Correcting Comma Errors in Learner Essays, and Restoring Commas in Newswire Text

Ross Israel
Indiana University
Memorial Hall 322
Bloomington, IN 47405, USA
raisrael@indiana.edu

Joel Tetreault
Educational Testing Service
660 Rosedale Road
Princeton, NJ 08541, USA
jtetreault@ets.org

Martin Chodorow
Hunter College of CUNY
695 Park Avenue
New York, NY 10065, USA
mchodoro@hunter.cuny.edu

Abstract

While the field of grammatical error detection has progressed over the past few years, one area of particular difficulty for both native and non-native learners of English, *comma placement*, has been largely ignored. We present a system for comma error correction in English that achieves an average of 89% precision and 25% recall on two corpora of unedited student essays. This system also achieves state-of-the-art performance in the sister task of restoring commas in well-formed text. For both tasks, we show that the use of novel features which encode long-distance information improves upon the more lexically-driven features used in prior work.

1 Introduction

Automatically detecting and correcting grammatical errors in learner language is a growing sub-field of Natural Language Processing. As the field has progressed, we have seen research focusing on a range of grammatical phenomena including English articles and prepositions (c.f. Tetreault et al., 2010; De Felice and Pulman, 2008), particles in Korean and Japanese (c.f. Dickinson et al., 2011; Oyama, 2010), and broad approaches that aim to find multiple error types (c.f Rozovskaya et al., 2011; Gamon, 2011). However, to the best of our knowledge, there has not been any research published specifically on correcting erroneous comma usage in English (though there have been efforts such as the MS Word grammar checker, and products like Grammarly and White Smoke that include comma checking).

There are a variety of reasons that motivate our interest in attempting to correct comma errors. First of all, a review of error typologies in Leacock et al. (2010) reveals that comma usage errors are the fourth most common error type among non-native writers in the Cambridge Learner Corpus (Nicholls, 1999), which is composed of millions of words of text from essays written by learners of English. The problem of comma usage is not limited to non-native writers; six of the top twenty error types for native writers involve misuse of commas (Connors and Lunsford, 1988). Given these apparent deficits among both non-native and native speakers, developing a sound methodology for automatically identifying comma errors will prove useful in both learning and automatic assessment environments.

A quick examination of English learner essays reveals a variety of errors, with writers both overusing and underusing commas in certain contexts. Consider examples (1) and (2):

(1) **erroneous**: If you want to be a master you should know your subject well.
corrected: If you want to be a master , you should know your subject well.

(2) **erroneous**: I suppose , that it is better to specialize in one specific subject.
corrected: I suppose that it is better to specialize in one specific subject.

In example (1), an introductory conditional phrase begins the sentence, but the learner has not used the appropriate comma to separate the dependent clause from the independent clause. The comma in this case helps the reader to see where one clause ends

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 284–294,
Montréal, Canada, June 3-8, 2012. ©2012 Association for Computational Linguistics

and another begins. In example (2), the comma after *suppose* is unnecessary in American English, and although this error is related more to style than to readability, most native writers would omit the comma in this context, so it should be avoided by learners as well.

Another motivating factor for this work is the fact that sentence internal punctuation contributes to the overall readability of a sentence (Hill and Murray, 1998). Proper comma placement can lead to faster reading times and reduce the need to re-read entire sentences. Commas also help remove or reduce problems arising from difficult ambiguities; the garden path effect can be greatly reduced if commas are correctly inserted after introductory phrases and reduced relative clauses.

This paper makes the following contributions:

- We present the first published comma error correction system for English, evaluated on essays written by both native and non-native speakers of English.
- The same system also achieves state-of-the-art performance in the task of restoring commas in well-edited text.
- We describe a novel annotation scheme that allows for robust mark up of comma errors and use it to annotate two corpora of student essays.
- We show that distance and combination features can improve performance for both the error correction and restoration tasks.

The rest of this paper is organized as follows. In section 2, we review prior work. Section 3 details our typology of comma usage. We discuss our choice of classifier and selection of features in section 4. In section 5, we apply our system to the task of comma restoration. We describe our annotation scheme and error correction system and evaluation in sections 6 and 7. Finally, we summarize and outline plans for future research in section 8.

2 Previous Work

The only reported research that we are aware of which specifically deals with comma errors in learner writing is reported in Hardt (2001) and Alegria et al. (2006), two studies that deal with Danish and Basque, respectively. Hardt (2001) employs an error driven approach featuring the Brill tagger (Brill, 1993). The Brill tagger works as it would for the part-of-speech tagging task for which it was designed, i.e. it learns rules based on templates by iterating over a large corpus. This work is also evaluated on native text where all existing commas are considered correct, and additional "erroneous" commas are added randomly to a sub-corpus, so that the tagger can learn from the errors. The system is tested on a distinct subset for the task of correcting existing comma errors and achieves 91.4% precision and 76.9% recall.

Alegria et al. (2006) compare implementations of Naive Bayes, decision-tree, and support vector machine (SVM) classifiers and utilize a feature set based on word-forms, categories, and syntactic information about each decision point. While the system is designed as a possible means for correcting errors, it is only evaluated on the task of restoring commas in well-formed text produced by native writers. The system obtains good precision (96%) and recall (98.3%) for correctly *not* inserting commas, but performs less well at actually inserting commas (69.6% precision, 48.6% recall).

It is important to note that the results in both of the projects are based on constructed errors in an otherwise native corpus which is free of any other contextual errors that might be present in actual learner data. Moreover, as we will show in section 6, errors of omission (failing to use needed commas) are much more common than errors of commission (inserting commas inappropriately) in the English as a Foreign Language (EFL) data that we use. Crucially, our error correction efforts described in section 7 must be able to account for noise and be able to insert new commas as well as remove erroneous ones, as we do evaluate on a set of English learner essays.

Although we have not found any work published specifically on correcting comma errors in English, for language learners or otherwise, there is a fairly large amount of work that focuses on the task of comma restoration. Comma restoration refers to placing commas in a sentence which is presented with no sentence internal punctuation. This task is mostly attempted in the larger context of Automatic Speech Recognition (ASR), since there are no absolute cues of where commas should be placed in a stream of speech. Many of these systems use feature sets that include prosodic elements that are clearly not available for text based work (see e.g., Favre et al., 2009; Huang and Zweig, 2002; Moniz et al., 2009).

There are, however, a few punctuation restoration projects that have used well-formed text-only data. Shieber and Tao (2003) explore restoring commas to the Wall Street Journal (WSJ) section of the Penn Treebank (PTB). The authors augment a HMM trigram-based system with constituency parse information at each insertion point. Using fully correct parses directly from the PTB, the authors achieve an F-score of 74.8% and sentence accuracy of 57.9%[1]. However, a shortcoming of this methodology is that it dictates that all commas are missing, but these parses were generated with comma information present in the sentence and moreover hand-corrected by human annotators. Using parses automatically generated with commas removed from the data, they achieve an F-score of 70.1% and sentence accuracy of 54.9%.

More recently, Gravano et al. (2009), who work with newswire text, including WSJ, pursue the task of inserting all punctuation and correcting capitalization in a string of text in a single pass, rather than just comma restoration, but do provide results based solely on comma insertion. The authors employ an *n*-gram language model and experiment with *n*-grams from size $n = 3$ to $n = 6$, and with different training data sizes. The result relevant to our work is their comma F-score on WSJ test data, which is just over 60% when using 5-grams and 55 billion training tokens. Baldwin and Joseph (2009) also restore punctuation and capitalization to newswire texts, using machine based learning with retagging. Their results are difficult to compare with our work because they use a different data set and do not focus on commas in their evaluation.

Lu and Ng (2010) take an approach that inserts all punctuation symbols into text. They use transcribed English and Chinese speech data and do not provide specific evaluation for commas, however one important contribution of their research to our current task is the finding that Conditional Random Fields (CRFs) perform better at this task than Hidden Event Language Models, another algorithm that has been used for restoration. One reason for this could be CRFs' better handling of long range dependencies because they model the entire sequence, rather than making a singular decision based on information at each point in the sequence (Liu et al., 2005). CRFs also do not suffer from the label bias problem that affects Maximum Entropy classifiers (Lafferty et al., 2001).

3 Comma Usage

One of the challenges present in this research is the ambiguity as to what constitutes "correct" comma usage in American English. For one thing, not all commas contribute to grammaticality; some are more tied to stylistic rules and preferences. While there are certainly rule-based decision points for comma insertion (Doran, 1998), particularly in the case of commas that set off significant chunks or phrases within sentences, there are also some commas that appear to be more prescriptive, as they have less of an effect on sentence processing (such as in example (2) in the introduction), and opposing usage rules for the same contexts are attested in different style manuals. A common example of opposing rules is the notorious serial or Oxford comma that refers to the final comma found in a series, which is required by the Chicago Manual of Style (University of Chicago, 1993), but is considered incorrect by the New York Times Manual of Style (Siegal and Connolly, 1999).

As a starting point, we needed to know what kinds of commas are taught by English language teachers, as well as what style manuals recommend and/or require. However, creating a list of comma uses was a non-trivial part of the process. After consulting style manuals (University of Chicago, 1993; Siegal and Connolly, 1999; Strunk and White, 1999) and popular ESL websites, we compiled a list of over 30 rules for use of commas in English. We took the most commonly mentioned rules and created a final

[1] Sentence accuracy is a measure used by some in the field that counts sentences with 100% correct comma decisions as correct, and any sentence where a comma is missing or mistakenly placed as incorrect. It is motivated by the idea that all commas are essential to understanding a sentence.

Rule	Example
Elements in a List	*Paul put the kettle on, Don fetched the teapot, and I made tea.*
Initial Word/Phrase	*Hopefully, this car will last for a while.*
Dependent Clause	*After I brushed the cat, I lint-rollered my clothes.*
Independent Clause	*I have finished painting, but he is still sanding the doors.*
Parentheticals	*My father, a jaded and bitter man, ate the muffin.*
Quotations	*“Why,” I asked, “do you always forget to do it?”*
Adjectives	*She is a strong, healthy woman.*
Conjunctive Adverbs	*I would be happy, however, to volunteer for the Red Cross.*
Contrasting Elements	*He was merely ignorant, not stupid.*
Numbers	*345,280,000*
Dates	*She met her husband on December 5, 2003.*
Geographical Names	*I lived in San Francisco, California, for 20 years.*
Titles	*Al Mooney, M.D., is a good doctor*
Introducing Words	*You may be required to bring many items, e.g., spoons, pans, and flashlights.*
Other	Catch-all rule for any other comma use

Table 1: Common Comma Uses

list of 15 usage rules (the 14 most common plus one miscellaneous category) for our annotation scheme, which is discussed in section 6. These rules are given in Table 1. The 16 rules that were removed from the list occurred in only one source or were similar enough to other rules to be conflated. It is worth noting here that while many of the comma uses in this table might be best served by some statistical methodology like the one we describe in section 4, one can envision fairly simple heuristic rules to insert commas and find errors in numbers, dates, geographical names, titles, and introducing words.

4 Classifier and Features

We use CRFs[2] as the basis for our system and treat the task of comma insertion as a sequence labeling task; each space between words is considered by the classifier, and a comma is either inserted or not. The feature set incorporates features that have proven useful in comma restoration and other error correction tasks, as well as a handful of new features devised for this specific task (combination and distance features). The full set of features used in our final system is given in Figure 1 along with examples of each feature for the sentence *If the teacher easily gets mad , then the child will always fear going to school and class*. The target insertion point is after the word *mad*.

Feature	Example(s)
Lexical and Syntactic Features	
unigram	easily, gets, mad, then, the
bigram	easily gets, gets mad, mad then, ...
trigram	easily gets mad, gets mad then, ...
pos_uni	RB, VBZ, JJ, RB, DT
pos_bi	RB VBZ, VBZ JJ, JJ RB, ...
pos_tri	RB VBZ JJ, VBZ JJ RB, ...
combo	easily+RB, gets+VBZ,mad+JJ, ...
first_combo	If+RB
Distance Features	
bos_dist	5
eos_dist	10
prevCC_dist	-
nextCC_dist	9

Figure 1: CRF Features with examples for:
If the teacher easily gets mad , then the child will always fear going to school and class.

4.1 Lexical and Syntactic Features

The first six features in Figure 1 refer to simple unigrams, bigrams, and trigrams of the words and POS tags in a sliding 5 word window (target word, +/- 2 words). The lexical items help to encode any idiosyncratic relationships between words and commas that might not be exploited through the examination of more in-depth linguistic features. For example, *then* is a special case of an adverb (RB) that is often preceded by a comma, even if other adverbs are not, so POS tags might not capture this relation-

[2] `http://crfpp.sourceforge.net/`

ship. The lexical items also provide an approximation of a language model or hidden event language model approach, which has proven to be useful in comma restoration tasks (see e.g. Lu and Ng, 2010).

The POS features abstract away from the words and avoid the problem of data sparseness by allowing the classifier to focus on the categories of the words, rather than the lexical items themselves. The combination (combo) feature is a unigram of the word+pos for every word in the sliding window. It reinforces the relationship between the lexical items and their POS tags, further strengthening the evidence of entries like *then_RB*. All of these features have been used in previous grammatical error detection tasks which target particle, article, and preposition errors (c.f., Dickinson et al., 2011; Gamon, 2010; Tetreault and Chodorow, 2008).

The first_combo feature keeps track of the first combination feature of the sentence so that it can be referred to by the classifier throughout processing the entire sentence. This feature is helpful when an introductory phrase is longer than the classifier's five word window. Figure 1 provides a good example of the utility of this feature, as *If the teacher easily gets mad* is so long that by the time the window has moved to the target position of the space following *mad*, the first word and POS, *If_RB*, which can often indicate an introductory phrase, is beyond the scope of the sliding window.

4.2 Distance Features

Next, we encode four distance features. We keep track of the following distances: from the beginning of the sentence (bos_dist), to the end of the sentence (eos_dist), from the previous coordinating conjunction (prevCC_dist), and to the next coordinating conjunction (nextCC_dist). All of these distance features help the classifier by encoding measures for components of the sentence that can affect the decision to insert a comma. These features are especially helpful over long range dependencies, when the information encoded by the feature is far outside the scope of the 5-word window the CRF uses. The distance to the beginning of the sentence helps to encode introductory words and phrases, which make up the bulk of the commas used in essays by learners of English. The distance to the end of the sentence is less obviously useful, but it can let the classifier know the likelihood of a phrase beginning or ending at a certain point in the sentence. The distances to and from the nearest CC are useful because many commas are collocated with coordinating conjunctions. The distance features, as well as first_combo, were designed specifically for the task of comma error correction, and have not, as far as we know, been utilized in previous research.

5 Comma Restoration

Before applying our system to the task of error correction, we tested its utility in restoring commas in newswire texts. Specifically, we evaluate on section 23 of the WSJ, training on sections 02-22. Here, the task is straightforward: we remove all commas from the test data and performance is measured on the system's ability to put the commas back in the right places. After stripping all commas from our test data, the text is tokenized and POS tagged using a maximum entropy tagger (Ratnaparkhi, 1996) and every token is considered by the classifier as either requiring a following comma or not. Out of 53,640 tokens, 3062 should be followed by a comma. We provide accuracy, precision, recall, F_1-score, and sentence accuracy (S Acc.) for these tests, along with results from Gravano et al. (2009) and Shieber and Tao (2003) in Table 2. The first system (LexSyn) includes only the lexical and syntactic features from Figure 1; the second (LexSyn+Dist) includes all of the features.

System	Acc.	P	R	F	S Acc.
LexSyn	97.4	85.8	64.9	73.9	60.5
LexSyn+Dist	**97.5**	**85.8**	66.3	**74.8**	**61.4**
Shieber & Tao	97.0	79.7	62.6	70.1	54.9
Gravano et al.	N.A.	57	**67**	≈61	N.A.

Table 2: Comma Restoration System Results (%)

As can be seen in Table 2, the full system (LexSyn+Dist) performs significantly better than WSJ LexSyn ($p < .02$, two-tailed), achieving an F-score of 74.8 on WSJ. This F-score outperforms Shieber and Tao's system, which was also tested on section 23 of the WSJ, by about 4% and our sentence accuracy of 61.5% is about 7% higher than theirs. Our F-score is also about 13% higher than that of Gravano et al. (2009), however, they evaluate on the entire WSJ section of the Penn Treebank, so it is not totally fair to compare results.

6 Annotation

For the comma restoration task, we needed only to obtain well-formed text and remove the commas to produce a test set. However, this is not so in the case of error correction. In order to test a system that corrects errors in learner essays, we need an annotated test corpus that tells us where the errors are. Although there are a handful of corpora that include punctuation errors in their annotation scheme, such as NUCLE (Dahlmeier and Ng, 2011) and HOO (Dale and Kilgarriff, 2010), there are none to our knowledge that focus specifically on commas. Thus, we designed and implemented our own annotation scheme on a set of essays to allow us the freedom to identify the most important aspects of comma usage for our work.

Our annotation scheme allows the mark-up of a number of aspects of comma usage. First, each comma in a text is marked as rejected or accepted by the annotator. Additionally, any space between words can be treated as an insertion point for a missing comma. The annotators also marked all accepted and inserted commas as either required or optional. Finally, the annotation also includes the appropriate usage rule from the set in Table 1.[3] In contrast, the NUCLE and HOO data sets do not have this granularity of information (the annotation only indicates whether a comma should be inserted or removed) and are not exhaustively annotated.

After a one-hour training session on comma usage rules, three native English speakers were given a set of ten learner essays comprising 3,665 tokens to annotate for comma errors. To assess the difficulty of the annotation task, we calculated agreement and kappa. Agreement is a simple measure of how often the annotators agree, and kappa provides a more robust measure of agreement since it takes chance into account (Cohen, 1960). Table 3 provides the results of these measurements. As can be seen in the table, the agreement is quite high at either 97 or 98%, and kappa is a bit lower, ranging from 72 to 81%. The agreement is likely so high due to the great number of decision points where it is obvious to any native writer that no comma is needed. To account for this imbalance, we also provide an adjusted agreement in the final column of the table that excludes all decisions where both annotators agree that no comma is necessary.

Annotators	Agreement	Kappa	Adj. agr.
1 & 2	97	74	61
1 & 3	98	72	61
2 & 3	98	81	76

Table 3: Agreement over Annotation Training Set (%)

After completing the training phase, we assigned one annotator the task of annotating our development and test data from two different corpora: essays written by English as a foreign language learners (EFL) and essays written by native speakers of English (Native). For both data sets we selected 60 essays for development and 60 essays for test. The annotation was carried out using an annotation tool developed in-house that gives the annotator an easy to use interface and outputs standoff annotations in xml format. (3) is an example of an annotated sentence from an EFL essay, where “$_{\times}$” marks a span for annotation.

(3) The new millenium ,$_{1}$ the 21st century $_{2}$ has dawned upon us $_{3}$ and this new century has brought many positive advancements in our daily lives .
1) Accept, required, parenthetical
2) Insert, required, parenthetical
3) Insert, required, independent clause

Table 4 provides the comma usage information for the essays in both sets used in development and testing. The table shows the total number of sentences, commas in the original text that were accepted by the annotator, and errors (rejected and missing commas) for the 60 essays in each set.

As can be seen in Table 4, the majority of existing commas (columns *Accept* plus *Rej*) in the texts were accepted by the annotator; about 84% in the EFL development set, 87% in the EFL test set, 85% in the Native development set, and 88% in the Native test set. The important fact uncovered by these numbers is that most of the commas that learners do

[3] The full annotation manual is available at http://www.cs.rochester.edu/~tetreaul/comma-manual.pdf

Data Set	Sent	Commas		
		Accept	Errors	
			Rej	Miss
EFL Dev	717	474	49	233
EFL Test	683	427	65	232
Native Dev	970	506	86	363
Native Test	839	377	50	314

Table 4: Comma Usage Statistics

use are correct. However, there are a great number of commas that the annotator inserted (over 80% of all errors are missing commas) meaning that these learners are more prone to underusing than overusing commas. Another interesting fact that can be gleaned from our annotation is that the top five comma uses, those listed in the first five rows of Table 1, account for more than 80% of all commas in these essays.

7 Error Correction

With a competitive comma restoration system in place, we turn to the primary task of correcting errors in learner essays. While the task remains similar to comma restoration, error correction in student writing brings a new set of challenges, especially when the writers are non-native. Newswire texts are most often well-formed, so the system should not experience interference from other contextual errors around the missing commas. Sentences taken from learner texts, though, often contain multiple errors that can make it difficult to focus on a single problem at a time. Spelling errors, for example, can exacerbate error correction efforts that use contextual lexical features because well-formed text that is often used for training data is usually free of such noise.

In these experiments, we use the annotated essays described in section 6 for evaluation and train on 40,000 sentences taken from essays written by both native and non-native high level college students. All of the essays are run through automatic spelling correction to reduce the noise in the test set before being tagged with the same tagger used in the comma restoration experiments.

Because we approach comma error correction as essentially a comma restoration task, we can we use largely the same system for error correction as we did for comma restoration. We still employ CRFs and label each space between words as requiring a comma or not, however, there is one significant change to our methodology for this task. Namely, we can leave the commas that were present in the text as provided by the writer as we pre-process the data for error correction, whereas they were removed in the comma restoration task. For error correction, the task is really comparing the system's answer to the annotator's and the learner's, as opposed to simply inserting commas into raw text. Leaving the learners' commas in the text does introduce some errors to the POS tagging phase. However, since over 85% of the existing commas in the development set were judged as acceptable by our annotator (cf. section 6) , the number of erroneous commas is not so great as to contaminate the system. Removing all of the commas would introduce unnecessary errors in the pre-processing phase.

We also augment the system with three post-processing filters that we tuned on the development set. One requires that the classifier be completely confident before a change is made to an existing comma; crf++ will give 100% confidence to a single class in some cases. This filter is based on the fact that 85% of the existing commas can be expected to be correct. A similar filter requires that the classifier be at least 90% confident in a decision to insert a new comma. The final filter, which overrides any other information provided by the system, does not allow commas to be inserted before the word *because*. These ensure high precision even though they may reduce recall.

Table 5 provides the accuracy, precision, recall, F-score, and number of errors in each set for tests on our 60 annotated EFL and Native essays, and the result for the combined corpus. The system performs quite well on the EFL test set, with scores of 94% precision, 31.7% recall, and 47.4% F-score for the LexSyn+Dist system. The results for the Native set are a bit lower, with 84.9% precision, 20% recall, and 32.4% F-score for the LexSyn+Dist system.

For both data sets, when the distance features are added to the model, precision increases by 1%, and in the EFL set, recall also increases. In keeping with practices established within the field of grammatical error correction, the system has been optimized for high precision even at the cost of recall, to ensure that feedback systems avoid confusing learners by

Data	System	Acc.	P	R	F	n
EFL	LexSyn	98.2	92.9	30.9	46.5	297
	LexSyn+Dist	98.3	**94.0**	**31.7**	**47.4**	297
Native	LexSyn	97.8	83.9	20.0	32.3	365
	LexSyn+Dist	97.8	84.9	20.0	32.4	365
Combined	LexSyn	98.1	88.7	24.9	38.9	662
	LexSyn+Dist	98.1	89.8	25.2	39.4	662

Table 5: Comma Error Correction Results (%)

marking correct comma usage as erroneous. Considering performance over all of the test data, the system achieves over 89% precision and 25% recall, results which are comparable to those in other error correction tasks. For example, the preposition error detection system described in Tetreault and Chodorow (2008) achieved 84% precision, 19% recall for prepositions.

It is worth noting that the results in Table 5 include commas that the annotator had marked as optional. For these, whatever decision the system makes is scored as correct. Since the grammaticality/readability of the sentence will not be affected by the presence or absence of a comma in these cases, we feel this is the fairest assessment of the system.

7.1 Error Analysis

In order to get a sense of what kinds of constructions are difficult for our system, we randomly extracted 50 sentences from the output that exhibited at least one wrong comma decision made by the system. The 50 sentences contained a combined total of 62 system errors. Among these cases, the most common context where the system makes the wrong decision is in introductory words and phrases, which is not surprising given the frequency with which commas occur in these environments in our development set (about 40% of all commas in the essays). In (4), for example, the first word, *Here*, should be followed by a comma. Since *Here* is not a common introductory word in this type of sentence structure in the training data, this is a difficult case for the system to correct.

(4) *Here we can get specific knowledge in the science that we like the most .*

The next most common misclassification involves comma splices, i.e. conjoining complete sentences with a comma rather than separating them with a full stop. In (5), for example, there should be a full stop between *college* and *I*, rather than a comma. This result is not surprising because the system is not yet equipped to deal with comma splices. Comma splices are a different type of phenomenon because correcting them requires removing the comma and inserting a full stop, essentially two separate steps rather than the single reject/accept step that the system currently handles.

(5) *I entered college, I could learn it and make an effort to achieve my goal.*

The next most common context for system errors was between clauses that are conjoined with a coordinating conjunction as in (6), where there should not be a comma. In (6), the second clause is actually a dependent clause, so no comma should precede the coordinating conjunction. There are a number of system errors dealing with commas between two independent clauses. For example in (7), our annotator recommended a comma between *things* and *but*, however the system did not make the insertion. The problem with these examples likely stems from the fact that the rule for comma usage in these contexts is not clearly stated, even in well-respected manuals, and therefore likely not clearly understood, even by high-level native writers. For example, the NYT style manual (Siegal and Connolly, 1999) states that "Commas should be used in compound sentences before conjunctions... When the clauses are exceptionally short, however, the comma may be omitted." Adding a feature that measures clause length might help, but even then the classifier must rely on training data that may have considerable variation as to what length of clauses requires an intervening comma.

(6) *They wants to see their portfolio, and what kind of skill do they have for company.*

(7) *I have many things but the best is my parents.*

Another facet of the data that consistently challenges the system is the existence of errors other than the commas in the sentences. Consider the sentence in (8), where **erroneous** is the original text from the essay and **corrected** is a well-formed interpretation.

(8) **erroneous:** *In the other hand , having just one specific subject , which represents a great downfall for many students*
corrected: *On the other hand, knowing only one subject is a downfall for many students.*

The comma after *subject* is unnecessary, but so is the word *which*. In fact, *which* would normally signify the beginning of a non-restrictive clause in this context, which should be set off with a comma. It is no surprise then, that the system has trouble removing commas in these types of contexts. At least 11 of the 62 system mistakes that we examined have grammatical errors in the immediate context of the comma in question, which makes the classification more difficult.

8 Summary and Conclusion

We presented a novel comma error correction system for English that achieves an average of 89% precision and 25% recall on essays written by learners of different levels and language backgrounds, including native English speakers. The system achieves state-of-the-art performance on the task of comma restoration, beating previous systems' F-score and sentence accuracy by 4% and 7%, respectively. We discovered that augmenting lexical features, which have been commonly used in previous work, with the combination and distance features can improve F-score by as much as 1% in both the error correction and comma restoration tasks. We also developed and implemented a novel comma error annotation scheme.

Additionally, we are interested in the effect of correct comma placement on other NLP processes. Jones (1994) and Briscoe and Carroll (1995) show that adding punctuation to grammars that utilize part-of-speech (POS) tags, rather than lexical items, adds more structure and reduces ambiguity as well as the number of parses for each sentence. Similarly, Doran (1998) and White and Rajkumar (2008) found that adding punctuation improved parsing results in tree-adjoining grammar (TAG) and combinatorial categorial grammar (CCG) parsing, respectively. These studies all highlight the importance of correctly inserted punctuation, especially commas, for parsing. Given these results, we believe that by enhancing the quality of the text, comma error correction will improve not only tagging and parsing, but also the ability of systems to correct many other forms of grammatical errors, such as those involving incorrect word order, number disagreement, and misuse of prepositions, articles, and collocations.

Acknowledgments

We would like to thank Melissa Lopez for help with annotating our corpora and Michael Flor for kindly developing an annotation tool for our purposes. We also thank Aoife Cahill, Robbie Kantor, Markus Dickinson, Michael Heilman, Nitin Madnani, and our anonymous reviewers for insightful comments and discussion.

References

Iñaki Alegria, Bertol Arrieta, Arantza Diaz de Ilarraza, Eli Izagirre, and Montse Maritxalar. 2006. Using machine learning techniques to build a comma checker for Basque. In *Proceedings of the COLING/ACL main conference poster sessions*.

Timothy Baldwin and Manuel Paul Anil Kumar Joseph. 2009. Restoring punctuation and casing in English text. In *Australasian Conference on Artificial Intelligence'09*.

E. Brill. 1993. *A Corpus-Based Approach to Language Learning*. Ph.D. thesis, The University of Pennsylvania, Philadelpha, PA.

Ted Briscoe and John Carroll. 1995. Developing and evaluating a probabilistic LR parser of part-of-speech and punctuation labels. In *Proceedings of the ACL/SIGPARSE 4th International Workshop on Parsing Technologies*.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Robert J. Connors and Andrea A. Lunsford. 1988. Frequency of formal errors in current college writing, or Ma and Pa Kettle do research. *College Composition and Communication*, 39(4).

Daniel Dahlmeier and Hwee Tou Ng. 2011. 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*. Association for Computational Linguistics.

Robert Dale and Adam Kilgarriff. 2010. Helping our own: Text massaging for computational linguistics as a new shared task. In *International Conference on Natural Language Generation*.

Rachele De Felice and Stephen Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. In *Proceedings of COLING-08*. Manchester.

Markus Dickinson, Ross Israel, and Sun-Hee Lee. 2011. Developing methodology for Korean particle error detection. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*. Portland, Oregon.

Christine Doran. 1998. *Incorporating Punctuation into the Sentence Grammar: A Lexicalized Tree-Adjoining Grammar Perspective*. Ph.D. thesis, University of Pennsylvania.

Benoit Favre, Dilek Hakkani-Tur, and Elizabeth Shriberg. 2009. Syntactically-informed models for comma prediction. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*.

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing: A meta-classifier approach. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Michael Gamon. 2011. High-order sequence modeling for language learner detection high-order sequence modeling for language learner error detection. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*.

Agustin Gravano, Martin Jansche, and Michiel Bacchiani. 2009. Restoring punctuation and capitalization in transcribed speech. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*.

Daniel Hardt. 2001. Comma Checking in Danish. In *Corpus Linguistics*.

Robin L. Hill and Wayne S. Murray. 1998. Commas and spaces: The point of punctuation. In *11th Annual CUNY Conference on Human Sentence Processing*.

Jing Huang and Geoffrey Zweig. 2002. Maximum entropy model for punctuation annotation from speech. In *Proceedings of ICSLP 2002*.

Bernard E. M. Jones. 1994. Exploring the role of punctuation in parsing natural text. In *Proceedings of the 15th conference on Computational linguistics - Volume 1*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel R. Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, and Mary Harper. 2005. Comparing hmm, maximum entropy, and conditional random fields for disfluency detection. In *In Proceeedings of the European Conference on Speech Communication and Technology*.

Wei Lu and Hwee T. Ng. 2010. Better punctuation prediction with dynamic conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Helena Moniz, Fernando Batista, Hugo Meinedo, and Alberto Abad. 2009. Prosodically-based automatic segmentation and punctuation. In *Pro-*

ceedings of the 5th International Conference on Speech Prosody.

Diane Nicholls. 1999. The cambridge learner corpus - error coding and analysis for writing dictionaries and other books for english learners. In *Summer Workshop on Learner Corpora*. Showa Woman's University.

Hiromi Oyama. 2010. Automatic error detection method for japanese particles. *Polyglossia*, 18.

Adwait Ratnaparkhi. 1996. A Maximum Entropy Model for Part-Of-Speech Tagging. In Eric Brill and Kenneth Church, editors, *Proceedings of the Empirical Methods in Natural Language Processing*.

Alla Rozovskaya, Mark Sammons, Joshua Gioja, and Dan Roth. 2011. University of Illinois system in HOO text correction shared task. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 263–266. Association for Computational Linguistics, Nancy, France.

Stuart M. Shieber and Xiaopeng Tao. 2003. Comma restoration using constituency information. In *Proceedings of the 2003 Human Language Technology Conference and Conference of the North American Chapter of the Association for Computational Linguistics*.

Allan M. Siegal and William G. Connolly. 1999. *The New York Times Manual of Style and Usage : The Official Style Guide Used by the Writers and Editors of the World's Most Authoritative Newspaper*. Crown, rev sub edition.

William Strunk and E. B. White. 1999. *The Elements of Style, Fourth Edition*. Longman, fourth edition.

Joel Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of COLING-08*. Manchester.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of the ACL 2010 Conference Short Papers*.

University of Chicago. 1993. *The Chicago Manual of Style*. University Of Chicago Press, Chicago, fourteenth edition.

Michael White and Rajakrishnan Rajkumar. 2008. A more precise analysis of punctuation for broad-coverage surface realization with CCG. In *Proceedings of the Workshop on Grammar Engineering Across Frameworks*.