Appendix for "Adapting Sequence Models for Sentence Correction"

Allen Schmaltz* **Yoon Kim** **Alexander M. Rush** **Stuart M. Shieber**

Harvard University

{schmaltz@fas,yoonkim@seas,srush@seas,shieber@seas}.harvard.edu

*Part of this work was completed while as an intern at Rakuten.

A Supplemental Material

Additional Model Training and Inference Details We provide additional replication details for our experiments here. Our code and related materials are available at the following url: https://github.com/allenschmaltz/grammar.

The training and tuning sizes of the AESW dataset are those after dropping sentences exceeding 126 tokens on the source or target side (in source sequences or target sequences with diff annotation tags) from the raw AESW dataset. All evaluation metrics on the development and test set are on the data without filtering based on sentence lengths.

As part of preprocessing, the sentences from the AESW XML are converted to Penn Treebank-style tokenization. Case is maintained and digits are not replaced with holder symbols for the sequence-to-sequence models. For the SMT models, the truecasing[1] and tokenization pipeline of the publicly available code is used. For consistency, all model output and all reference files are converted to cased Moses-style tokenization prior to evaluation.

For the CHAR model, the L_2-normalized gradients were constrained to be ≤ 1 (instead of ≤ 5 with the other models), and our learning rate schedule started the learning rate at 0.5 (instead of 1 for the other models) for stable training. The maximum sequence length of 421 was used for models given character sequences, which was equivalent to the maximum sequence length of 126 used for models given word sequences. The maximum sequence lengths were increased by 1 for the models with the +DOM features. The training and tuning set sizes cited in Section 3 are the number of sentences from the raw dataset after dropping sentences exceeding these maximum sequence lengths.

In practice, we were able to train each of the purely character-based models (e.g., the CHAR+BI+DOM model) with a single NVIDIA Quadro P6000 GPU with 24 GB of memory in about 3 weeks with a batch size of 12.

For the sequence-to-sequence models, the closed vocabularies were restricted to the 50,000 most common tokens, and a single special <unk> token was used for all remaining low frequency tokens. An <unk> token generated in the target sentence by the WORD and CHARCNN models was replaced with the source token associated with the maximum attention weight. The "open" vocabularies were only limited to the space of characters seen in training.

For the phrase-based machine translation baseline model from the work of Junczys-Dowmunt and Grundkiewicz (2016), for dense features, we used the stateless edit distance features and the stateful Operation Sequence Model (OSM) of Durrani et al. (2013)[2]. Since for our controlled data experiments we removed the language model features associated with external data, we did not use the word-class language model feature, so for the sparse features, we used the set of edit operations on "words with left/right context of maximum length 1 on words" (set "E0C10" from the original paper), instead of those dependent on word classes.

The training and tuning splits for the phrase-based machine translation models were the same as for the sequence-to-sequence models. For tuning, we used Batch-Mira, setting the background corpus decay rate to 0.001, as in previous work.

[1] Here, the truecase language model is created from the training $\mathbf{t}$ sequences (or where applicable, the target with diffs).

[2] The OSM features use the SRI Language Modeling Toolkit (SRILM) (Stolcke, 2002).

As in previous work, we repeated the tuning process multiple times (in this case, 5 times) and averaged the final weight vectors.

The sequence-to-sequence models were decoded with a beam size of 10.

Decoding of the SMT models used the same approach of Junczys-Dowmunt and Grundkiewicz (2016) (i.e., the open-source Moses decoder run with the cube pruning search algorithm).

In our experiments, we do not include additional paragraph context features, since the underlying AESW data appears to have been collected such that nearly all paragraphs (including those containing a single sentence) contain at least one error; thus, modeling paragraph information provides additional signal that seems unlikely to reflect real-world environments.

CoNLL-2014 Shared Task For training, we used the copy of the Lang-8 corpus distributed in the repo for the code of Junczys-Dowmunt and Grundkiewicz (2016): https://github.com/grammatical/baselines-emnlp2016. We filtered the Lang-8 data to remove duplicates and target sentences containing emoticon text, informal colloquial words (e.g., "haha", "lol", "yay"), and non-ascii characters. Target sentences not starting with a capital letter were dropped, as were target sentences not ending in a period, question mark, exclamation mark, or quotation mark. (Target sentences ending in a parenthesis were dropped as they often indicate informal additional comments from the editor.) In the combined NUCLE and Lang-8 training set, source sentences longer than 79 tokens and target sentences longer than 100 tokens were dropped. This resulted in a training set with 1,470,992 sentences. Diffs were created using the Python class difflib.SequenceMatcher.

For tuning on the dev set[3], a coarse grid search between 0 and 1.0 was used to set the four bias parameters associated with each diff tag. (Training was performed without re-weighting.) The bias parameter (in this case 0.7) yielding the highest M^2 score on the decoded dev set was chosen for use in evaluation of the final test set. The M^2 scores across the tuning runs on the dev set for the WORD+BI model are shown in Table 1.

For future comparisons to our work on the CoNLL-2014 shared task data, we recommend using the preprocessing scripts provided in our code repo (https://github.com/allenschmaltz/grammar).

[3]Previous work, such as Junczys-Dowmunt and Grundkiewicz (2016), also used the CoNLL-2013 set for tuning.

Bias parameter	Precision	Recall	$F_{0.5}$
0.0	72.34	0.97	4.60
0.1	69.74	1.51	6.96
0.2	72.00	2.57	11.23
0.3	69.05	4.14	16.68
0.4	67.19	6.08	22.31
0.5	61.03	8.76	27.82
0.6	51.75	11.41	30.31
0.7	**46.66**	**15.35**	**33.14**
0.8	40.01	18.68	32.57
0.9	34.49	22.08	31.00
1.0	30.17	24.90	28.94

Table 1: M^2 scores on the CoNLL-2013 dev set for the WORD+BI model.

Table 2 The seven columns of Table 2 appearing in the main text are Micro $F_{0.5}$ scores for the errors within each frequency grouping. There are a total of 39,916 replacement changes. The replacements are grouped in regard to the changes within the opening and closing deletion tags and subsequent opening and closing insertion tags, as follows: (1) whether the replacement involves (on the deletion and/or insertion side) a single punctuation symbol (comma, colon, period, hyphen, apostrophe, quotation mark, semicolon, exclamation, question mark); (2) whether the replacement involves (on the deletion and/or insertion side) a single article (a, an, the); (3) non-article, non-punctuation grouped errors with frequency greater than 100 in the gold training data; (4) non-article, non-punctuation grouped errors with frequency less than or equal to 100 and greater than or equal to 5; (5) non-article, non-punctuation grouped errors with frequency less than 5 and greater than or equal to 2; (6) non-article, non-punctuation grouped errors with frequency equal to 1; (7) non-article, non-punctuation grouped errors that never occurred in the training data. Note that the large number of unique instances occurring for the "punctuation" and "articles" classes are a result of the large number of errors that can occur on the non-article, non-punctuation side of the replacement. The Micro $F_{0.5}$ scores are calculated by treating each individual error (rather than the agglomerated classes here) as binary classifications.

References

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013. Can markov models over minimal translation units help phrase-based smt? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 399–405, Sofia, Bulgaria. Association for Computational Linguistics.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1546–1556, Austin, Texas. Association for Computational Linguistics.

Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904, Denver.