decrease the score of candidate sentences, we set them all to zero for the classification task. We want the learned SYNTACTICDIFF model to have full control over the generated edits that appear as features. In contrast to the monolingual translation task, we prefer to *minimize* the number of no-ops, since each edit operation is used as a feature; more no-ops means less information is represented. The edit weights are easily set if the user requires, $e.g.$ to ignore a particular operation. Finally, we leave V^{INS} and $V^{SUB}(w)$ the same as the summarization task.

	CHN	ENS	HKG	IDN	JPN	KOR	PAK	PHL	SIN	THA	TWN
CHN	**92**	0	0	0	1	1	1	1	0	1	3
ENS	0	**90**	1	0	2	2	0	2	1	1	1
HKG	10	3	**64**	1	2	0	1	2	8	5	4
IDN	2	1	1	**83**	0	1	1	3	1	6	1
JPN	1	1	0	1	**94**	2	0	0	0	0	1
KOR	5	1	1	1	7	**76**	1	1	0	6	1
PAK	1	0	1	0	0	1	**94**	2	0	1	0
PHL	4	1	0	0	0	2	1	**84**	2	5	1
SIN	1	2	0	1	0	2	1	5	**87**	1	0
THA	2	0	0	2	1	3	0	1	0	**90**	1
TWN	12	1	2	0	3	6	2	2	1	5	**66**

Figure 4.3: Classification task: confusion matrix of combined features on the ICNALE corpus. Overall accuracy of 85.9%. Percentages have been rounded for readability. Each $(row, column)$ index represents the fraction of times row is labeled as $column$; thus all rows sum to 100%.

Results

Fig 4.2 shows a comparison between the three methods: unigram words baseline, SYNTACTICDIFF, and a combination. While unigram words does outperform edit features in F_1 and accuracy, a combination is able to increase both measures at a significance level of $p < 0.001$. This shows that the syntactic edit features capture an orthogonal perspective of the student essays compared to the lexical features.

Fig 4.3 shows a confusion matrix of the eleven classes using the combined features. Each row is a distribution over which class label was chosen for the given row name; the diagonal represents a correct categorization. From this, we see that Japanese and Pakistani students are confidently modeled. Students from Hong Kong and Taiwan and more easily confused with native Chinese speakers, which is logical.

The most informative features for some selected classes are shown in Fig 4.4 according to information gain [49]. Information gain is a commonly-used feature selection metric in the machine learning and information retrieval communities. It describes the difference in entropy by knowing the presence or absence of a specific term appearing in a class.

Some features are obvious and not as informative to the human reader: Chinese and Korean students overuse *China* and *Korea* compared to the reference

CHN	HKG	ENS	JPN	KOR
remove(people's)	remove(hong)	remove(<s>)	remove(seat)	remove(<s>)
remove(china)	remove(kong)	insert(is)	remove(nonsmoking)	remove(korea)
insert(the)	insert(the)	remove(bad)	remove(tobacco's)	remove(sterility)
remove(harmony)	sub(forced→forcing)	insert(a)	sub(so→be)	insert(or)
sub(people's→people)	remove(don't)	unmodified	remove(can't)	remove(rice)
remove(etc)	remove(carcinogenic)	remove(good)	remove(foods)	remove(habit)
insert(such)	sub(affected→affecting)	insert(such)	remove(opinion)	remove(non)
sub(terrible→terribly)	remove(country)	insert(to)	remove(two)	sub(fair→fairly)

Figure 4.4: Classification task: edit features selected via information gain for 5 of the 11 classes in the ICNALE corpus.

language model. Less apparent (yet still useful) edits are the Chinese students' overuse of *etc*, the Hong Kong students' underuse of *the*, the Japanese students' mixup between *so* and *be*, and the Korean students' differentiation between *fair* and *fairly*. We also notice that the native English-speaking students have *unmodified* as a main feature, meaning the perplexity-based candidate scoring preferred their original sentences over edited ones.

There are also a few artifacts of the tokenization method; the sentence marker <s> appears as a top feature, implying that English and Korean speakers tend to have shorter sentences, at least compared to the reference model.

4.3 Fluency Scoring

In fluency scoring, we wish to determine a writer's ability to write as close to native L2 as possible. Unlike regular essay scoring (as discussed in chapter 2), our fluency scoring is not a measure of how well a particular essay prompt was answered. Rather, it is topic-independent and must only rely on the style and grammar of the text.

Experimental Setup

We use the English proficiency grade levels from the ICNALE corpus [48] to determine if the SYNTACTICDIFF edit features can be used to distinguish between different fluency levels. The corpus has its essays categorized into four levels:

- **A20**: "waystage", with 960 essays
- **B11**: "lower threshold", with 1904 essays

Features	$\|V\|$	DL_{avg}	F_1	$Acc.$
Unigram words	9,021	129	67.6	68.0
SYNTACTICDIFF	12,279	56	60.4	61.1
Combined	21,300	185	**68.9**	**69.7**

Figure 4.5: Fluency scoring task: comparison between the three methods on the ICNALE essays. Displayed are vocabulary size, average document length, F_1 score, and accuracy. Unfortunately, the increase in performance using the combined features is not statistically significant.

- **B12**: "upper threshold", with 1872 essays
- **B20**: "higher", with 464 essays

Since the class distribution is not balanced, we combine the lower two groups and the higher two groups together. This also ensures we are operating on a true classification problem with roughly balanced classes (55% baseline), and not a regression. We use the exact same SYNTACTICDIFF setup as described in section 4.2 since this is also a text categorization problem. These settings are $k = 5, n = 5$ and the same V^{INS} and $V^{SUB}(w)$.

Results

Results for this task are displayed in Fig 4.5. Compared to the previous classification task, this task is clearly much harder. For example, in the 11-class NLI problem, unigram words achieved almost 82% accuracy. While here, in the 2-class fluency problem, unigram words reached just 68% accuracy.

Like the NLI task, SYNTACTICDIFF features underperformed unigram words, but their combination resulted in a higher accuracy. Unlike the NLI task though, the increase in accuracy and F_1 score was not statistically significant at any meaningful level. We hypothesize that the grade level partitioning of students as a proxy for L2 fluency is simply not an accurate enough measure. For example, it's very probable that the proficiency level of L2 writing varies greatly across grades, let alone across the classes and nationality backgrounds.

In the best case testing scenario, we would have gold standard data based on human evaluators of fluency, rather than a rough classroom partitioning. It would still be interesting for future work to investigate if any other syntactic

features may play a role in separating the student fluency. For example, our previous work in structural tree features [29] might capture sentence patterns that are only taught at a higher level.

Although a challenging problem, we can in conclusion say that the SYNTACTICDIFF features may have a role in separating fluency levels based on L2 grade level.

4.4 Summarization of Non-Native Text

Summarizing student essays can give insight into how they are written. Comparable essays will have similar deviances from fluent English. Does a group of students make similar errors? Can we target specific problem areas depending on the group of students we speak to? Or, can we pair students with complementary strengths and weaknesses?

Topic models such as latent Dirichlet allocation [21] are a powerful text analysis tool. After running a topic modeling algorithm, each document in a corpus is assigned a distribution over a fixed number of topics. A topic itself is a distribution over the corpus vocabulary.

For example, in a scientific literature corpus using a unigram words representation, a document may be dominated by (*e.g.*) topic two. The three highest probability words in topic two could be *cell*, *molecule*, and *gene*, implying that the document is mainly about biology.

We can use the power of topic models to simultaneously cluster and summarize errors in non-native English essays. As we will see, unigram (and even bigram) words will not be sufficient to understand common fluency differences between documents. Instead, using SYNTACTICDIFF edit features as new tokens to replace the original words in text captures what we'd like to see: is a group of students confused about article use? A bag-of-words method only shows the presence of a term being useful; a bag-of-edits representation shows presence, absence, and substitution.

For each cluster (collection of documents with a concentration of a particular topic), we are delivered the common differences between the essays and the native English essays. For a semi-supervised method, these native essays could be a small subset judged acceptable by the instructors. For a completely unsupervised method, the reference language model could be from a similar

corpus or a previous semester.

Experimental Setup

We compare SYNTACTICDIFF edit tokens with unigram and bigram words using the 2,800 ICNALE essays debating public smoking. We hypothesize that the edit tokens will be more interpretable than the competing methods.

Each document is treated as a "bag-of-edits", where SYNTACTICDIFF is run on each sentence in every document. For example, a small feature vector for a document could be

$$\{insert(the) : 3, substitute(a \rightarrow an) : 1, remove(of) : 2\}.$$

We run LDA from META on this alternate document representation and examine the distributions of edits and topics that result. The inference method used is collapsed variational Bayes [50] with hyperparameters set to 0.1, encouraging sparse distributions.

Since the summarization task is unsupervised, we have no clear objective for parameter selection. Thus, we leave the weights at zero. However, based on the observed output, the user is free to adjust the penalties in order to perturb the results in a direction he or she chooses. Perhaps only substitutions are currently of interest. Due to space constraints, we do not investigate further than all zeroed weights. We set $k = 1$ to get the most likely change to the original sentence, and set $n = 3$. We set V^{INS} to the same function word list as the error correction task and used the full Porter2 stemmer for $V^{SUB}(w)$ since there was no requirement for such precise substitutions. The LDA inference is run with a maximum of one thousand iterations, though all three representations converged before this limit.

Results

Fig 4.6 compares vocabulary sizes and iteration runtime for the LDA inference. Since the SYNTACTICDIFF edits have much lower dimensionality than the word vectors, inference is significantly faster, even on this relatively small dataset. Fig 4.7 shows a sample of topics learned from the ICNALE smoking corpus.

Features	$\|V\|$	DL_{avg}	Iteration t	$500t$
Unigram words	11,580	256	3.2s	26.7m
Bigram words	130,411	255	5.4s	45.0m
SYNTACTICDIFF	2,079	15	0.4s	3.3m

Figure 4.6: Summarization task: different tokenization methods for 16 topics on the ICNALE smoking corpus. Displayed are vocabulary size, average document length, LDA inference iteration time, and the time for 500 iterations for comparison.

We can see the *n*-gram representations capture more content-based themes while the edit tokens capture syntactic similarities. For unigram words, topic 1 deals with the physiological concerns of smoking. Topic 12 discusses banning smoking in restaurants, while topic 15 is more nationally-focused. Topic 4 may be of some use, suggesting an overuse of personal pronouns.

Bigram words have almost the same interpretability as unigram words. Topic 4 is similar to topic 12 from the unigram model. Each topic is more of a theme, rather than a collection of grammatical differences. We only see positive essay tokens in each topic, as opposed to lacking (missing) ones.

On the other hand, the SYNTACTICDIFF edits give some insight into the syntactic structure of the student essays. For example, consider these excerpts from three different documents: "*Because it is so bad to mom with baby*", "*In restaurant when people...*", "*...go to restaurant to have meal*". Each student has article use errors which `insert(a)` from topic 4 would fix. The word *a* would never appear in an *n*-gram topic model because it is absent in each of these documents.

The same three essays also have an overuse of the word *so*, which `remove(so)` from topic 4 would make more fluent: *so bad, so scared, so dead*. In fact, the first essay contains the phrase *so bad* five times in about fifteen sentences. The third essay contains the sentence "*The smoke make many people feel so bad.*" Aside from the *so* issue as before, there is a subject-verb disagreement between *the smoke* and *make*. While other essays may correctly use the verb *make*, these particular essays use it in an incorrect way such that `sub(make->makes)` is a correction.

Unigram Words

topic 1	topic 4	topic 8	topic 12	topic 15
cancer	i	the	restaurant	i
lung	very	of	banned	japan
smokers	my	tobacco	all	ban
disease	he	cigarettes	country	japanese
heart	was	government	agree	a
nicotine	think	quit	reasons	government
cause	don't	increase	in	just
passive	when	decrease	people	think

Bigram Words

topic 1	topic 4	topic 8	topic 12	topic 15
smoke cigarette	restaurant owners	if you	the media	passive smoker
smoking zone	smoking bans	you are	harmful for	active smoker
can make	bars and	you can	responsibility of	active smokers
sick in	customers would	when you	hotels or	in indonesia
global warming	or non	you smoke	cigarette companies	the active
this policy	ban on	for your	have shown	can disturb
public space	in bars	around you	and teenagers	more dangerous
make many	smoke filled	yourself and	or anything	all restaurant

SYNTACTICDIFF

topic 1	topic 4	topic 8	topic 12	topic 15
insert(the)	insert(a)	remove(you)	remove(so)	remove(area)
remove(opinion)	remove(so)	insert(to)	insert(for)	sub(seat→seats)
remove(cigarettes)	sub(lung→lungs)	sub(reason→reasons)	insert(in)	remove(of)
sub(give→giving)	sub(make→makes)	sub(ban→banning)	remove(not)	sub(stop→stopped)
remove(bans)	remove(healthy)	remove(us)	sub(have→having)	remove(again)
insert(you)	remove(reasons)	remove(person)	remove(nonsmoker)	insert(i)
remove(totally)	remove(as)	insert(are)	remove(that)	remove(all)
sub(cause→causes)	remove(even)	remove(better)	remove(increasing)	insert(it)

Figure 4.7: Summarization task: 5 of 16 topics learned from the ICNALE smoking corpus with three tokenization methods. The *n*-gram methods capture writing themes while SYNTACTICDIFF captures similar errors.

CHAPTER 5

DISCUSSION

This chapter addresses various questions and hypotheses brought up in our study, particularly the generality of SYNTACTICDIFF and the new application and research directions it can potentially open up.

5.1 Generality

As a new way of representing text, SYNTACTICDIFF is very general and robust; just like the bag-of-words representation, the bag-of-edits representation can be applied to *arbitrary* text data to obtain interesting variations of text representation. Its applications are also not restricted to improving text representation as we will further discuss later.

In general, we should think of SYNTACTICDIFF as a general framework, rather than a particular algorithm. Virtually all the components in SYNTACTICDIFF are configurable; most obviously, edit weight values, n-gram settings, and the reference corpus. Edit weights and n-grams values do not necessarily contribute to any specific syntactic meaning. Rather, these settings are for tuning a model against some objective function, which can vary according to applications (*e.g.*, in the grammatical error correction case, we set edit weights to optimize F_1 score).

The reference language model from the reference corpus plays a more important role in the meaning of each edit. It steers the edit transformations in a particular direction, coaxing each candidate sentence to align with the reference. In our experiments, we considered the reference to be gold standard language, since our tasks dealt with non-native English speakers. Modifying each sentence to minimize its distance with well-formed English makes sense. However, there are many ways to choose and set the reference, enabling the support of other interesting tasks.

For example, suppose we operate on a sentiment analysis dataset. We have a reference model of very positive sentences, and use SYNTACTICDIFF to translate candidate sentences to match the reference. Depending on the sentiment polarity of the candidate sentences, do negative sentences have a different pattern of edits than positive ones? It is in this way that the reference language model choice influences the significance of each edit.

In our experiments, we defined four edit weight penalties. In practice, these could be almost anything the user desires. Returning to the sentiment analysis task, imagine an edit weight penalty that is imposed if the words *no* or *not* are inserted. Or, if a word has a positive sentiment affiliation a penalty is also triggered. Finally, what if at each iteration, a penalty is imposed if the edit operation changes the polarity of the sentence? Some of these suggestions require a classifier in the candidate generation stage; alternatively, sentiment valence scores [51] could be used as a crude (yet effective) judgement.

Sentence edit features themselves are also configurable; for instance, we could include the previous word or word index. Then `insert(the)` could be come `insert(the|in)` meaning add *the* after *in* or `insert(4, the)` representing add *the* in the fourth position in the sentence.

Due to space constraints, we could not investigate all possible variations described above, but we envision much future work in this direction.

5.2 Beam Search

Beam search is a common algorithm used for decoding in statistical machine translation [52]. Decoding is the processes of combining short phrases in the target language together until a complete sentence in the target language is reached. Expanding all possible hypotheses to find the best would certainly be intractable (all possible phrase combinations). To mitigate this problem, beam search only explores the search graph along the k most likely candidate partial sentences. We say here that the algorithm uses a beam size of k.

We can use a similar technique in SYNTACTICDIFF candidate generation. The only difference here is that there isn't an immediately obvious end to our search. In machine translation, the most likely path to the end of the translated sentence is our goal, but in SYNTACTICDIFF there is no clear "end" to our translation (*i.e.*, application of edit operations).

In this scenario, we have several options to determine when to end. For one, we could edit until the perplexity per word is within some threshold of the reference corpus average. This would signify that our edits have translated to a level that is comparable to the style of the reference. Alternatively, we could edit up to a certain depth of edits. There are tradeoffs in each method. For the first, we sacrifice potential "decoding" time in order to get an optimal score. However, it's possible that we can never get to a satisfactory score or we modify the sentence too much and change its meaning. For these reasons, we'd still have to set a maximum edit number. In the second proposed method—the depth cutoff—we sacrifice potential fluency with respect to the reference corpus for a guaranteed running time. In both cases, we can still use the parameter k to set the beam size.

Using beam search in SYNTACTICDIFF would be a relatively simple modification since it already uses a priority queue internally to keep track of the best candidates. Instead of using recursive calls to expand the search space, after each set of edit operations is performed, the top k items can be popped from the priority queue and run through the SYNTACTICDIFF code again, where the new candidates are again added back onto the queue. This process is repeated until either of the proposed stopping conditions is met.

The original beam search formulation in statistical machine translation has some nice theoretical guarantees, but since our objective is not the same, these claims would likely not hold. However, it is clear to see that using beam search can result in increased accuracy as well as decreased computation time.

5.3 Applications

The four tasks that we have applied SYNTACTICDIFF to only represent a few of the many possible uses, but even these already have a very broad scope:

Text transformation

In our first task (of grammar correction), the edits are used directly to search for an optimal transformation of an original sentence. We may view this as an interesting new retrieval model that allows us to use the original sentence as a query to "retrieve" a relevant sentence that best matches the query, where

"matching" is based on the edits that we allow. By varying the edits allowed, their weights, and the choice of reference language model, this can potentially support many interesting text transformations that can easily go beyond grammatical error correction (like improvement of coherence, retrieval of opposite opinions, or text summarization).

Improving text representation for machine learning

In our second task, we used the edits to represent text in a supervised learning setting, and showed superior performance of such a text representation in comparison to existing text representation methods for the task of native language identification. Supervised learning is widely applied in many text processing tasks. Thus SYNTACTICDIFF can be potentially useful for improving text representation for many of these tasks.

Note that we do not have to solely rely on edits for text representation, and can in general combine edit-based representation with content-based representation. This would provide an interesting general and robust way to represent text. Moreover, such an improved representation can easily be exploited in the feedback process of a retrieval task where we face the problem of supervised or semi-supervised learning from a set of feedback documents and the representation of these feedback documents can be improved with SYNTACTICDIFF.

Stylistic analysis

Fluency scoring is an application of style analysis or author profiling [53]. This subfield of text mining and NLP seeks to categorize the author of an anonymous piece of text. For example, the text may be a forum post or a newspaper article. Attempting to describe the author of either of these could be useful for tasks such as digital forensics and computer (or other) security.

SYNTACTICDIFF allows such stylistic comparisons to be made by comparing a current piece of text to syntactic or word choice differences to a reference corpus. This is an extension and application of the text representation for improved machine learning algorithms. Our third task embodies this notion in the form of fluency scoring (*i.e.*, considering fluency level as a particular style of writing). Of course, the SYNTACTICDIFF edit features can be used to

capture other such styles to solve problems in authorship attribution, deception detection, or even plagiarism detection.

Comparative text mining

In our fourth task, we used the edits to represent the original text in an unsupervised learning setting (*i.e.*, topic modeling), which enabled discovery of interesting clusters of related edits. It is very easy to imagine the use of this strategy for many other unsupervised learning methods such as matrix factorization. Also, there are many variants of the basic topic models that can perform more sophisticated topic analysis.

All these algorithms can be combined with SYNTACTICDIFF to open up interesting new opportunities for comparative text mining. Once again, we can vary the edits and reference language model to steer the analysis in many ways to satisfy the need of a particular application. Revealing differences and similarity at the level of edits enables understanding of very subtle differences in opinions and language usage that simply cannot be captured by the standard unigram words approach.

There are also some other applications we can envision. For example, can slight syntactic differences captured by transformation edits aid in detecting patients likely to commit suicide [54]? In information retrieval, translating a query into the language style of the corpus could yield better search results than leaving the query in its original state. In clustering, a standard bag-of-words representation could be used for the similarity measure. To measure cluster cohesiveness, SYNTACTICDIFF could be used to see whether the language of each cluster is similar with respect to the corpus as a whole. We anticipate many more creative uses of this framework in other text mining tasks to be possible.

5.4 Semantic Diff

Using a customized SYNTACTICDIFF for each task allows researchers to gain insight into the differences between subsets of a corpus. How much customization is required to push SYNTACTICDIFF to SEMANTICDIFF?

We have already seen how edit penalty types can be imposed in an ad hoc manner, and how their weights can be chosen intuitively. Although we set $V^{SUB}(w)$ to be a stemmer, it could just as easily be a thesaurus or negator, focused on word sense disambiguation.

We can get even more creative knowing the parts of speech of each word. What if we only insert articles and determiners instead of a list of common function words? We can even design penalty weights for the part of speech. Is it more important to remove a determiner than it is a verb? It depends on the application, and can be learned automatically. Although these many possibilities greatly expand the search space, more advanced candidate selection algorithms such as beam search [55] (described above) can easily be applied.

With a basis for penalty creation, it would be possible to create penalty types on the fly during a training phase. We can break the definition of a penalty into context and an argument. For instance, one context could be surrounding part of speech tags, and the argument is the current word examined in an edit operation. Once SYNTACTICDIFF operates in this format, we can arbitrarily create penalties.

Given all these modifications enabling increased generality, we assert that SEMANTICDIFF is not only attainable, but will form the landscape of edit-based rich text meaning.

CHAPTER 6

CONCLUSIONS

This thesis consisted of two main parts. The first introduced the landscape of the *non-native text analysis* field in text mining and natural language processing. The second proposed a general algorithm, SYNTACTICDIFF, that is applied in this domain and is shown to capture signals from text that are not possible to describe in standard text representation schemes. The main contribution of this thesis is contained in the second part, although the first part is a very useful introduction to an emerging field and provides some concrete motivation for an important application of SYNTACTICDIFF.

SYNTACTICDIFF is a novel and general framework for many text mining tasks that examines syntactic differences between current text and a reference background collection. These differences are captured in weighted edit operations `insert`, `remove`, and `substitute`. These text object edits can not only be used to generate an alternative representation of text data that is complementary with the current content-based representation, but also be used to directly support a wide range of interesting applications.

We evaluated the generality and effectiveness of SYNTACTICDIFF using four distinct representative tasks: grammatical error correction, native language identification, fluency scoring, and corpus summarization. In all areas, SYNTACTICDIFF provided advantages in its unique representation, clearly demonstrating its empirical benefit.

In the first, we achieved remarkable performance considering our generality compared to other systems in a shared task. In the second task, we increased the accuracy of a baseline native language identification system by augmenting with SYNTACTICDIFF edit features. In the third task, we showed that the addition of SYNTACTICDIFF edit features are able to improve the classification accuracy compared to unigram words alone. In the last, we were able to summarize grammatical errors better than baseline systems.

As discussed in detail in chapter 5, our exploration in this thesis was only

the tip of the iceberg of the SYNTACTICDIFF framework's great potential; there are many interesting future directions to further explore, particularly in leveraging such a new representation of text in all kinds of applications, exploring different configurations for various comparative text analysis tasks, and further generalizing the framework to capture more semantic meaning—moving from SYNTACTICDIFF to SEMANTICDIFF.

REFERENCES

[1] B. Council, "How Many People Speak English?" April 2014, http://www.britishcouncil.org/learning-faq-the-english-language.htm.

[2] C. Leacock, M. Chodorow, M. Gamon, and J. R. Tetreault, *Automated Grammatical Error Detection for Language Learners, Second Edition*, ser. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2014.

[3] K. Kukich, "Techniques for Automatically Correcting Words in Text," *ACM Comput. Surv.*, pp. 377–439, 1992.

[4] J. Tetreault, D. Blanchard, and A. Cahill, "A Report on the First Native Language Identification Shared Task," in *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, 2013, pp. 48–57.

[5] Ng, Hwee Tou and Wu, Siew Mei and Briscoe, Ted and Hadiwinoto, Christian and Susanto, Raymond Hendy and Bryant, Christopher, "The CoNLL-2014 Shared Task on Grammatical Error Correction," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, 2014, pp. 1–14.

[6] E. Stamatatos, W. Daelemans, B. Verhoeven, B. Stein, M. Potthast, P. Juola, M. A. Sánchez-Pérez, and A. Barrón-Cedeño, "Overview of the Author Identification Task at PAN 2014," in *CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers*, Sheffield, UK, 2014/09/18 2014.

[7] J. Tetreault, J. Burstein, and C. Leacock, Eds., *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*. Baltimore, Maryland: Association for Computational Linguistics, June 2014.

[8] S. Corston-Oliver, M. Gamon, and C. Brockett, "A Machine Learning Approach to the Automatic Evaluation of Machine Translation," in *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*. Toulouse, France: Association for Computational Linguistics, July 2001, pp. 148–155.

[9] M. Gamon, A. Aue, and M. Smets, "Sentence-level MT Evaluation Without Reference Translations: Beyond Language Modeling," in *In European Association for Machine Translation (EAMT)*, 2005.

[10] J. Lee and S. Seneff, "Automatic Grammar Correction for Second-Language Learners," in *In Interspeech. ISCA*, 2006.

[11] C. Brockett, W. B. Dolan, and M. Gamon, "Correcting ESL Errors Using Phrasal SMT Techniques," in *ACL*. The Association for Computer Linguistics, 2006.

[12] A. Rozovskaya and D. Roth, "Training Paradigms for Correcting Errors in Grammar and Usage," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, ser. HLT '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 154–162.

[13] R. West, Y. A. Park, and R. Levy, "Bilingual Random Walk Models for Automated Grammar Correction of ESL Author-produced Text," in *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, ser. IUNLPBEA '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 170–179.

[14] D. Dahlmeier and H. T. Ng, "Correcting Semantic Collocation Errors with L1-induced Paraphrases," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, July 2011, pp. 107–117.

[15] D. Dahlmeier and H. T. Ng, "Grammatical Error Correction with Alternating Structure Optimization," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 915–923.

[16] R. K. Ando and T. Zhang, "A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data," *J. Mach. Learn. Res.*, vol. 6, pp. 1817–1853, Dec. 2005.

[17] M. Koppel, J. Schler, and K. Zigdon, "Determining an Author's Native Language by Mining a Text for Errors," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 624–628.

[18] J. Brooke and G. Hirst, "Robust, Lexicalized Native Language Identification," in *COLING*, 2012, pp. 391–408.

[19] E. Stamatatos, "A Survey of Modern Authorship Attribution Methods," *Journal of the American Society of Inf. Sci. Technol.*, vol. 60, no. 3, pp. 538–556, 2009.

[20] S. Granger, "The International Corpus of Learner English: A New Resource for Foreign Language Learning and Teaching and Second Language Acquisition Research," pp. 538–546.

[21] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[22] O. Tsur and A. Rappoport, "Using Classifier Features for Studying the Effect of Native Language on the Choice of Written Second Language Words," in *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, ser. CACLA '07, Stroudsburg, PA, USA, 2007, pp. 9–16.

[23] S.-M. J. Wong and M. Dras, "Contrastive Analysis and Native Language Identification," in *Australasian Language Technology Association Workshop 2009*, 2009, p. 53.

[24] S.-M. J. Wong and M. Dras, "Exploiting Parse Structures for Native Language Identification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11, Stroudsburg, PA, USA, 2011, pp. 1600–1610.

[25] S.-M. J. Wong, M. Dras, and M. Johnson, "Exploring Adaptor Grammars for Native Language Identification," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, ser. EMNLP-CoNLL '12, Stroudsburg, PA, USA, 2012, pp. 699–709.

[26] M. Johnson, T. L. Griffiths, and S. Goldwater, "Adaptor Grammars: A Framework for Specifying Compositional Nonparametric Bayesian Models," in *NIPS*, 2006, pp. 641–648.

[27] B. Swanson and E. Charniak, "Native Language Detection with Tree Substitution Grammars," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ser. ACL '12, Stroudsburg, PA, USA, 2012, pp. 193–197.

[28] P. Blunsom and T. Cohn, "Unsupervised Induction of Tree Substitution Grammars for Dependency Parsing," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 1204–1213.

[29] S. Massung, C. Zhai, and J. Hockenmaier, "Structural Parse Tree Features for Text Representation," in *International Conference on Semantic Computing*, 2013, pp. 9–16.

[30] S. Kim, H. Kim, T. Weninger, J. Han, and H. D. Kim, "Authorship Classification: A Discriminative Syntactic Tree Mining Approach," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '11. New York, NY, USA: ACM, 2011, pp. 455–464.

[31] D. Blanchard, J. Tetreault, D. Higgins, A. Cahill, and M. Chodorow, "TOEF11: A Corpus of Non-Native English," *Technical Report, Educational Testing Service*, 2013.

[32] Y. Attali, J. Burstein, Y. Attali, J. Burstein, M. Russell, D. T. Hoffmann, Y. Attali, and J. Burstein, "Automated Essay Scoring with E-Rater V.2," *Journal of Technology, Learning, and Assessment*, 2006.

[33] D. E. Powers, J. Burstein, M. Chodorow, M. E. Fowles, and K. Kukich, "Stumping e-Rater: Challenging the Validity of Automated Essay Scoring," *Computers in Human Behavior*, vol. 18, no. 2, pp. 103–134, 2002.

[34] K. Yasuda, E. Sumita, G. Kikui, F. Sugaya, T. Takezawa, and S. Yamamoto, "Automatic Measuring of English Language Proficiency Using MT Evaluation Technology," in *Proceedings of the Workshop on eLearning for Computational Linguistics and Computational Linguistics for eLearning*, ser. eLearn '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004, pp. 53–60.

[35] S. Valenti, F. Neri, and R. Cucchiarelli, "An Overview of Current Research on Automated Essay Grading," *Journal of Information Technology Education*, vol. 2, p. 2003, 2003.

[36] A. Siddharthan, "A Survey of Research on Text Simplification," pp. 259–298, 2014.

[37] E. Abrahamsson, T. Forni, M. Skeppstedt, and M. Kvist, "Medical Text Simplification using Synonym Replacement: Adapting Assessment of Word Difficulty to a Compounding Language," in *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*. Gothenburg, Sweden: Association for Computational Linguistics, April 2014, pp. 57–65.

[38] S. Wubben, A. van den Bosch, and E. Krahmer, in *50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, 2012, pp. 1015–1024.

[39] Z. Zhu, D. Bernhard, and I. Gurevych, "A Monolingual Tree-based Translation Model for Sentence Simplification," in *23rd International Conference on Computational Linguistics*, 2010, pp. 1353–61.

[40] T. Lappas and M. Vlachos, "Customizing Search Results for Non-native Speakers," in *21st ACM International Conference on Information and Knowledge Management*, 2012, pp. 1829–1833.

[41] P. S. and M. Ostendorf, "Text Simplification for Language Learners: a Corpus Analysis," in *In Proceedings of the International Speech Communication Association Special Interest Group on Speech and Language Technology in Education*, 2007, pp. 69–72.

[42] S. Maxwell, "The Effects of Two Types of Text Modification on English Language Learners' Reading Comprehension: Simplification versus Elaboration," 2011.

[43] N. Shah, J. Bradley, S. Balakrishnan, A. Parekh, K. Ramchandran, and M. Wainwright, "Some Scaling Laws for MOOC Assessments," in *KDD Workshop on Data Mining for Educational Assessment and Feedback*, 2014.

[44] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, 2000.

[45] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer, "Peer and Self Assessment in Massive Online Classes," *ACM Trans. Comput.-Hum. Interact.*, vol. 20, no. 6, pp. 33:1–33:31, 2013.

[46] D. Dahlmeier, H. T. Ng, and S. M. Wu, "Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English," in *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications*, 2013, pp. 22–31.

[47] Ng, Hwee Tou and Wu, Siew Mei and Wu, Yuanbin and Hadiwinoto, Christian and Tetreault, Joel, "The CoNLL-2013 Shared Task on Grammatical Error Correction," in *Proceedings of the 17th Conference on CoNLL*, 2013.

[48] S. Ishikawa, "The ICNALE and Sophisticated Contrastive Interlanguage Analysis of Asian Learners of English," in *Learner Corpus Studies in Asia and the World*, 2013, pp. 91–118.

[49] Z. Zheng, X. Wu, and R. Srihari, "Feature Selection for Text Categorization on Imbalanced Data," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 80–89, 2004.

[50] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh, "On Smoothing and Inference for Topic Models," in *Proceedings of the 25th Conference on Uncertainty in AI*, 2009, pp. 27–34.

[51] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Found. Trends Inf. Retr.*, vol. 2, no. 1-2, pp. 1–135, 2008.

[52] C. Tillmann and H. Ney, "Word Reordering and a Dynamic Programming Beam Search Algorithm for Statistical Machine Translation," *Comput. Linguist.*, vol. 29, no. 1, pp. 97–133, Mar. 2003.

[53] S. Argamon, M. Koppel, J. W. Pennebaker, and J. Schler, "Automatically Profiling the Author of an Anonymous Text," *Commun. ACM*, vol. 52, no. 2, pp. 119–123, Feb. 2009.

[54] J. P. Pestian, P. Matykiewicz, and M. Linn-Gust, "What's In a Note: Construction of a Suicide Note Corpus," *Biomedical Informatics Insights*, vol. 5, pp. 1–6, 2012.

[55] P. Norvig, *Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp*, 1st ed. Morgan Kaufmann Publishers Inc., 1992.