A Low-Resource Approach to the Grammatical Error Correction of Ukrainian

Frank Palma Gomez
Queens College, CUNY
frankpalma12@gmail.com

Alla Rozovskaya
Queens College, CUNY
arozovskaya@qc.cuny.edu

Dan Roth
University of Pennsylvania
danroth@seas.upenn.edu

Abstract

We present our system that participated in the shared task on the grammatical error correction of Ukrainian. We have implemented two approaches that make use of large pre-trained language models and synthetic data, that have been used for error correction of English as well as low-resource languages. The first approach is based on finetuning a large multilingual language model (mT5) in two stages: first, on synthetic data, and then on gold data. The second approach trains a (smaller) seq2seq Transformer model pre-trained on synthetic data and finetuned on gold data. Our mT5-based model scored first in "GEC only" track, and a very close second in the "GEC+Fluency" track. Our two key innovations are (1) finetuning in stages, first on synthetic, and then on gold data; and (2) a high-quality corruption method based on round-trip machine translation to complement existing noisification approaches.[1]

1 Introduction

This paper describes our submission in the shared task on the Grammatical Error Correction (GEC) of Ukrainian (Syvokon and Romanyshyn, 2023) that was organized as part of the Workshop on Ukrainian Natural Language Processing (UNLP 2023), in conjunction with EACL 2023.

Ukrainian is an Indo-European language from the East-Slavic language family, and is most closely related to Russian and Belarusian. In the context of GEC, Ukrainian is a low-resource language and is under-explored. A dataset of Ukrainian native and non-native texts annotated for errors was recently released (Syvokon and Nahorna, 2021), however, to the best of our knowledge, no systems have been benchmarked on this dataset.

We have developed two approaches. The first approach is based on the method proposed in earlier work (Rothe et al., 2021) that finetunes a multilingual mT5 model on gold GEC data.[2] Because mT5 is pre-trained with an objective that is not appropriate for GEC, we propose a 2-stage finetuning strategy, where we finetune first on native data with synthetic noise, and then further finetune on the gold GEC data. We show that this two-stage approach is beneficial and provides a large boost compared to an mtT5 model finetuned on gold data only. Our model scored first in the "GEC only" track and a very close second in the "GEC+Fluency" track (0.08 point difference from the top submission).

Our second system is a smaller seq2seq Transformer model pre-trained on synthetic data and finetuned on gold data. We propose a novel method of generating synthetic errors using back-translation. Unlike previous approaches, we do not use full-sentence translations but only extract back-translation pairs that are then used for introducing errors in native data.

We present related work on GEC in Section 2. Section 3 describes our approach. Section 4 briefly describes the Ukrainian GEC dataset. Section 5 presents our experimental results on the validation and test data, as well as additional evaluation by error type. Section 6 concludes.

2 Background

Most effort in GEC research concentrated on correcting errors made by English as second language writers. More recently, there has been interest in developing approaches and resources in GEC for other languages, including Arabic (Mohit et al., 2014), German (Naplava and Straka, 2019), Russian (Rozovskaya and Roth, 2014), Chinese, and Spanish (Rothe et al., 2021). Earlier approaches to GEC include rule-based methods and machine learning classifiers for correcting a specific type of mistake (e.g. article or preposition) (Tetreault

[1] Code is available at https://github.com/knarfamlap/low-resource-gec-uk

[2] We used the smaller (base and large) models only in our experiments, due to the sizes of mT5 models.

Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP), pages 114–120

et al., 2010; Foster, 2010; Rozovskaya and Roth, 2013; Dahlmeier and Ng, 2012). For an overview of approaches and methods in GEC, we refer the reader to Bryant et al. (2022).

Current approaches to GEC can be broken down into two categories: sequence-to-sequence (seq2seq) generation (Jianshu et al., 2017; Chollampatt and Ng, 2018; Grundkiewicz and Junczys-Dowmunt, 2019), and sequence-to-editing (seq2edits) (Omelianchuk et al., 2020; Awasthi et al., 2019; Li and Shi, 2021). Both approaches achieve state-of-the-art performance on English GEC. In the seq2edits framework, the task is viewed as a sequence labeling problem (Omelianchuk et al., 2020) that tags text spans with appropriate error tags, leaving the rest of the text unchanged.

Because the seq2edits approach requires human input, as it depends on constructing language-specific edit operations, we adopt the seq2seq framework. Seq2seq approaches have demonstrated strong empirical results in GEC (Chollampatt and Ng, 2018; Yuan and Briscoe, 2016; Grundkiewicz et al., 2019; Grundkiewicz and Junczys-Dowmunt, 2019; and R. Grundkiewicz and S. Guha and K. Heafield, 2018; Kiyono et al., 2019a; Zhao et al., 2019; Jianshu et al., 2017; Yuan and Briscoe, 2016; Katsumata and Komachi, 2019; Xie et al., 2018). Due to lack of gold training data, it is common to first *pre-train* a model on native data where the source side has been corrupted with artificial noise. The pre-trained model is typically further finetuned on the available gold data.

Pre-trained language models (PLMs) Recently, finetuning PLMs has become a standard paradigm for many NLP tasks. In GEC, PLMs have been mainly used in English where models have been finetuned on large amounts of hand-labeled data (Kaneko et al., 2020; Malmi et al., 2019; Omelianchuk et al., 2020). Katsumata and Komachi (2020) apply PLMs in a multilingual setting, by finetuning BART (Lewis et al., 2020). However, even when using a large number of hand-labeled examples, they achieve results that are way below state-of-the-art.

In this work we adopt the approach of Rothe et al. (2021) and make use of mT5 (Xue et al., 2021), a multilingual variant of T5 (Raffel et al., 2020), a pre-trained text-to-text Transformer. mT5 has been pre-trained on mC4 corpus, a subset of Common Crawl, covering 101 languages and composed of about 50 billion documents (Xue et al., 2021).

Rothe et al. (2021) finetune mT5 on GEC gold data for Russian, German, and Czech languages, although SOTA results are only achieved, when they re-train mT5 with a different objective and use an extremely large model xxl with 13B parameters. We use the original mT5 models of smaller sizes and show that it is possible to achieve competitive results by pre-training first on synthetic data.

3 The Models

We have implemented two approaches that draw on methods that showed competitive performance in multilingual low-resource settings. Our first (larger) model makes use of mT5 but is finetuned in two stages – on synthetic data (we refer to this stage as pre-training on synthetic data), and then finetuning on gold data. Our second (smaller) model is a seq2seq Transformer model pre-trained on synthetic data (from scratch) and finetuned on gold data. As our baseline for the second model, we use a model pre-trained on synthetic data generated using standard spell-based transformations. We show that adding synthetic noise from back-translations results in a 3-point improvement over the baseline. Because both approaches make use of synthetic data, we describe the data generation methods below.

Generating synthetic data Standard *data corruption methods* typically use a variety of heuristics: random character and token transformations (Schmaltz et al., 2016; Lichtarge et al., 2019a), confusion sets generated from a spellchecker (Grundkiewicz and Junczys-Dowmunt, 2019; Naplava and Straka, 2019), or a morphological analyzer (Choe et al., 2019), or round-trip translation (Lichtarge et al., 2019a).

We have experimented with two baseline data generation techniques for low-resource settings: (1) spell-based transformations and (2) part-of-speech (POS)-based transformations. Both of the methods rely on the idea of using *confusion sets* that specify for each target word occurring in a native corpus a list of highly confusable words. These lists are used to generate synthetic errors.

Spell-based transformations This approach showed state-of-the-art performance in English (Bryant et al., 2019; Grundkiewicz and Junczys-Dowmunt, 2019; Grundkiewicz et al., 2019), and other languages (Naplava and Straka, 2019; Flachs et al., 2021). Spell-based confusions include highly confusable words based on edit

distance obtained from a dictionary available in a spellchecker. Because Aspell is an open-source spellchecker, it is common to use Aspell to generate spell-based confusion sets. We use Aspell with the Ukrainian dictionary in this work to create spell-based confusions. More detail about the method can be found in Naplava and Straka (2019). We follow Naplava and Straka (2019) for the parameter values for token replacement, deletions, and insertions.

POS-based transformations Confusion sets in this method are generated based on part-of-speech (POS) tag of the target word to be replaced: given a word and its POS tag (Choe et al., 2019), the target word is replaced with its inflectional variant that corresponds to the same base form (e.g. "walks" would be replaced with "walking", "walked" or "walk"). Flachs et al. (2021) use Unimorph morphological analyzer and tagger (McCarthy et al., 2020). Although POS-based transformations showed promising results for Russian, our initial experiments using pymorphy (Korobov, 2015) did not yield competitive results, and we do not report these experiments.

Back-translation (BT) The motivation for using BT is to generate more diverse errors that cannot be generated using the baseline spell-based transformations. We hypothesize that many fluency errors, such as choosing an incorrect word, will manifest themselves in the machine translation output as back-translated words that are semantically close to the target. The input to BT are sentences from a native Ukrainian corpus. The sentences are translated into another language (pivot), and then back into *Ukrainian*. We use English as the pivot: A sentence is translated into English, where top n translation hypotheses are generated. For each hypothesis, top m back-translations into *Ukrainian* are generated. For each unique word in *Ukrainian*, the back-translated words that are aligned to it are treated as potential synthetic errors.

Crucially, in contrast to other approaches that employ back-translation (Lichtarge et al., 2019b), we do not make use of the entire resulting back-translated sentences, but only generate targeted confusion sets of relevant errors that are used to corrupt the data. Further, we generate multiple hypotheses in each direction. We use the BT approach in conjunction with the spell-based transformations (see Section 5). We use the neural machine translation systems of Tiedemann and Thottingal (2020) to translation from Ukrainian into English and back.

	Percentage (%)		
Error	**Train**	**Valid (R_1)**	**Valid (R_2)**
Punctuation	36.9	32.8	29.8
Spelling	19.5	21.8	17.8
F/PoorFlow	8.9	12.1	16.0
F/Style	8.5	8.7	9.1
G/Case	6.2	6.5	3.7
F/Calque	6.4	4.1	4.9
G/Structure	2.3	2.2	3.8
F/Repetition	1.2	2.2	1.9
F/Collocation	1.2	1.6	1.2
F/Other	0.7	-	0.3
G/Prep	1.3	1.5	2.5
G/Number	0.9	1.0	1.3
G/Conjunction	1.1	1.0	0.7
G/Gender	1.3	0.9	0.7
G/VerbVoice	0.7	0.7	1.0
G/VerbAForm	0.2	0.7	0.1
G/Tense	0.4	0.6	1.4
Other	1.0	0.4	3.3
G/Other	0.6	0.4	-
G/PartVoice	0.3	0.3	0.1
G/Particle	0.2	0.2	0.2
G/Comparison	0.4	0.2	0.1
G/Participle	-	0.1	0.1
G/Aspect	0.2	0.1	0.3
Total	35,431	1,922	2,731

Table 1: Learner error distributions by category (on the training and validation data). *G* stands for grammar, and *F* stands for fluency. The validation data has two references per sentence (R_1 and R_2).

4 The Ukrainian GEC Data

The data used in the shared task comes from UA-GEC, a corpus of social media texts written by native speakers and learners of Ukranian (Syvokon and Nahorna, 2021). The shared task organizers have provided training and validation data. The training data is annotated with 1 reference, and the validation set is annotated with 2 references for each sentence. The gold corrections are provided in the standard M2 format (Ng et al., 2014), and the edits are labeled with the corresponding error tags. There are 24 error categories, broadly classified with a prefix into Grammar (G) and Fluency (F) corrections (with the exception of spelling and punctuation errors that do not have a prefix). The shared task includes 2 tracks: "GEC+Fluency", where the systems are evaluated with respect to all errors, and the "GEC only" track where the fluency

edits are removed. Table 1 shows the distribution of errors in the training and validation data. As can be observed, punctuation and spelling errors constitute the majority of edits (over 50%), and grammar errors are less frequent. This may be because the Ukrainian corpus contains a lot of data from native speakers, as opposed to language learners (Syvokon and Nahorna, 2021).

5 Experiments and Results

Below, we present experimental results for the two models that we implemented. Our submissions for both tracks are the same, except that the models are finetuned on the gold data for each respective track. We first present a set of experiments on the data in the "GEC+fluency" track. We report results of the submitted systems for both tracks in Section 5.3.

Corrupting monolingual Ukrainian data Both models use synthetic data. We corrupt sentences from the Ukrainian partition of CC-100 (Wenzek et al., 2020), which contains high-quality data from Common Crawl. We tokenize the data using Stanza (Qi et al., 2020), the same tokenizer that is used to tokenize the gold data. We use spell-based transformations (see Section 3) to corrupt the monolingual data (but see also 5.2).

Evaluation We report the scores measured by ERRANT scorer (Felice and Briscoe, 2015), and report performance on correction.

5.1 mT5-Based Models

First, we evaluate mT5-based models. We have experimented with 2 models: base and large. Although xl and xxl models showed much stronger performance (Rothe et al., 2021), these models were too large (3.7B and 13B parameters, respectively). mT5 base and mT5 large have 580M and 1.3B parameters, respectively. We first finetune both models on the gold training data and evaluate on the validation set (see Table 2).

Pre-training on synthetic data Because mT5 has been pre-trained with span-prediction objective that is not optimal for GEC, Rothe et al. (2021) re-train the model, by splitting the paragraphs into individual sentences and corrupting the sentences with a set of operations that drop, insert, or swap tokens and characters. Their resulting gT5 model significantly outperforms the finetuned mT5 models. Since gT5 is not publicly available, we make use of the mT5 models, however, to account for the fact that mT5 may not be optimal for GEC, we introduce an additional pre-training step and pre-train mT5 on synthetic data with spell-based corruptions (see Section 3). We finetune mT5, using the original hyper-parameters in Xue et al. (2021). When finetuning, we utilize a max context length of 128 tokens, a batch size of 32, and a global seed of 42 for all experiments related to mT5.

	P	R	$F_{0.5}$
mT5 base	63.64	33.29	53.83
mT5 large	65.26	39.74	**57.83**

Table 2: mT5 models finetuned on gold training data. Results on **valid** ("GEC+Fluency"). Best result is in bold.

Model	P	R	$F_{0.5}$
mT5 base	63.64	33.29	53.83
mT5 large	65.26	39.74	57.83
mT5 base + 2M synth.	72.05	39.69	61.94
mT5 large + 2M synth.	73.95	41.84	64.11
mT5 large + 10M synth.	72.08	47.87	**65.45**

Table 3: mT5 pre-trained on synthetic data, and finetuned on gold training data. Results on **valid** ("GEC+Fluency"). Best result is in bold.

Results are shown in Table 3. Pre-training on synthetic data boosts the performance significantly, but almost 7 points. Increasing the size of the synthetic data used for pre-training further boosts the performance by 1 F-score point.

5.2 Transformer seq2seq Models Trained on Synthetic Data

The model We use the Transformer sequence-to-sequence model (Vaswani et al., 2017) implemented in the Fairseq toolkit (Ott et al., 2019). We use the "Transformer (big)" settings and the parameters specified in (Kiyono et al., 2019b) for `Pretrain` setting. The models are pre-trained on synthetic data until convergence using 3 seeds (1,2, and 3) and then further finetuned on gold training data. The gold training data is also used as the validation set. We ensemble the best checkpoints from each run during inference.

Pre-training with spell-based synthetic errors Table 5 shows experimental results on the validation set. The top two rows show models pre-trained on 15M synthetic sentences (single model results and an ensemble of 3 best checkpoints).

Back-translation based errors Our next experiment evaluates the contribution of back-translation based errors. The errors are introduced on top of

		GEC+Fluency			GEC only		
Model	**Number of params.**	**P**	**R**	**$F_{0.5}$**	**P**	**R**	**$F_{0.5}$**
mT5 large	1.3B	73.21	53.22	68.09	76.81	61.39	73.14
seq2seq	275M	69.91	53.78	65.96	72.32	63.13	70.27

Table 4: Results on the test data of the submitted systems for both tracks.

Model	**P**	**R**	**$F_{0.5}$**
Spell (15M, single)	62.0	46.8	58.2
Spell (15M, ens.)	65.6	47.4	60.9
Spell+BT (15M, single)	65.1	48.5	60.9
Spell+BT (15M, ens.)	68.3	49.0	63.3
Spell+BT (35M, single)	63.8	50.0	60.4
Spell+BT (35M, ens.)	67.8	50.5	**63.4**

Table 5: Seq2seq models pre-trained on synthetic data and finetuned on the gold training data. Results on the "GEC+Fluency" validation set (average over 3 random seeds for single models). *BT* stands for back-translation. Best result is in bold.

	Recall	
Error	**mT5 large**	**Seq2seq**
Calque	23.1	22.5
Case	23.3	15.0
Flow	8.0	10.6
Punc.	65.4	67.5
Spelling	48.6	51.2
Structure	13.5	11.7
Style	10.6	13.4

Table 6: Recall performance per error type for the most frequent error types on the validation set for the two submitted systems.

the spell-based confusions, with an error rate of 10%. We note that because these errors do not target every word, on average 5% of additional words are being corrupted in this stage. The second segment of Table 5 illustrates that adding back-translation errors improves the results by 3 points.

Effect of the synthetic data size Finally, we train models on more synthetic data (35M examples). We do not observe an improvement compared to using 15M examples (bottom segment of Table 5).

5.3 Submitted Systems

For the mT5-based model, we submitted an mT5 large pre-trained on 10M synthetic examples, and further finetuned on the gold data. The seq2seq model is pre-trained on 35M synthetic examples (Spell+BT) and finetuned on gold training data. We use 3 random seeds and the inference is an ensemble over the best checkpoints for the 3 runs. For each track, we finetune on the gold data for the corresponding track. Results are shown in Table 4. Note that the mT5 model is finetuned on the gold training and validation data in the "GEC+Fluency" track, and is finetuned on the gold training data in the "GEC only" track. Seq2seq models are finetuned on the gold training data for both tracks.

5.4 Evaluation by Error Type

Evaluating performance by individual error type is extremely useful, as it allows us to understand what type of mistakes each model is good at correcting, and which errors are more difficult. However, evaluating by error type requires classifying the edits made by the automated systems. In other languages, automatic tools for classifying edits have been built (Bryant et al., 2017; Belkebir and Habash, 2021; Rozovskaya, 2022). However, we can compute the recall of each model, by using gold error tags available in the M2 files. We report recall for the submitted systems (on the validation data) for the most frequent error types. Results are shown in Table 6. Note that because we cannot evaluate the precision of correcting these error types, these results cannot be used to directly compare performance on different errors. Nevertheless, this evaluation suggests that currently both systems mainly correct punctuation and spelling errors, whereas fluency errors, such as flow, and style prove to be the most challenging.

6 Conclusion

We have presented our submission that participated in the shared task on the Grammatical Error Correction of Ukrainian. Our submission includes two systems. The first system pre-trains mT5 on synthetic data and finetunes on the gold GEC data. We have shown that introducing this two-stage approach is crucial to achieving strong results when using mT5. We have also proposed a novel synthetic data generation method that extracts confusion pairs from multiple back-translation hypotheses that are aligned with the original sentence.

Limitations

The results shown in this work may not necessarily reflect performance on other languages with similar amounts of resources or even Ukrainian language error correction performed on a different domain. The methods described in this work require use of GPU resources that may not be available to all researchers.

Acknowledgments

We thank the anonymous reviewers for their insightful comments.

References

M. Junczys-Dowmunt and R. Grundkiewicz and S. Guha and K. Heafield. 2018. Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task. In *NAACL*.

A. Awasthi, S. Sarawagi, R. Goyal, S. Ghosh, and V. Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *EMNLP-IJCNLP*.

R. Belkebir and N. Habash. 2021. Automatic error type annotation for arabic. In *CoNLL*.

C. Bryant, M. Felice, Ø. Andersen, and T. Briscoe. 2019. The BEA-19 shared task on grammatical error correction. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

C. Bryant, M. Felice, and T. Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *ACL*.

C. Bryant, Z. Yuan, M. R. Qorib, H. Cao, H. T. Ng, and T. Briscoe. 2022. Grammatical error correction: A survey of the state of the art. *Computational Linguistics*.

Y. J. Choe, J. Ham, K. Park, and Y. Yoon. 2019. A neural grammatical error correction system built on better pre-training and sequential transfer learning . In *BEA Workshop*.

S. Chollampatt and H.T. Ng. 2018. A Multilayer Convolutional Encoder-Decoder Neural Network for Grammatical Error Correction . In *Proceedings of the AAAI*. Association for the Advancement of Artificial Intelligence.

D. Dahlmeier and H. T. Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of EMNLP-CoNLL*.

M. Felice and T. Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *NAACL-HLT*.

S. Flachs, F. Stahlberg, and S. Kumar. 2021. Data strategies for low-resource grammatical error correction. In *BEA Workshop*.

J. Foster. 2010. "cba to check the spelling": Investigating parser performance on discussion forum posts. In *NAACL*.

R. Grundkiewicz and M. Junczys-Dowmunt. 2019. Minimally-augmented grammatical error correction. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT*.

R. Grundkiewicz, M. Junczys-Dowmunt, and K. Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

J. Jianshu, Q. Wang, K. Toutanova, Y. Gong, S. Truong, and Jianfeng J. Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *ACL*.

M. Kaneko, M. Mita, S. Kiyono, J. Suzuki, and K. Inui. 2020. Encoder-Decoder Models Can Benefit from Pre-trained Masked Language Models in Grammatical Error Correction. In *ACL*.

S. Katsumata and M. Komachi. 2019. (almost) unsupervised grammatical error correction using synthetic comparable corpus. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

S. Katsumata and M. Komachi. 2020. Stronger baselines for grammatical error correction using a pretrained encoder-decoder model. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*.

S. Kiyono, J. Suzuki, M. Mita, T. Mizumoto, and K. Inui. 2019a. An empirical study of incorporating pseudo data into grammatical error correction. In *EMNLP-IJCNLP*.

S. Kiyono, J. Suzuki, M. Mita, T. Mizumoto, and K. Inui. 2019b. An empirical study of incorporating pseudo data into grammatical error correction. In *EMNLP*.

M. Korobov. 2015. Morphological analyzer and generator for russian and ukrainian languages. In *Analysis of Images, Social Networks and Texts*.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*.

P. Li and S. Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for Chinese grammatical error correction. In *ACL*.

J. Lichtarge, C. Alberti, S. Kumar, N. Shazeer, N. Parmar, and S. Tong. 2019a. Corpora Generation for Grammatical Error Correction. In *NAACL*.

J. Lichtarge, C. Alberti, S. Kumar, N. Shazeer, N. Parmar, and S. Tong. 2019b. Corpora generation for grammatical error correction . In *NAACL*.

E. Malmi, S. Krause, S. Rothe, D. Mirylenka, and A. Severyn. 2019. Encode, tag, realize: High-precision text editing. In *EMNLP-IJCNLP*.

A. D. McCarthy, C. Kirov, M. Grella, A. Nidhi, P. Xia, K. Gorman, E. Vylomova, S. J. Mielke, G. Nicolai, M. Silfverberg, T. Arkhangelskiy, N. Krizhanovsky, A. Krizhanovsky, E. Klyachko, A. Sorokin, J. Mansfield, V. Ernštreits, Y. Pinter, C. L. Jacobs, R. Cotterell, M. Hulden, and D. Yarowsky. 2020. UniMorph 3.0: Universal Morphology . In *LREC*.

B. Mohit, A. Rozovskaya, N. Habash, W. Zaghouani, and O. Obeid. 2014. The first QALB shared task on automatic text correction for Arabic. In *ANLP Workshop*.

J. Naplava and M. Straka. 2019. Grammatical error correction in low-resource scenarios. In *W-NUT Workshop*.

H.T. Ng, S.M. Wu, T. Briscoe, C. Hadiwinoto, R. Susanto, and C. Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of CoNLL: Shared Task*.

K. Omelianchuk, V. Atrasevych, A. Chernodub, and O. Skurzhanskyi. 2020. GECToR ? Grammatical Error Correction: Tag, Not Rewrite . In *Building Educational Applications Workshop (BEA)*.

M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *NAACL*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

S. Rothe, J. Mallinson, E. Malmi, S. Krause, and A. Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *ACL*.

A. Rozovskaya. 2022. Automatic Classification of Russian Learner Errors. In *LREC*.

A. Rozovskaya and D. Roth. 2013. Joint learning and inference for grammatical error correction. In *Proceedings of EMNLP*.

A. Rozovskaya and D. Roth. 2014. Building a State-of-the-Art Grammatical Error Correction System. In *Transactions of ACL*.

A. Schmaltz, Y. Kim, A. M. Rush, and S. M. Shieber. 2016. Sentence-level grammatical error identification as sequence-to-sequence correction . In *Proceedings of the NAACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-16)*.

O. Syvokon and O. Nahorna. 2021. UA-GEC: Grammatical Error Correction and Fluency Corpus for the Ukrainian Language.

O. Syvokon and M Romanyshyn. 2023. The UNLP 2023 shared task on grammatical error correction for Ukrainian. In *Proceedings of the Second Ukrainian Natural Language Processing Workshop*.

J. Tetreault, J. Foster, and M. Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of ACL*.

J. Tiedemann and S. Thottingal. 2020. OPUS-MT ? Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. 2017. Attention is all you need. *In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems*.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*.

Z. Xie, G. Genthial, S. Xie, A. Y. Ng, and D. Jurafsky. 2018. Noising and Denoising Natural Language: Diverse Backtranslation for Grammar Correction. In *NAACL*.

L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. . In *NAACL*.

Z. Yuan and T. Briscoe. 2016. Grammatical error correction using neural machine translation. In *NAACL*.

W. Zhao, L. Wang, K. Shen, R. Jia, and J. Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *NAACL*.