Compare Results

Old File:		New File:
2024.emnlp-main.942.pdf	versus	**2024_emnlp-main_942.pdf**
9 pages (277 KB)		**10 pages (148 KB)**
10/31/2024 10:33:04 PM		2/9/2026 2:13:41 PM

Total Changes

991

Content

153 Replacements
168 Insertions
226 Deletions

Styling and Annotations

260 Styling
184 Annotations

Go to First Change (page 2)

LLM-based Code-Switched Text Generation for Grammatical Error Correction

Tom Potter*
University of Manchester
thomas.potter@postgrad.manchester.ac.uk

Zheng Yuan
King's College London
zheng.yuan@kcl.ac.uk

Abstract

With the rise of globalisation, code-switching (code-switching) has become a ubiquitous part of multilingual conversation, posing new challenges for natural language processing (NLP), especially in Grammatical Error Correction (grammatical error correction). This work explores the complexities of applying grammatical error correctionsystems to code-switchingtexts. Our objectives include evaluating the performance of state-of-the-art grammatical error correctionsystems on an authentic code-switchingdataset from English as a Second Language (English as a Second Language) learners, exploring synthetic data generation as a solution to data scarcity, and developing a model capable of correcting grammatical errors in monolingual and code-switchingtexts. We generated synthetic code-switchinggrammatical error correctiondata, resulting in one of the first substantial datasets for this task, and showed that a model trained on this data is capable of significant improvements over existing systems. This work targets English as a Second Languagelearners, aiming to provide educational technologies that aid in the development of their English grammatical correctness without constraining their natural multilingualism.

1 Introduction

Code-switching (code-switching), the practice of fluidly alternating between two or more languages in conversation, has become commonplace in recent years. This linguistic phenomenon, emerging as a natural consequence of multilingualism, is now widely accepted in social and professional settings[??]. Many works have highlighted the utility and cultural importance of code-switchingin general conversation [??]. Further research indicates that these advantages extend to language learning, with code-switchingoffering many pedagogical benefits. These include increasing students' access to content and improving their confidence. nguyen2022building discuss the mechanisms for this, where students use a familiar language to grasp foreign, complex concepts. code-switchingcan also serve as a scaffolding tool, helping to bridge gaps in a student's comprehension of a language and enabling them to build upon existing knowledge. These benefits reduce the barriers between a student and their target language and help promote a learning environment conducive with active exploration and deeper understanding. Therefore, it is essential that English as a Second Language (English as a Second Language) learners are not penalised for expressing their cultural identity through code-switching.

Example 1: According to the test, [lacks in me → my shortcomings] are and .

Example 2: When we [call → say] do we actually mean a glove compartment in English?

Figure 1: Examples of grammatical error correctionin English as a Second Languagelearner language.

Grammatical error correction (grammatical error correction) is the task of automatically detecting and correcting errors in text. Research on grammatical error correctionfor code-switchingtext remained largely unexplored. chan2024grammatic[??]were the first to demonstrate that exposing a sequence-tagging grammatical error correctionmodel to code-switchingtext during the training process improves performance compared to a monolingual system. However, further work is essential to ensure language technology is inclusive and reflective of real-world linguistic practices. Figu[??]?? shows two examples of code-switchingfrom our target population with their grammatical corrections.

Despite significant advancements in grammatical error correctionin recent years, a gap persists in addressing code-switchingtexts, with monolingual grammatical error correctiondatasets labelling code-switchingas a type of error [??]. There are several reasons for this, the most prominent being the scarcity of high-quality training data, a problem that plagues monolingual grammatical error correctionsystems.

[1]The definition of code-switchingis a subject of ongoing debate. Throughout this work, we use the term code-switchingto refer specifically to the type of language mixing exhibited by English as a Second Languagelearners.

The unique linguistic features of code-switching, including its variable syntax, semantics and pragmatics, add additional complexity to this task. Monolingual seq2seq grammatical error correctionmodels, e.g. T5[?], struggle with code-switchingtext as they fail to represent the non-English inputs, resulting in their inability to output the code-switchingtext. On the other hand, multilingual seq2seq models and edit-based grammatical error correctionmodels like GECToR [?] can handle code-switchingtext but struggle with the ambiguity present at language switching points. This ambiguity challenges the models' ability to accurately correct the text.

This paper aims to bridge this gap. Firstly, to address the data scarcity issue, we propose a method for generating high-quality synthetic code-switchinggrammatical error correctiondata, using which we produce, to our knowledge, one of the first substantial datasets labelled for this task.[2] Secondly, we train a token classification-style grammatical error correctionsystem, tailored to correct errors in texts produced by English as a Second Languagelearners. This demographic is significant for our study as they not only present consistent code-switchingpatterns but also stand to benefit greatly from a grammatical error correctionsystem capable of handling code-switchingtext.

2 Data

2.1 Genuine CSW GEC Dataset

One of the only datasets labelled for grammatical error correctionwhich does not remove code-switchingtext, is the Lang-8 dataset [?], sourced from the Lang-8 language learning platform. This dataset, when filtered to contain entries where code-switchingis present, offers a foundation of authentic data, comprises 5,875 pairs of ungrammatical and corrected sentences across 6 code-switchinglanguage pairs: English-Japanese (81.9%), English-Korean (13.0%), English-Traditional Chinese (3.4%), English-Russian (1.2%), English-Thai (0.5%) and English-Arabic (0.1%).

The crowd-sourced nature of Lang-8 required manual validation to ensure accuracy. We tasked an annotator with the responsibility of verifying the original corrections in the dataset, as well as combing for missed errors, incorrect annotations and over-annotations.

2.2 Synthetic CSW GEC Data Generation

Given the small size of the available code-switchinggrammatical error correctiondataset, we introduced a 2-step approach to synthetic code-switchinggrammatical error correctiondata generation. First, we generated grammatically correct code-switchingsentences. This is followed by the introduction of errors.

[2]This dataset is available on GitHub.

2.2.1 Step 1: CSW Text Generation

Three different synthetic data generation techniques have been explored to generate code-switchingdata.

Translation-based CSW Text Generation required a monolingual corpus, a machine translation (MT) algorithm, and a sentence parser. To generate a code-switchingutterance, we used the Stanford Parser v4.5.4 [?] to build a syntactic parse tree. We then randomly selected and translated a subtree using the ArgosTranslate MT package [?]. This method generates plausible code-switchingtext. However, performance is dependent on the strength of the parsing and translation algorithms; and the style of language within the corpus. To approximate the style of our authentic code-switchingtext, we used corrected monolingual sentences from the Lang-8 corpus.

Parallel Corpus-based CSW Text Generation avoids the need for a translation algorithm. Instead, we used the same Stanford Parser, this time with Spanish, French and German configurations; and the AWESOME word-alignment model [?], to identify parts of parallel corpora labelled for MT with similar syntactic structure. For this method, we use the Europarl corpus [?] due to the grammatical quality of its English component. Under the Equivalence Constraint Theory [?], these areas are where code-switchingis likely to take place. We, therefore, randomly chose overlapping subtrees as candidates for injection of non-English text. Although this method does not require MT, it is reliant on performant word-alignment and parsing systems; these are rare for many languages.

LLM Prompting-based CSW Text Generation The other methods of generating code-switchingtext rely on injecting a second language into existing monolingual corpora. Hence, they are not able to recreate one of the main switching styles shown by English as a Second Languagelearners—code-switchingas a genuine pragmatic strategy. A common reason for this style of switching is when quoting another language. It is difficult to recreate this style using a monolingual foundation as sentences like *The Japanese word for "dog" is "* seldom appear in authentic monolingual corpora.

To generate diverse code-switchingtexts without relying on existing corpora or inaccurate alignment algorithms, we leveraged the strong general knowledge of Large Language Models (LLMs). We demonstrated that OpenAI's GPT-3.5 [?] can create high-quality code-switchingsentences when shown examples of authentic utterances. Along with genuine code-switchingtexts, we supplied a one-shot example of how to use the switching styles of an existing code-switchingtext to generate a new sentence.[3]

Comparison of Synthetic CSW Text We used several code-switchingmetrics to quantify the qualities of code-switchingtexts: Code Mixing Index (CMI) [?], Multilingual Index (M-Index) [?], Probability of Switching (I-Index) [?].

[3]The full prompt can be seen in Appendix A.

Table 1: Quantitative Description of the Genuine and Generated CSW Datasets Using Various CSW Metrics.

Metric	Genuine CSW	LLM CSW	Translation CSW
CMI	15.52	16.14	27.81
M-Index	0.007	0.004	0.015
I-Index	0.21	0.21	0.30
Burstiness	-0.07	-0.04	0.03
CF1	6.38	5.82	17.13
CF2	19.77	19.03	31.11
CF3	18.34	17.61	30.05

Burstiness [**?**], and Complexity Factor (CF1-3) [**?**]. Table **??** shows the value of each metric for our genuine code-switchingdataset, as well as for these 3 synthetic code-switchingdatasets. We can see that the LLM prompting-based dataset was superior in its similarity to the authentic code-switchingdata. Using this method, we generated a corpus of 73,293 utterances covering over 20 English language pairs, including the 6 language pairs in the original dataset.[4]

2.2.2 Step 2: Synthetic Error Generation

Several works have shown the effectiveness of rule-based error injection for grammatical error correctiondata generation. Many use the PIE-synthetic dataset [**?**], a perturbed version of the 1BW corpus [**?**]. For each sentence, the authors introduce between 0 and 4 errors of random type. We extended this work by introducing a new subset of error types that are not only more common in English as a Second Languagelearners, but also are areas where the state-of-the-artperformance collapses when faced with code-switchingtext: noun, pronoun, word order, determiner, and punctuation errors.[5]

To increase the diversity of errors, we adopted a second style of error injection, Backtranslation [**?**]. By swapping the source and target sentences of a monolingual dataset, we trained a GECToR-based system to induce errors in our synthetic code-switchingsentences.

Using these methods, we created two datasets: Syn-CSW PIE and Syn-CSW Rev-GECToR. After removing pairs containing no injected errors, we are left with 70,180 and 18,159 sentences each.

3 CSW GEC Systems

For our grammatical error correctionsystem targeting code-switchingtexts, we chose a GECToR model [**?**], with a RoBERTa-base foundation, due to its proven efficacy with limited training data and stronger performance on code-switchingtexts compared to seq2seq models. We added a new code-switchingdetect to the error detection head, adding the ability to detect code-switchingtokens.

Following Yarnavskyi2022ensembling, we used a 3-stage training schedule. In the first, we used the same distilled 1BW corpus, and added all our synthetic code-switchinggrammatical error correctiondata. For the second, we used several grammatical error correctiondatasets: NUCLE [**?**], FCE [**?**], W&I Locness [**?**], Lang-8 and our 2 synthetic code-switchingdatasets. As our genuine code-switchingdataset is a subset of the Lang-8 corpus, we checked and removed any duplicates. Following previous works, we finished training using the W&I Locness dataset due to its superior quality. In this final stage, we added a sampled subset of our synthetic code-switchingsentences and 90% of our authentic code-switchingdata, ensuring exposure to synthetic and genuine code-switchingtext. At each stage, we reserved 5% of the data for validation.[6] Finally, we tuned inference parameters using a grid-search to optimise the $F_{0.5}$ on the final validation set. By beginning with pre-training on large amounts of lower-quality data in the early stages, this multi-stage learning process allows the model to first build a robust grammatical error correctionfoundation before refining it with high quality data in the latter stages. This approach allows the model to learn incrementally, reducing the risk of the model being overwhelmed by the complexity of the task from the outset.

4 Results and Analysis

4.1 Baseline Comparisons

We compared our model against two well-established systems: a RoBERTa-base GECToR model [**?**], with near state-of-the-artperformance on the BEA-2019 test set [**?**], and a seq2seq T5 model [**?**]. To assess these models, we evaluated their performance on the BEA-2019 test set and the remaining 10% of our authentic code-switchingdata. The ERRANT [**?**] grammatical error correctionevaluation results, as outlined in Table **??**, demonstrate a clear degradation in performance when these two systems are applied to code-switchingtexts. The ERRANT toolkit detects and classifies edits between source and target sentence pairs into predefined error categories. It enables the comparison of a proposed set of edits with a reference set, providing a way of calculating metrics, such as precision and recall, across these categories.

[4] The LLM does not always generate the language pairs we ask for. However, these sentences are still included in the dataset categorised under their actual language pair.

[5] Error type analysis is presented in Appendix B.

[6] An exact breakdown of contributions by each dataset is given in Appendix C.

Table 2: ERRANT-based Precision, Recall and $F_{0.5}$ Scores of Baselines and Our Model Throughout Training

Model	BEA-2019 Test			Genuine CSW		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
Existing GEC systems						
GECToR	77.88	53.07	71.22	71.14	27.08	53.67
T5-Small	62.03	47.19	58.34	11.70	24.98	13.09
Our CSW GEC systems						
Stage 1	67.23	53.88	64.05	66.15	26.04	50.57
Stage 2	72.64	51.73	67.20	65.41	29.93	52.87
Stage 3	74.32	53.40	68.92	84.66	22.92	55.02
Inference Tweaks	69.01	58.40	66.59	76.02	38.67	63.71

4.2 Detailed Model Performance

The progression of our model throughout training provided insights into its evolving capabilities and effectiveness of our synthetic data. We monitored several metrics, including the ERRANT precision, recall and $F_{0.5}$ score, for the BEA-2019 test set and the remaining unused 10% of our genuine code-switchingdataset. These metrics, as displayed in Table **??**, indicate a steady improvement in the ability to handle code-switchingtexts. Notably, the performance on the code-switchingdataset shows a significant leap in the final stages, where the contribution of our synthetic dataset is largest. This improvement in code-switchingtext handling did slightly compromise the model's performance on monolingual grammatical error correctiontasks, as seen on the BEA-2019 test set. This suggests a trade-off inherent in specialising the model for code-switchingcontexts. However, our model remains competitive amongst state-of-the-artmonolingual grammatical error correctionsystems of its size.

Three illustrative examples of our model's corrections, taken from the code-switchingtest set, can be seen in Figure **??**. The first example demonstrates a case where the model has correctly identified all of the changes required, including the incorrect capitalisation of a word, a missing word, and some missing punctuation. The second example shows a "near miss"; here, the model has correctly identified the majority of the changes required but dropped the "I" whilst rearranging the start of the sentence. Finally, the third example presents a scenario where the model has fallen slightly short, failing to recognise the need for "were" instead of "was" in this hypothetical context.

4.3 Inference Tweaking and Error Thresholds

The inference tweaking phase was crucial in tuning the balance between precision and recall. The changes made here, particularly lowering the minimum error thresholds before the model makes an edit, indicated a clear attempt to force the model to make more corrections. While this slightly lowered precision on monolingual errors, it significantly enhanced the performance on code-switchingtext.

To determine that the improved performance of our proposed model was not entirely due to the different inference configuration, we conducted a similar grid search for the existing GECToR model. However, instead of using the Stage 3 validation dataset, as we did with our model, we used the code-switchingtest set directly. The highest $F_{0.5}$ achieved by the baseline model was 56.46, providing evidence that our proposed model beats all inference configurations of the previous GECToR system when applied to code-switchingtexts.

Gold Correction 1: We have many [New → new] words for [∅ → the] unemployed [∅ → :] "" [∅ → ,] "" [∅ → ,] "".
Proposed Correction 1: We have many [New → new] words for [∅ → the] unemployed [∅ → :] "" [∅ → ,] "" [∅ → ,] "".
Gold Correction 2: [I and my girlfriend → My girlfriend and I] looked [∅ → at a] picture called "" (Immaculate Conception).
Proposed Correction 2: [I and my girlfriend → My girlfriend and] looked [∅ → at a] picture called "" (Immaculate Conception).
Gold Correction 3: If he [was a → were] Japanese, I suppose I [replied → would reply] like this: "800m".
Proposed Correction 3: If he was [a → ∅] Japanese, I suppose I [replied → would reply] like this: "800m".

Figure 2: Three examples of model's proposed corrections from the CSW test set.

4.4 Synthetic Data Impact

The synthetic code-switchingtext and error injection methods were central to this project. The resemblance of our synthetic text to real English as a Second Languagelearner data, as shown by the similarity metrics in Table **??**, is a testament to the effectiveness of our chosen generation method. The improvements in $F_{0.5}$ scores provide further evidence of this.

Our extended PIE-synthetic dataset aimed to introduce four error types common in English as a Second Languagestudents: noun, pronoun, punctuation and word errors. When compared to the monolingual GECToR, our model is stronger in all of these areas.[8] This provides strong evidence that the targeted approach to error injection was successful in boosting the model's ability in these areas.

[7] Implementation details are presented in Appendix D.
[8] Error type analysis of our model is given in Appendix E.

5 Conclusion

The primary aim of this paper was to build a grammatical error correctionsystem capable of effectively correcting English errors in code-switchingtext, whilst maintaining competitive performance on monolingual data. To address the scarcity of code-switchingdata, we explored methods of generating synthetic code-switchingtext. We used several code-switchingmetrics to establish that the LLM prompting-based approach was the most capable of generating text resembling the content in our genuine dataset. From there, we used two error injection methods to create the first substantial datasets labelled for code-switchinggrammatical error correction. This significantly expanded the training data available. Importantly, it also opened up opportunities for future research in code-switchinggrammatical error correctionand code-switchingNLP more generally. We demonstrated the efficacy of our synthetic data generation techniques by training the first grammatical error correctionmodel aimed at correcting errors in code-switchingtexts. Our model showed a clear improvement in performance on code-switchingdata, surpassing the state-of-the-artin this area.

6 Limitations

This research, while comprehensive, encounters several limitations that highlight areas for potential improvement. One primary limitation lies in the overrepresentation of Japanese in the genuine code-switchingdataset. This raises questions about the model's applicability to a broader range of language pairs. This is an unfortunate consequence of using a dataset sourced from Lang-8, a Japanese language learning network. Although our method demonstrated that it could generate texts from a wider range of language pairs, it is possible that all code-switchingdata used shows a bias towards Japanese styles of code-switching. Such a bias in our system could inadvertently lead to reduced accessibility and effectiveness for English as a Second Languagelearners who code-switchingwith languages other than Japanese. If this system were to be used as an aid in English as a Second Languageeducation, steps should be taken to ensure that it does not contribute to existing inequalities present in language learning platforms. English learning tools should be accessible regardless of the student's native language and future work should focus on developing more inclusive datasets to help mitigate these risks.

Another possible limitation relates to the style of model chosen. The sequence tagging method was selected due to its lower data requirements, but this decision may have constrained the capabilities of the model. Many of the errors typical of English as a Second Languagestudents require complex restructuring, which is a notably difficult task for edit-based grammatical error correctionsystems. Although the data needs are more substantial, it is likely that NMT grammatical error correctionsystems may fare better as they are not constrained by a limited vocabulary of edits.

To assess the likeness of our generated code-switchingtext, we introduced several common code-switchingmetrics. Although useful, these metrics are not very sophisticated, and often struggle to accurately capture the nuances of code-switchingpatterns across different sub-populations. These language patterns can have a substantial impact on the optimal approaches to problems across code-switchingNLP, and hence, the field would benefit from further research in this area.

Ideally, we would have conducted a human study to evaluate the quality of our synthetic data. However, given the constraints of the project, it was not possible, and we acknowledge this as a limitation of our work.

Finally, we reported results for a RoBERTa-base GECToR system. Although we also tested other base models, including BERT, DeBERTa and ELECTRA, we did not look at larger models or ensemble systems. Future extensions could explore this area, building upon the observation that larger models or simple voting ensembles can yield better results than the smaller base models [?].

In summary, whilst the current work makes significant contributions to the field of grammatical error correctionfor code-switchingtext, these limitations indicate crucial areas for further research and development.

References

References

[Awasthi et al.2019] Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4259–4269, Hong Kong, China. Association for Computational Linguistics.

[Barnett et al.2000] Ruthanna Barnett, Eva Codó, Eva Eppler, Montse Forcadell, Penelope Gardner-Chloros, Roeland van Hout, Melissa Moyer, Maria Carme Torras, Maria Teresa Turell, Mark Sebba, Marianne Starren, and Sietse Wensing. 2000. The lides coding manual: A document for preparing and analyzing language interaction data version 1.1—july, 1999. *International Journal of Bilingualism*, 4(2):131–132.

[Beatty-Martínez et al.2020] Anne L Beatty-Martínez, Christian A Navarro-Torres, and Paola E Dussias. 2020.

Codeswitching: A bilingual toolkit for opportunistic speech planning. *Frontiers in Psychology*, 11:1699.

[Brown et al.2020] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

[Bryant et al.2019] Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

[Bryant et al.2017] Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

[Chan et al.2024] Kelvin Wey Han Chan, Christopher Bryant, Li Nguyen, Andrew Caines, and Zheng Yuan. 2024. Grammatical error correction for code-switched sentences by learners of English. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7926–7938, Torino, Italia. ELRA and ICCL.

[Chelba et al.2013] Ciprian Chelba, Tomás Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005.

[Dahlmeier et al.2023] Daniel Dahlmeier, Hwee Tou Ng, Siew Mei Wu, et al. 2023. Nus corpus of learner english (nucle). National University of Singapore, NLP Group. Available: `https://www.comp.nus.edu.sg/~nlp/corpora.html`.

[Dou and Neubig2021] Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online. Association for Computational Linguistics.

[Falbo and LaCroix2021] Arianna Falbo and Travis LaCroix. 2021. Est-ce que vous compute? code-switching, cultural identity, and AI. *CoRR*, abs/2112.08256.

[Finlay2023] P.J. Finlay. 2023. Argos translate. Open-source offline translation library written in Python.

[Gambäck and Das2016] Björn Gambäck and Amitava Das. 2016. Comparing the level of code-switching in corpora. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1850–1855, Portorož, Slovenia. European Language Resources Association (ELRA).

[Ghosh et al.2017] Souvick Ghosh, Satanu Ghosh, and Dipankar Das. 2017. Complexity metric for code-mixed social media text. *Computación y Sistemas*, 21(4):693–701.

[Goh and Barabási2008] K.-I. Goh and A.-L. Barabási. 2008. Burstiness and memory in complex systems. *Europhysics Letters*, 81(4):48002.

[Guzmán et al.2017] Gualberto Guzmán, Joseph Ricard, Jacqueline Serigos, Barbara E. Bullock, and Almeida Jacqueline Toribio. 2017. Metrics for Modeling Code-Switching Across Corpora. In *Proc. Interspeech 2017*, pages 67–71.

[Klein et al.2017] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017), System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

[Koehn et al.2012] Philipp Koehn et al. 2012. European parliament proceedings parallel corpus.

[Manning et al.2014] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

[Mizumoto et al.2013] Tomoya Mizumoto, Toshikazu Tajiri, Takuya Fujino, Seiji Kasahara, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2013. NAIST Lang-8 Learner Corpora. Language Learner Corpora

compiled from Lang-8 SNS. Available for research and educational purposes.

[Nguyen et al.2022] Li Nguyen, Zheng Yuan, and Graham Seed. 2022. Building educational technologies for code-switching: Current practices, difficulties and future directions. *Languages*, 7:220.

[Omelianchuk et al.2020] Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA→Online. Association for Computational Linguistics.

[Rizvi et al.2021] Mohd Sanad Zaki Rizvi, Anirudh Srinivasan, Tanuja Ganu, Monojit Choudhury, and Sunayana Sitaram. 2021. GCM: A toolkit for generating synthetic code-mixed text. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 205–211, Online. Association for Computational Linguistics.

[Rothe et al.2021] Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

[Stahlberg and Kumar2021] Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47, Online. Association for Computational Linguistics.

[Tarnavskyi et al.2022] Maksym Tarnavskyi, Artem Chernodub, and Kostiantyn Omelianchuk. 2022. Ensembling and knowledge distilling of large sequence taggers for grammatical error correction. In *Accepted for publication at 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, Dublin, Ireland.

[Yannakoudakis et al.2011] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

[Yow et al.2018] W. Quin Yow, Jessica S. H. Tan, and Suzanne Flynn. 2018. Code-switching as a marker of linguistic competence in bilingual children. *Bilingualism: Language and Cognition*, 21(5):1075–1090.

A Example LLM Prompt

Figure **??** shows an example LLM prompt used to generate synthetic code-switchingsentences from genuine examples. As we are using a private subset of the Lang-8 dataset, we are not permitted to share any of the code-switchingtexts.

Settings: [no prose]
For each of the following code-switched sentences, generate a new sentence that uses the same two languages and a similar style of code-switching. The topic should be different. Ensure you use the correct grammar in the English portion of the sentence. Make sure that each sentence contains 2 languages. Only return the sentences and their number. You must follow all of the instructions.
For example, given the source sentence and label: 1. This food is called "".
An acceptable answer would be: 1. This animal is called a "".
Do not include any other information in the generated sentences. The 10 real examples are as follows:
1. [CSW SENTENCE] 2. [CSW SENTENCE] 3. [CSW SENTENCE] 4. [CSW SENTENCE] 5. [CSW SENTENCE] 6. [CSW SENTENCE] 7. [CSW SENTENCE] 8. [CSW SENTENCE] 9. [CSW SENTENCE] 10. [CSW SENTENCE]

Figure 3: An Example LLM Prompt Used to Generate CSW Text

B Error Type Analysis of SOTA

Table shows a breakdown of the performance of a single RoBERTa Large-based GECToR system trained purely on monolingual grammatical error correctiondata when applied to two datasets, our genuine code-switchingdataset and the BEA-2019 test set. These datasets are approximately the same size. The model used represents a current near-state-of-the-artsingle model sequence tagging-based grammatical error correctionsystem measured using $F_{0.5}$ on the BEA-2019 test set. For brevity, we have removed categories with a low number of examples in either dataset or where performance is not significantly different.

Table 3: $F_{0.5}$ Scores, TP, FP, FN, and Differences in $F_{0.5}$ Scores (BEA - CSW) for Different Categories in the BEA-2019 Test Split and our Genuine CSW Dataset.

Category	BEA-2019 Test				Genuine CSW			
	$F_{0.5}$	TP	FP	FN	$F_{0.5}$	TP	FP	FN
DET	80.45	432	80	205	46.27	472	351	1336
NOUN	47.85	29	16	94	4.34	21	147	1725
ORTH	75.96	201	30	198	36.45	181	264	522
OTHER	39.51	113	77	557	3.55	39	241	4373
PREP	75.44	263	58	196	39.14	241	251	870
PRON	66.38	62	19	81	20.71	21	32	274
PUNCT	80.93	786	165	266	0.35	1	286	284
VERB	52.59	61	27	167	18.08	49	77	802
VERB:FORM	81.62	151	30	50	37.20	61	90	155
VERB:SVA	88.64	128	14	26	57.58	114	79	104
VERB:TENSE	65.55	145	62	133	33.80	116	172	448
WO	58.08	23	5	63	6.33	2	11	104

Table 4: Sentence Count and Contribution of Stage 2 Datasets

Dataset	Sentences
Lang-8	985,683 (80.54%)
W&I Locness	68,608 (5.61%)
NUCLE	54,258 (4.43%)
FCE	26,929 (2.20%)
Syn-CSW PIE	70,181 (5.73%)
Syn-CSW Rev-GECToR	18,160 (1.48%)
Total	1,223,819

of the genuine code-switchingdataset was retained for testing purposes. Table **??** details the contributions to this stage from each dataset.

Table 5: Sentence Count and Contribution of Stage 3 Datasets

Dataset	Sentences
W&I Locness	68,608 (67.23%)
Syn-CSW Rev-GECToR	18,160 (17.80%)
Syn-CSW PIE	10,000 (9.80%)
CSW Genuine	5,279 (5.17%)
Total	102,047

C Training Data Schedule

In this section, we explicitly detail the data used at each stage of the training process.

Stage 1 For the initial pre-training stage, we used the distilled dataset proposed by the state-of-the-art[**?**]. This dataset was constructed by extracting corrections from the monolingual 1BW corpus [] using the highest performing GECToR ensemble. Through this dataset, we shuffled our PIE-synthetic code-switchingdataset. We deemed this dataset to be of lower quality than its Rev-GECToR counterpart. Consequently, it was used earlier in the training process. This provided roughly 1,200,000 examples for the initial training phase of which we split between train and validation sets according to a ratio of 19:1. Our synthetic code-switchingsentences comprised approximately 5.65% of this dataset. We aimed to keep this percentage small in this phase of the training process to allow the model to first learn to correct errors in monolingual texts. In later stages, we boosted the contribution of the code-switchingdata.

Stage 2 For the second stage, we shuffled several grammatical error correctiondatasets. These are NUCLE [**?**], FCE [**?**], W&I Locness [], Lang-8 [] and our 2 newly created code-switchingdatasets. As our genuine code-switchingdataset is a subset of the private Lang-8 corpus, we checked and removed any duplicates. Table **??** shows the overall contributions of each corpus towards the stage 2 dataset. Similar to the previous stage, the data was split into train and validation sets.

Stage 3 For the final stage, we combined the high quality W&I Locness dataset with a sampled subset of the genuine code-switchingdata and a sampled subset of the synthetic code-switchingtexts. Again, the stage 3 dataset is split into train and validation sets. The remaining unused subset of the

D Inference Hyperparameters

After training our model, we used the validation dataset from stage 3 to tune 2 inference parameters. These are:

- `additional_confidence`—This value is added to the probability of the `$KEEP` token. If this value is high, recall is likely to decrease and precision increase. The grid search found the best value of this to be 0.
- `min_error_probability`—For a change to be made to a sentence, the probability of at least one token in the sentence being an error must be higher than the `min_error_probability`. If this value is high, then precision is likely to be higher and recall lower. The grid search found the best value of this to be 0.4.

E Error Type Analysis of Proposed Model

By exploring the ERRANT error classifications of our proposed model when applied to the code-switchingtest dataset, we can further explore the effectiveness of our synthetic data in addressing the problematic areas identified in Appendix **??**. A breakdown of the precision, recall and $F_{0.5}$ score for each of the previously identified categories is shown in Table **??**.

Table 6: Precision (P), Recall (R) and $F_{0.5}$ Score of Our Proposed Model for Targeted Error Types in the Genuine CSW Test Dataset

Category	P	R	$F_{0.5}$
NOUN	0.2857	0.0833	0.1923
PRON	0.7647	0.4643	0.6771
PUNCT	0.7143	0.1139	0.3460
WO	0.7778	0.2000	0.4930