Automatic Detection of Comma Splices

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

Martin Chodorow
Department of Psychology
Hunter College
City University of New York
martin.chodorow@hunter.cuny.edu

Abstract

In English text, independent clauses should be demarcated with full-stops (periods), or linked together with conjunctions. Non-native speakers are often prone to linking them improperly with commas instead of conjunctions, producing comma splices. This paper describes a method to detect comma splices using Conditional Random Fields (CRF), with features derived from parse tree patterns. In experiments, our model achieved an average of 0.91 precision and 0.28 recall in detecting comma splices, significantly outperforming both a baseline model using only local features and a widely used commercial grammar checker.

1 Introduction

English text consists of a sequence of clauses linked and separated by punctuation and conjunctions. To separate two independent clauses, one uses a full-stop (period); to link together two related clauses, one typically uses a semicolon or a comma with an appropriate conjunction, which can be either coordinate ("and", "but", "or") or subordinate ("because", "so"). For example, to link the two related clauses "it was raining" and "we stayed home", one may use a comma and the conjunction "so", yielding the complex sentence "It was raining, so we stayed home". When a comma is used instead of a full-stop, or when it is used without a conjunction (e.g., "It was raining, we stayed home"), the result is a comma splice[1].

[1]Note that a list of noun phrases with a missing conjunction (e.g., "I like apples, oranges.") is not a comma splice.

Our use of the term comma splice also includes improper linking of verb phrases. This occurs when a comma is used without a conjunction (e.g., "We stayed home, watched TV."); without a relative pronoun (e.g., "The boy chased after the rat, fled into the sewer"); or with the wrong verb form (e.g., "Waterborne pathogens are the pathogenic microorganisms, includes bacteria"). Comma splices are not only considered poor writing style, but they also compromise the readability of a text.

Although native speakers have been found to commit a substantial number of common splice errors (Connors and Lunsford, 1988; Lunsford and Lunsford, 2008), non-native speakers appear to be especially prone to producing them, possibly due to interference from syntactic differences in L1 (Tseng and Liou, 2006; Bennui, 2008; Rahimi, 2009). This may be especially true for L1s where comma splices are frequently found and are not considered mistakes, such as in Chinese (Lin, 2002). Comma splices are one of the errors addressed in the 2014 CoNLL Shared Task on Grammatical Error Correction (Ng et al., 2014). They are annotated in many learner corpora, including the NUS Corpus of Learner English (Dahlmeier et al., 2013) and the EF-Cambridge Open Language Database (Geertzen et al., 2013).

This paper addresses the task of detecting comma splices. We report human agreement in detecting these errors and propose a CRF model to automatically detect them. Our best model, which uses features derived from parse trees produced by the Stanford parser (Klein and Manning, 2003), significantly outperforms both a baseline that does not consider

28th Pacific Asia Conference on Language, Information and Computation pages 551–560

syntactic information and a widely used commercial grammar checker.

Recently, there has been much effort in developing writing assistance systems that can automatically correct errors in text written by non-native speakers. Such systems focus mostly on word or phrase-level errors, such as the misuse of articles (Han et al., 2006), prepositions (Tetreault et al., 2010) and verbs (Tajiri et al., 2012). Although these errors do involve long-distance grammatical constructions, this paper is the first report of a research effort to address the improper linking of clauses, a sentence-level error.

Our ultimate goal, after detecting a comma splice, is to automatically correct it. We will, however, not treat the correction task here because it concerns a host of other issues, such as automatic analysis of style, to choose between splitting the comma splice into two sentences ("It was raining. We stayed home.") and conjoining them ("It was raining, so we stayed home"), as well as inference of discourse relations (Marcu and Echihabi, 2002), to choose an appropriate conjunction (e.g., use of "so" rather than "because" in the above example).

The rest of the paper is organized as follows. After reviewing previous research in related areas (section 2), we describe our approach for comma splice detection (section 3). We then describe our datasets, report on human agreement and experimental results (section 4), followed by our conclusions (section 5).

2 Previous work

A comma splice may be the result of a misuse of punctuation (comma instead of full-stop), a misuse of verb form (finite instead of participle), or a missing conjunction. Hence, our work can draw on previous research on detecting and correcting punctuation, verb and conjunction errors.

Automatic punctuation restoration had originally been applied on output of automatic speech recognition systems (Stolcke and Shriberg, 1996; Huang and Zweig, 2002), but has more recently been expanded to written text (Gravano et al., 2009; Baldwin and Joseph, 2009). These techniques can be used to detect fused sentences, which result "when a writer puts no mark of punctuation and no coordinating conjunction between independent clauses" (Hacker and Sommers, 2011), a phenomenon also common to ESL writers but distinct from comma splices.

A more related task is the correction of comma usage, an error type that ranks first in ESL writing (Donahue, 2001). The task of inserting missing commas and deleting unnecessary ones has been approached as a sequence labelling problem (Israel et al., 2012), where each space between words was considered by a CRF model to determine whether a comma should be present. Features such as POS, bi-grams, and distances to the nearest conjunctions were effective and these will form the basis of our baseline model. The comma errors addressed in Israel et al. (2012), however, are distinct from ours. Instead of adding in missing commas or deleting unnecessary ones, our focus is on the improper linking of clauses that manifests as wrongly used commas, which cannot be fixed by simply removing them.

Work by Lee and Seneff (2008) on correcting the misuse of verb forms is relevant to detecting comma splice errors that involve participles. They found that verb form errors result in predictable irregularities in parse trees which can be used as cues for error detection. We follow their approach of using parse tree patterns, but will incorporate these patterns in a machine learning framework rather than a rule-based system.

We are not aware of any previous work on detecting or restoring missing conjunctions, but this task is implicitly or explicitly performed by four existing systems that give feedback about comma splices. The Criterion Online Writing Service (Burstein et al., 2004) identifies errors, including comma splices, in student essays and suggests possible corrections. Grammarly[2] scans a paragraph of text and suggests "punctuation between clauses" when comma splices are detected. WhiteSmoke[3] underlines the problematic comma and suggests that it should be replaced with either a full-stop or a semi-colon. The grammar checker embedded in Microsoft Word, perhaps the most widely used system, also gives feedback about comma splices. To the best of our knowledge, the first three do not explicitly consider parse tree patterns; we will evaluate our approach against the

[2] www.grammarly.com

[3] www.whitesmoke.com

fourth.

In addition to these four, a number of writing assistance systems have also been built for the two *Helping Our Own* shared tasks (Dale and Kilgarriff, 2011; Dale et al., 2012) and two *CoNLL* shared tasks (Ng et al., 2013; Ng et al., 2014). Run-on sentences and comma splices were among the 28 error types introduced in the CoNLL-2014 shared task (Ng et al., 2014). Among teams that tackled individual error types, none addressd run-on sentences and comma splices. Among teams that attempted to correct all error types, many obtained good results for word- and phrase-level errors, but none achieved any recall for run-on errors and comma splices.

3 Approach

We cast comma splice detection as a sequence labeling task, using a linear-chain CRF as our model. Each comma in a sentence is to be tagged as `T[rue]` (it is a comma splice) or `F[alse]` (it is not). Consider the sentence "Then, he chased after the rat, fled into the sewer, and died." It should be labeled as `FTF`, since only the second comma constitutes a comma splice (the relative pronoun "which" should follow the comma). In our datasets, consecutive comma splices are relatively uncommon; this preference can be captured by transition features in the linear-chain CRF.

Table 1 shows our list of features. The baseline features replicate those in (Israel et al., 2012); there are then four "clause features" indicating linguistics characteristics of the neighboring clauses[4], but without considering syntactic parse trees; finally, there are five features derived from parse tree patterns.

3.1 Baseline features

Our baseline features include the first word in the clause preceding the comma and the two words to the left and right of the comma, together with their POS and a combined feature with both the word and its POS. We also include the word and POS bigrams of the tokens to the left and right of the comma. In addition, there are four distance features: the number of tokens in the clauses preceding and following the comma, and the distances from the comma to the nearest conjunction to its left and right. All of these can be obtained without syntactic parsing.

3.2 Clause features

We identified four additional features that help prevent the system from flagging the commas around non-restrictive clauses as comma splices (e.g., "The powder diffractometer, Siemens D500, was used in this experiment."; and "The insurance industry, however, is now suffering."), thereby reducing the number of false positives. These features include the number of nouns/pronouns in the clauses preceding and following the comma, and two binary features that indicate whether the clauses contain any verbs. We selected these features because the addition of non-restrictive clauses in the middle of the sentences often results in segments of words without verbs or nouns.

3.3 Parse features

When a sentence contains two or more improperly joined clauses, its parse tree will be "disturbed" because the missing linkage prevents the parser from properly processing the clauses after the first comma. We identified several parse patterns that are characteristic of comma splices, as shown in Table 2.

A comma splice may consist of two improperly joined clauses (e.g., "It was raining, we stayed home"), which tend to produce a parse tree with an S, followed by the comma, an NP and a VP to its right (Pattern S+NP+VP)[5]. Three or more improperly joined clauses (e.g., "The pink shirt is $20, black skirt is $18, dark pant is $15.") tend to result in a parse tree with multiple S siblings (Pattern S+S). A comma splice may also involve improperly joined VPs (e.g., "It can help salesperson to promote up-sales and cross sales, provide better services."), which tend to produce a parse tree with two VP siblings immediately below another VP (Pattern VP+VP). In addition, two binary features for partial pattern matches are included: whether there is an S in the clause to the left of the comma, and whether

[4] We use the term 'clause' here to refer to all words between the comma and the nearest comma to its left or right.

[5] This pattern also appears when the first half of the sentence is a participial phrase that modifies the rest of the sentence. The pattern is therefore ignored if the sentence begins with either a present participle or a past participle.

there is an NP followed by a VP in the clause to the right.

Accurate extraction of parse features depends on the quality of the parse trees, but non-native errors in the sentence often cause the parser to produce unexpected tree patterns (Foster et al., 2008), hence causing noise in the parse tree features. In general, parsers perform better on shorter sentences. To reduce this kind of interference, therefore, we remove those parts of the sentence that cannot contain comma splices.

Unlike the task of sentence compression for summarization (Knight and Marcu, 2000; Filippova and Strube, 2008), we do not need to preserve important words or the meaning of the original sentence. Rather, we aim to preserve the phrases in the sentence that can potentially result in comma splices and strip away the rest so that the parser has the best chance to produce the expected parse patterns.

Specifically, using the parse tree of the original sentence, we remove (1) introductory phrases at the beginning of a sentence, which include transition phrases such as "for example", as well as prepositional phrases and adverbials[6]; (2) clauses that are properly connected to the rest of the sentence by a coordinate conjunction; (3) subordinate clauses that are properly connected to the rest of a sentence by subordinate conjunctions or relative pronouns; and (4) dialogue tags such as "he claimed" or "he argued"[7]. The simplified sentence is then re-parsed before feature extraction.

4 Experiments

We first describe our datasets (sections 4.1 and 4.2) and report on the human agreement on comma splices (section 4.3), and then we discuss our experimental results (section 4.4).

4.1 Training Set

We automatically produced training data from the Penn Treebank (Marcus et al., 1993). While in-domain training data is likely to yield better performance, we chose to use only general-domain training data in our experiments so as to provide a realistic estimate of system performance on arbitrary learner text.

Similar to (Foster and Andersen, 2009), we artificially introduced comma splices into the text by removing conjunctions and relative pronouns to the right of commas. To ensure that the generated sen-

Feature	Example
Baseline features	
Left words	raining, was
Left POS	VBG, VBD
Left combo	raining_VBG, was_VBD
Right words	we, stayed
Right POS	PRP, VBD
Right combo	we_PRP, stayed_VBD
First word in left clause	it
First POS in left clause	PRP
First combo in left clause	it_PRP
Left word bigram	was_raining
Right word bigram	we_stayed
Left POS bigram	VBD_VBG
Right POS bigram	PRP_VBD
# tokens in left clause	3
# tokens in right clause	3
Distance to nearest left conjunction	-
Distance to nearest right conjunction	-
Clause features	
# nouns/pronouns in left clause	1
# nouns/pronouns in right clause	2
has verb in left clause	yes
has verb in right clause	yes
Parse features	
Pattern S+S	no
Pattern S+NP+VP	yes
Pattern VP+VP	no
S in left clause	yes
NP and VP in right clause	yes

Table 1: List of features. Example values for each feature are drawn from the comma of the sentence "It was raining, we stayed home".

[6]The list of phrases are taken from http://www.msu.edu/user/jdowell/135/transw.html

[7]We used a list of 292 verbs that are the hyponyms of the words "express" and "convey" in WordNet 3.0 (Miller, 1995).

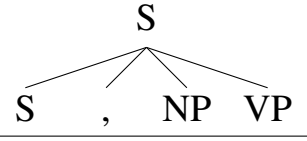
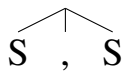
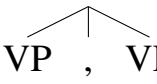
Feature	Pattern	Example
Pattern S+NP+VP		[S][It was raining], [NP][we] [VP][stayed home.]
Pattern S+S		[S][The pink shirt is $20], [S][black skirt is $18], [S][dark pant is $15].
Pattern VP+VP		It can help salesperson [VP][to [VP][promote up-sales and cross sales] , [VP][provide better services]].

Table 2: Parse tree patterns distinctive of comma splices, illustrated with examples.

tence is a comma splice, we need to ensure that the removed conjunction or relative pronoun was serving as the link between two clauses or verb phrases. For this purpose, we manually identified several parse patterns. In the parse tree, a conjunction and the elements that it joins together are always on the same level — the level of coordination (Bies et al., 1995). We looked to the right of the conjunctions in the trees and only removed those that were followed by either an "S" or a "VP". Relative clauses are adjoined to the head noun phrase, and both the relative pronoun and the clause are put inside the SBAR level. We removed only those relative pronouns that were followed by either an "S" or a "VP" and with an "SBAR" parent, which in turn had an "NP" parent. For example, the "and" was removed in the sentence "Mr. Katzenstein would have learned something, and it's possible Mr. Morita would have too.", and the relative pronoun "which" was removed in the sentence "Cray Research is transferring about $53 million in assets, primarily those related to the Cray-3 development, which has been a drain on Cray Research 's earnings.".

Another way to create a comma splice is to fuse two sentences together and replace the full-stop of the first sentence with a comma. However, comma splices introduced with this method do not reflect well the actual mistakes that English learners make, especially in terms of lexical features. For example, we observed that it is more common for comma splices to occur before pronouns than before proper nouns in the students' writing, but it would not be the case for the sentences created with this method. Therefore, we did not include comma splices introduced by fusing sentences together.

Out of 13159 instances of commas, this method yielded 2775 comma splices.

4.2 Test Sets

Although run-on sentences and comma splices were among the 28 error types introduced in the CoNLL-2014 shared task (Ng et al., 2014), the test set used in the task only contained about 26 such errors, and is therefore too small for our purpose. We evaluated our system on two test sets[8]: the learner corpus at City University of Hong Kong (Lee and Webster, 2012) (henceforth, the "CityU Set") and the EF-Cambridge Open Language Database (Geertzen et al., 2013) (henceforth, the "Cambridge Set").

CityU Set. The learner corpus at City University of Hong Kong consists of academic writing by university students, most of whom are native speakers of Chinese. Three of the error categories in this corpus are concerned with comma splices — "new sentence", "conjunction missing" and "missing relative pronoun". We randomly selected 550 sentences that are marked with one of these three categories: 215 with "new sentences", 215 with "conjunction missing", and 120 with "missing relative pronoun". Not every sentence marked with these categories is

[8]In another potential source of data, the NUCLE corpus (Dahlmeier et al., 2013), the annotation for comma splices is non-exhaustive, and would require additional human annotation to measure precision.

a comma splice since the error tags cover other error types as well, e.g., fused sentences, missing conjunctions for NPs, missing complementizer "that", etc. Human annotation (section 4.3) is therefore necessary to tell these apart. We also randomly selected 300 sentences from the corpus that are not marked with any of the three categories, with the sole constraint that their average length be similar to those of the marked sentences. Among the 1247 commas in these sentences, 235 were marked by at least one of the annotators as comma splices. Among sentences with comma splices, most contain only one; only about 10% contain two or more.

Cambridge Set. The Cambridge Set consists of writing submitted by language learners to an online school of EF Education First. The database has been partially error-annotated and the error category "New sentence" covers most comma splices. We used the writing by Chinese students, totaling 1.3 million words. Unfortunately, the annotation for run-on errors is not exhaustive, so human annotation was needed.

We selected a subset of 400 sentences marked with the "New sentence" error in the corpus and 400 unmarked sentences for annotation. This subset contains 2206 commas, of which 951 were marked as comma splices by one of the human annotators.

4.3 Human agreement

We asked two annotators, one a native speaker of English and the other a near-native speaker, to identify comma splices in 850 sentences drawn from the CityU Set. We first measured the agreement between the annotators on whether a sentence contained a comma splice, without regard to the location. The kappa was 0.90. Next, we investigated how often the annotators agreed on the location of the comma splice. Using one annotator as the gold standard, the precision is 91% and the recall is 92%. Most disagreements involve two consecutive commas enclosing a subordinate phrase, e.g., the phrase headed by "because" in the sentence "The most time consuming part is to purchase components, because most of the components were not sold in Hong Kong, it was need to purchase them in Mainland China". One annotator, attaching the "because" phrase to the preceding clause, identified the first comma as a comma splice; the other annotator, attaching the "because" phrase to the following clause, identified the second comma as a comma splice.

On the Cambridge Set, we measured the agreement between the annotator and original annotations in the corpus. Using the original annotations as the gold standard, the recall of the annotator is 0.91. Most disagreements involve the treatment of informal language. For example, the annotator considered it acceptable to use a comma in the sentence "I can cook dinner for you, please buy something for me." while the original annotation changed the comma to a full-stop.

4.4 Baselines

We evaluated two baseline systems in our experiments. First, we trained a CRF model on the Penn Treebank (section 4.1) with the baseline features. We computed a second baseline using the grammar checker in Microsoft Word 2013. We configured Microsoft Word's grammar checker to capture all error types and inspected each comma that the checker marked as a mistake, then compared the commas it flagged with our results. Two of Word's error types are relevant to our experiment: "Comma splice" and "Comma use". In the first case, the grammar checker would flag the comma as "Comma splice" and suggest that it be replaced with a semi-colon. In the second case, the grammar check would highlight the clauses before and after the comma, and suggest that an "and" should be added after it.

4.5 Results

We used CRF++ (Kudo, 2005) in our experiments. In our CRF model with the full feature set (Table 1), the parse features were extracted both from the sentences and from the output of the Stanford parser (Klein and Manning, 2003). Following (Israel et al., 2012), we used a filter that required the classifier to be at least 90% confident in a positive decision before flagging the comma as a comma splice. We adopted the evaluation metric used in the CoNLL-2014 shared task, $F_{0.5}$, which emphasize precision twice as much as recall because it is important to minimize false alarms for language learners[9].

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

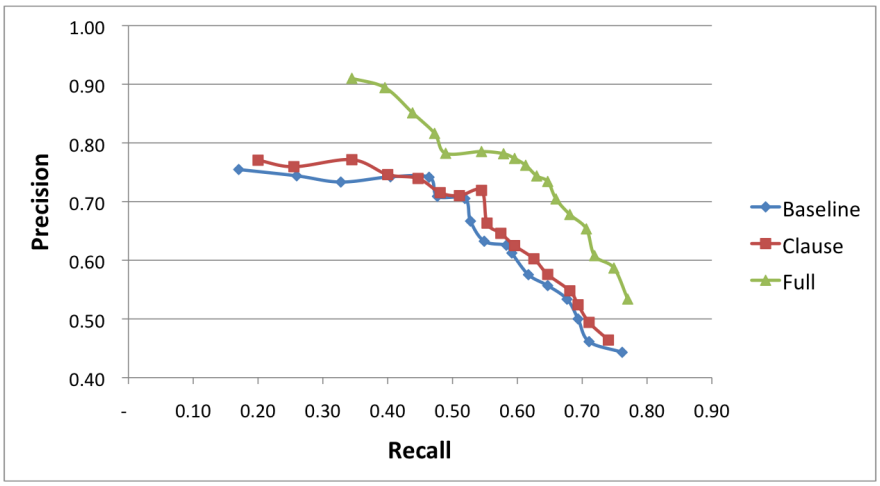


Figure 1: The precisions and recalls of the baseline, clause, and full system on the CityU Set when the probability threshold was decreased from 0.9 to 0.1 with a 0.05 interval.

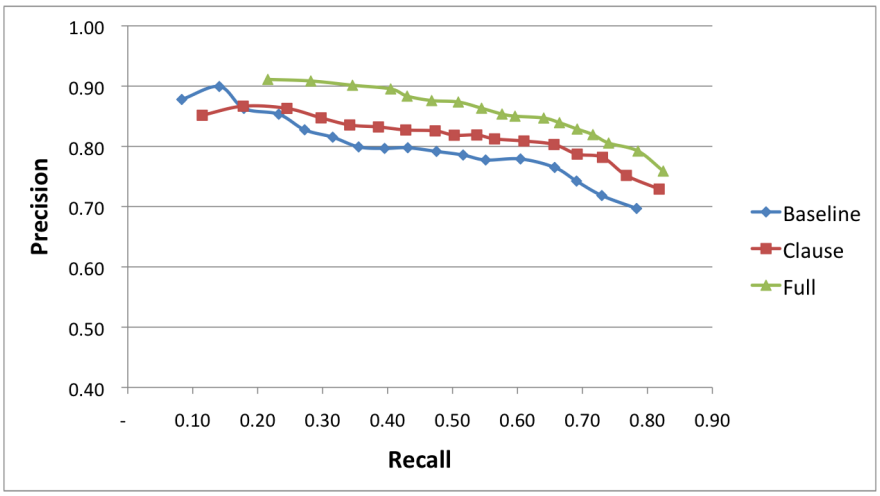


Figure 2: The precisions and recalls of the baseline, clause, and full system on the Cambridge Set when the probability threshold was decreased from 0.9 to 0.1 with a 0.05 interval.

On cross-validation of the training set, our baseline system achieved 0.82 precision, 0.29 recall and an F-measure of 0.60. The inclusion of clause features yielded 0.78 precision, 0.30 recall and an F-measure of 0.59 while the full system yielded 0.87 precision, 0.45 recall and an F-measure of 0.73.

The results for the test sets are shown in Table 3. On the CityU Set, our baseline system achieved 0.75 precision, 0.17 recall and an $F_{0.5}$ of 0.45; it performed better on the Cambridge Set, at 0.88 precision, 0.08 recall and an $F_{0.5}$ of 0.30. The Microsoft Word grammar checker achieved similar results as the baseline system on the CityU set, but outperformed it on the Cambridge Set, at 0.90 precision, 0.13 recall and $F_{0.5}$ of 0.41.

The clause features improved upon the baseline system in recall on both sets, at 0.20 for the CityU Set and 0.11 for the Cambridge Set. In terms of precision, they improved performance on the CityU set (0.77), but were unhelpful for the Cambridge set (0.85).

The full system improved upon the two baselines and the clause system in both precision and recall, performing at 0.91 precision, 0.34 recall and an $F_{0.5}$ of 0.69 for the CityU set; and slightly lower, at 0.91 precision, 0.22 recall and an $F_{0.5}$ of 0.55, for the Cambridge set. All these improvements are statis-

tically significant[10].

On both test sets, many of the errors were due to sentences with non-standard vocabulary and real-word spelling errors. such as misspelling "maybe" as "may be", or "besides" as "beside". Both phenomena can yield an unexpected parse tree, causing a missed parse pattern.

For the CityU set, performance was hurt by the structurally more complicated sentences. The system failed to flag comma splices that involve three or more clauses, i.e., "S_1, S_2, S_3", where both "S_1, S_2" and "S_2, S_3" would form perfectly correct sentences (e.g., "The most time consuming part is to purchase components, because most of the components were not sold in Hong Kong, it was need to purchase them in Mainland China").

Performance on the Cambridge Set was helped by shorter and structurally simpler sentences, which resulted in more accurate parsing, but was hurt by the presence of many consecutive comma splices (e.g., "He is student, he is always wearing school uniform, my name is Songlin.") and unconventional use of conjunctions such as beginning a sentence with "but", which are rare in the training data. The Cambridge Set also contained plenty of informal sentences, for which the rules concerning the use of commas are less rigid. For example, while the system marked the sentence "Hi granny, my name is Winky." as a comma splice, the annotators did not because using a comma in this situation is commonly acceptable.

→ Corpus	CityU Set	Cambridge Set
↓ System	P/R/$F_{0.5}$	P/R/$F_{0.5}$
Full	0.91/0.34/0.69	0.91/0.22/0.55
Clause	0.77/0.20/0.49	0.85/0.11/0.37
Baseline	0.75/0.17/0.45	0.88/0.08/0.30
MS Word	0.74/0.15/0.41	0.90/0.13/0.41

Table 3: Precision, recall and F-measure for comma splice detection. "Baseline" refers to the CRF model trained only on the baseline features (Table 1). "Clause" refers to the CRF model that uses both baseline features and clause features. "Full" uses the full feature set. "MS Word" refers to the grammar checker embedded in Microsoft Word 2013.

[10]At p <= 0.05 by McNemar's test.

4.6 Precision-Recall Trade-off

The precision-recall balance can be adjusted based on the probability threshold above which a comma is flagged as a comma splice. Figures 1 and 2 show the degree to which precision can be traded off for recall by using different thresholds. For example, when a threshold of 0.65 was used, the precision of the full system on the CityU set dropped to 0.79 while recall rose to above 0.5.

On both test sets, the precision and recall of the full system are consistently higher than the baseline and clause systems. The drop in precision for the CityU set is steeper than that of the Cambridge set. This may be because the sentences in the CityU set are generally more complicated than those in the Cambridge set. In order for the system to perform with a high precision, a greater degree of recall has to be sacrificed.

5 Conclusion

We have introduced a new task — detection of comma splices, a common mistake made by non-native speakers in English writing — and have shown a high level of agreement among human annotators.

We have also applied a CRF model to comma splice detection. Our best system uses parse tree-based features and achieved an average of 0.91 precision and 0.28 recall. It significantly outperformed a baseline system that does not consider syntactic features, and a widely used commercial grammar checker.

In future work, we aim to further raise detection accuracy by improving parser robustness, and to tackle the task of suggesting repairs for comma splices.

Acknowledgments

The work described in this paper was supported by a Strategic Research Grant (Project No. 7008166) from City University of Hong Kong.

References

Timothy Baldwin and Manuel Paul Anil Kumar Joseph. 2009. Restoring Punctuation and Casing in English

Text. *Proc. Australasian Conference on Artificial Intelligence*.

Pairote Bennui. 2008. A study of L1 interference in the writing of Thai EFL students. *Malaysian Journal of ELT Research* 4:72–102.

Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. 1995. Bracketing guidelines for Treebank II style Penn Treebank project. *Technical Report*, University of Pennsylvania.

Jill Burstein, Martin Chodorow, and Claudia Leacock. 2004. Automated Essay Evaluation: the Criterion Online Writing Service. *AI Magazine*.

Robert Connors and Andrea 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, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English. *Proc. 8th Workshop on Innovative Use of NLP for Building Educational Applications*.

Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 pilot shared task. *Proc. 13th European Workshop on Natural Language Generation*, p.242–249.

Robert Dale, Ilya Anisimoff, and George Narroway. 2012. HOO 2012: A report on the preposition and determiner error correction shared task. *Proc. 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 54–62.

Steven Donahue. 2001. Formal errors: Mainstream and ESL students. Presented at the *2001 Conference of Two-Year College Association (TYCA)*.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. *Proc. Fifth International Natural Language Generation Conference* pp. 25–32.

Jennifer Foster, Joachim Wagner, and Josef van Genabith. 2008. Adapting a WSJ-trained parser to grammatically noisy text. *Proc. ACL*.

Jennifer Foster and Oistein E. Andersen. 2009. GenERRate: generating errors for use in grammatical error detection. *Proc. Fourth Workshop on Innovative Use of NLP for Building Educational Applications*.

Jeroen Geertzen, Theodora Alexopoulou, and Anna Korhonen. 2013. Automatic linguistic annotation of large scale L2 databases: The EF-Cambridge Open Language Database (EFCAMDAT). *Proc. 31st Second Language Research Forum (SLRF)*.

Agustin Gravano, Martin Jansche, and Martin Bachiani. 2009. Restoring Punctuation and Capitalization in Transcribed Speech. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

D. Hacker and N. Sommers. 2011. *Rules for writers*. Macmillan.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting Errors in English Article Usage by Non-Native Speakers. *Natural Language Engineering*, 12(2).

Jing Huang and Geoffrey Zweig. 2002. Maximum entropy model for punctuation annotation from speech. *Proc. ICSLP* p. 917–920.

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. *Proc. ACL*.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *Proc. National Conference on Artificial Intelligence* pp. 703–710.

Ross Israel, Joel Tetreault and Martin Chodorow. 2012. Correcting Comma Errors in Learner Essays, and Restoring Commas in Newswire Text. *Proc. NAACL*.

Taku Kudo. 2005. CRF++: Yet another CRF toolkit. Obtained from http://crfpp.sourceforge.net.

John Lee and Stephanie Seneff. 2008. Correcting Misuse of Verb Forms. *Proc. ACL*.

John Lee and Jonathan Webster. 2012. A Corpus of Textual Revisions in Second Language Writing. *Proc. ACL*.

F. Y. Lin. 2002. Preferred structures in Chinese-English translation. Master's thesis, National Changhua University of Education, Taiwan.

Andrea A. Lunsford and Karen J. Lunsford. 2008. Mistakes are a Fact of Life: A National Comparative Study. *College Composition and Communication* 59(4):781–806.

Daniel Marcu and Abdessamad Echihabi. 2002. An Unsupervised Approach to Recognizing Discourse Relations. *Proc. ACL*.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM* 38(11):39–41.

Mitchell Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2).

Hwee Tou Ng, Siew Mei Wu, Wu, Y., Hadiwinoto, C., and Tetreault, J. 2013. The CoNLL-2013 shared task on grammatical error correction. *Proc. CoNLL: Shared Task*.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 Shared Task on Grammatical Error Correction. *Proc. CoNLL: Shared Task*.

Mohammad Rahimi. 2009. The role of teacher's corrective feedback in improving Iranian EFL learners'

writing accuracy over time: is learner's mother tongue relevant? *Reading and Writing*, 22(2):219–243.

Andreas Stolcke and Elizabeth Shriberg. 1996. Automatic linguistic segmentation of conversational speech. *Proc. Fourth International Conference on Spoken Language Processing (ICSLP)*.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. *Proc. ACL*.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using Parse Features for Preposition Selection and Error Detection. *Proc. ACL*.

Yen-Chu Tseng and Hsien-Chin Liou. 2006. The effects of online conjunction materials on college EFL students' writing. *System*, 34(2):270–283.