Comparative Study of Models Trained on Synthetic Data for Ukrainian Grammatical Error Correction

Maksym Bondarenko
Columbia University
New York, USA
mb5018
@columbia.edu

Artem Yushko
Carleton College
Northfield, USA
yushkoa
@carleton.edu

Andrii Shportko
Northwestern University
Evanston, USA
andreshportko2026
@u.northwestern.edu

Andrii Fedorych
Taras Shevchenko
National University
Kyiv, Ukraine
andrii.t.fedorych
@gmail.com

Abstract

The task of Grammatical Error Correction (GEC) has been extensively studied for the English language. However, its application to low-resource languages, such as Ukrainian, remains an open challenge. In this paper, we develop sequence tagging and neural machine translation models for the Ukrainian language as well as a set of algorithmic correction rules to augment those systems. We also develop synthetic data generation techniques for the Ukrainian language to create high-quality human-like errors. Finally, we determine the best combination of synthetically generated data to augment the existing UA-GEC corpus and achieve the state-of-the-art results of 0.663 $F_{0.5}$ score on the newly established UA-GEC benchmark. The code and trained models will be made publicly available on GitHub and HuggingFace. [1] [2]

1 Introduction

Grammatical error correction (GEC) models have achieved significant results for English (Bryant et al., 2022). However, GEC for Ukrainian is an open challenge for multiple reasons. Even though Ukrainian is the official language of Ukraine with more than 40 million speakers worldwide[3], there are still few NLP corpora, studies, or tools available (Pogorilyy and Kramov, 2020). This lack of resources may be explained by the small pool of speakers (less than one percent of the world population), but also the many intrinsic difficulties of Ukrainian, including the historical suppression of the language by the USSR, the high prevalence of its mixture with Russian (surzhyk), and the formal context in which texts are commonly written (Buk and Rovenchak, 2003). The biggest difficulty is that Ukrainian is a low-resource language, and has only one annotated GEC dataset available (Syvokon and Nahorna, 2021) and few high-quality pre-trained transformer models compared to English[4]. Additionally, Ukrainian is a morphologically complex language, which makes its grammar correction more challenging with token-based models. The Ukrainian language also does not have rigid word order, which makes syntactic analysis more difficult. Finally, Ukrainian grammar contains lots of exceptions that are not widely known within the main body of native speakers, which makes GEC even more challenging (Syvokon and Nahorna, 2021).

In this paper, we outline the thought process behind and the development of seq2tag and NMT models for the Ukrainian language. Moreover, we outline the creation of algorithmic correction rules in addition to those architectures. We also assemble a clean corpus of 1mln sentences and develop generators of high-quality human-like errors for it. Finally, we compare our models and find the best combination of synthetically-made data with the existing UA-GEC corpus and achieve the best results on the newly established UA-GEC benchmark.

2 Related Work

2.1 Models for GEC

Two dominant state-of-the-art approaches for GEC in the English language are transformer-based neural machine translation (NMT) and sequence tagging (seq2tag) (Bryant et al., 2022). NMT approach treats GEC as a translation task, where the model must learn to translate from the errorful language to grammatically sound sentences (Sennrich et al., 2016). Recent advances in machine translation mean that many effective architectures and pre-trained models for translation exist, and applying them to GEC requires simple fine-tuning on GEC corpora. Sequence tagging as described in PIE (Awasthi et al., 2019) leverages the fact that

[1] `https://github.com/pravopysnyk-ai/unlp`
[2] `https://huggingface.co/Pravopysnyk`
[3] `https://photius.com/rankings/languages2.html`
[4] `https://huggingface.co/models?pipeline_tag=fill-mask&language=uk&sort=downloads`

most tokens in the sentence do not need to be corrected and treats GEC as a token classification task, where each token is assigned a label to either keep it as it is, delete it, or replace it. In the GECToR paper (Omelianchuk et al., 2020), they build on this idea by introducing a number of G-transforms that decrease the number of required labels without sacrificing error coverage to achieve better data efficiency. Both approaches have their strengths and weaknesses in application to the Ukrainian GEC. As the recent research (Flachs et al., 2021) showed, NMT models for non-English languages require less pre-training since translation models for them already exist. On the other hand, they have longer inference times than sequence tagging due to the need to generate the entire sequence and lower interpretability and customization due to the black-box nature of the models. Sequence tagging approach as described in GECToR (Omelianchuk et al., 2020) requires the development of many complex g-transforms, which must be even more numerous for Ukrainian due to its morphological complexity. Additionally, seq2tag requires more pretraining because there are no existing models for Ukrainian.

2.2 Data Generation

A common technique used in English for training neural GEC models is synthetic data generation (Flachs et al., 2021). Synthetic data generation is even more crucial for Ukrainian since no large natural corpora exist. Most commonly used in English techniques include rule-based generation, back-translation, and round-trip translation, as well as leveraging public editing data through datasets such as Lang8 and Wiki Edits (Stahlberg and Kumar, 2021). The advantage of rule-based generation is that it leverages fundamental asymmetry that generating errors is much simpler than correcting them and therefore is possible to do programmatically (Awasthi et al., 2019). Back translation is a reverse task that uses deep learning (DL) models to recreate error patterns in existing human-annotated datasets (Sennrich et al., 2016). Round-trip-translation is based on the assumption that many translation models are still imperfect and that flow and style mistakes will be produced through the chain of translation (Lichtarge et al., 2019). Finally, the Wiki Edits and Lang8 datasets are available for any language (Faruqui et al., 2018).

3 Models

3.1 NMT

Most GEC systems that perform best on GEC benchmarks are based on the NMT architecture[5]. To extend those results to the Ukrainian language, we take the publicly available pre-trained mBART-50 model[6] and fine-tune it on UA-GEC augmented with our synthetically generated data. We choose to focus on mBART as it has previously shown the most promising results in the MT setting among comparable models (Tang et al., 2020). We train models with weight decay rate of 0.01 and the well-established and reliable native tokenizer and optimizer publicly available at HuggingFace[7]. All NMT models for 5 epochs with batch size of 32 and learning rate of 2e-5 on a single A100 Tensor Core GPU available at Google Colab. Average training time is 15 minutes.

3.2 SEQ2TAG

Despite showing the best results on GEC benchmarks (Bryant et al., 2017), NMT-based GEC systems suffer from multiple issues which make them far less convenient for deployment in the real world:

- Slow inference speed.
- Low interpretability and explainability; they require additional functionality to explain corrections, e.g., grammatical error type classification (Bryant et al., 2019)

To develop an alternative model of sequence tagging, we adopted GECToR's approach (Omelianchuk et al., 2020). Our GEC sequence tagging model is an encoder made of pre-trained Ukrainian-specific XLM-ROBERTa transformer [8] stacked with two linear layers and with softmax layers on the top.

We create a system of hand-made token-level transformations T(x_i) to match the target text by applying them to the corresponding source tokens ($x_1 ... x_N$). According to previous research, transformations increase the coverage of grammatical

[5] http://nlpprogress.com/english/grammatical_error_correction.html

[6] https://huggingface.co/facebook/mbart-large-50

[7] https://huggingface.co/docs/transformers/preprocessing

[8] https://huggingface.co/ukr-models/xlm-roberta-base-uk

error corrections for limited output vocabulary size for the most common grammatical errors, such as Spelling, Noun Number, Subject-Verb Agreement, and Verb Form (Yuan, 2017). Since no research has been conducted on native/non-native mistakes in the Ukrainian language, we adopt the classification used in the English language and applied by the GECToR team (Omelianchuk et al., 2020).

On the basic level, we use four types of token-level transformations, adopted from (Omelianchuk et al., 2020):

1. $KEEP – keeps the current token unchanged
2. $DELETE – deletes the current token
3. $APPEND – adds a new token to the current one, followed by a space
4. $REPLACE – replaces the current token with a different one.

Then, we add them to our custom-made G-Transformations, outlined in the Synthetic Data Generation section.

To correct the text, for each input sentence token x_i, $1 \leq i \leq N$ from the source sequence $(x_1 ... x_N)$, the model predicts the tag-encoded token-level transformation $T(x_i)$. These predicted tag-encoded transformations are then applied to the sentence, resulting in a modified sentence.

We train models of this type with variable training parameters for each run. We provide detailed overview of those for each model in the source code. On average, seq2tag models train for 8 hours on Google Colab GPUs.

3.3 Rule-Based Correction

The final approach that we try to apply to the Ukrainian language is rule-based correction. All existing services for Ukrainian GEC are rule-based[9], so we developed a few rule-based corrections to augment our models as well. However, we found that this approach requires additional research, which falls outside the scope of this paper.

4 Synthetic Data Generation

To train and test our models, we rely on the generation of synthetic data. In the following subchapters, we explain what techniques we use to imitate natural errors for the large corpus of correct sentences.

Many of our modules include detailed controls that allow us to modify data to suit our needs. This means we can provide high-quality data, with error patterns representative of those available in human-annotated corpora.

4.1 Clean data

Generating errors in existing sentences (errorification) necessitates the existence of a large, error-free corpus to errorify. To address this, we turn to web scrapping, since multiple authors have already addressed its usefulness for low-resource languages (Ghani et al., 2001). One commonly used technique is to send requests composed of mid-frequency n-grams to a search engine to gather bootstrap URLs, which use a breadth-first strategy to crawl the web page in search of meaningful information, such as documents or words (Sharoff, 2006). This is the technique that we apply to different news websites. The raw HTML content is fetched and converted to UTF-8 using a mixture of requests and BeautifulSoup. Then, we fix the remaining encoding artifacts with ftfy (Speer, 2019) and remove unicode emojis. Another crucial step is to normalize the Unicode points used for dashes, spaces, quotes etc., and strip any invisible characters. Furthermore, to simplify the process of tokenization, we enforce a single convention for all spaces around quotes and colons, e.g. no space inside quotes colons after the closing quote. Finally, to split text into sentences, we implement pymorphy in Python and apply it in three main ways: existing newlines are preserved, colons and semi-colons are considered segmentation hints, and sentences are required to start with an uppercase.

As a result, we compose a corpus of 1,030,582 high-quality error-free sentences from 62K URLs across 3,472 domains.

4.2 Punctuation

The first kind of error we use is punctuation errors. As they attribute the most errors in the UA-GEC dataset (Syvokon and Nahorna, 2021) and most English-language datasets (Bryant et al., 2019), we infer that this is the most common kind of mistake. To synthetically generate punctuation errors, we create the error probability matrix that replaces each mark (space between words was also counted as a mark) with any other one according to the randomly generated probability. Then, this matrix is applied to each sentence from our dataset. The resulting matrix looks like this:

[9] https://languagetool.org/

index		**","**	**;**	**:**	**—**	**-**	**.**	**?**	**!**	**…**
	0.95	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
","	0.41	0.59	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
;	0.80	0.09	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00
:	0.86	0.05	0.00	0.05	0.04	0.00	0.00	0.00	0.00	0.00
—	0.87	0.05	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00
-	0.80	0.00	0.00	0.00	0.03	0.17	0.00	0.00	0.00	0.00
.	0.16	0.00	0.00	0.00	0.00	0.00	0.84	0.00	0.00	0.00
?	0.80	0.00	0.00	0.00	0.00	0.00	0.04	0.12	0.04	0.00
!	0.80	0.00	0.00	0.00	0.00	0.00	0.10	0.10	0.00	0.00
…	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20

Where row is the original mark, column is the new mark, number is the transformation probability.

English	Of course, the past cannot be changed, one can only observe and gently shrug.
corr	Звісно, минуле не можна змінити, тільки спостерігати і немічно розводити руками.
incorr	Звісно минуле не можна змінити тільки спостерігати і немічно розводити руками.

Table 1: Punctuation marks highlighted in red skipped in the incorr sentence

4.3 Grammar

To agree the gender of an adjective, verb, or pronoun with a corresponding head of the phrase, we develop an algorithm based on (Moskalevskyi) morphosyntactic parsing. For each correct sentence, the following conditions must be met:

$$
\begin{gathered}
\forall w, (w \in sent \wedge Ps(w) \\
\in \{PRON, NOUN, PROPN, ABBR\} \wedge w \in \Upsilon \\
\wedge (\exists a (a \in \mathbf{N}) : HL(w) = a) : \forall w_2 (w_2 \neq w \\
\wedge w_2 \in \{ADJ, VERB, PRON\} \\
\wedge HL(w_2) = HL(w) \\
\wedge R(w_2) = R(w) \\
\wedge N(w_2) = N(w) \wedge V(w_2) = V(w)))
\end{gathered}
$$

Where:

- $\forall w$ means "for all w"
- $w \in sent$ means word "w is in a sentence"
- $Ps(w)$ is the part of speech of w
- Υ is a set of possible heads for the sentence
- $HL(w)$ is a function that determines the level relative to the head of the sentence (determined according to the rules of the Mova Institute: predicate $>$ subj $>$ nsubj $>$ obj $>$ iobj $>$ obl $>$ advmod $>$ csubj $>$ xcomp $>$ ccomp $>$ advcl)
- $R(w)$ is the gender of w, $N(w)$ is the number of w and $V(w)$ is the case of w

In simpler terms, this means that for a sentence to be grammatically correct in Ukrainian, all words in the sentence that are adjectives, verbs, or pronouns must agree in gender, number, and case with the noun, abbreviation, or pronoun that is the head of the phrase.

English	Bright curtain hung on the ledge.
corr	Яскравий тюль висів на карнизі.
incorr	Яскрава тюль висіла на карнизі.

Table 2: Words highlighted in red were modified to not agree in gender with the head of the phrase

4.4 Lexics

Russism is a word that has never existed in Ukrainian and was transliterated. For treating that object, we (0) check if the word doesn't exist in Ukrainian vocabulary Υ (I) observe letter patterns in transliterating russisms into Russian, (II) generate each possible way of transliterating a russism in Russian, (III) filter generated combinations through a Russian vocabulary Ω. (IV) Translate back.
More formally, let the Russian-Ukrainian dictionary be a Map $\mathcal{D}(w) = u$. Let $\mathcal{T}(r)$ be all possible Russian transliterations of the russism word r.
Therefore, a Ukrainian correspondence u of russism r may be:

$$\forall r \notin \Upsilon \exists u \in \{\mathcal{D}(i) : i \in \Omega \cap \mathcal{T}(r)\}$$

$\mathcal{T}(r)$ is a closed-form algorithm. We identify the most common russification patterns of the most commonly used words that suffer from the substitution of russisms in Ukrainian. Based on dictionary (Tyhyj, 2009), we generate a set of rules that allow for a double conversion between the russism and the correct word.

To generate datasets, we use a set of correct Ukrainian sentences, which are then translated into Russian with a certain probability. After the translation, we replace the Ukrainian word with the most commonly used Russian loanwords.

By using a probabilistic approach to the translation and substitution with a russism, we are able to generate datasets that accurately reflect the current state of Ukrainian language usage. This approach can be extended to other languages and can be used to develop strategies for improving language usage and reducing reliance on loanwords.

English	The owner of the house where storks nested, always was considered a respected man.
corr	Власник будинку, де гніздилися лелеки, завжди вважався шанованим чоловіком.
incorr	Владєлєц будинку, де гніздилися аїсти, всєгда вважався шановним чоловіком.

Table 3: Words highlighted in red have been russified

4.5 Fluency

We developed 2 modules to generate errors in style and flow. In developing our errors, we relied on analyzing human-annotated data and rules of good writing for Ukrainian language. One module takes in a sentence, and identifies numbers written in words, after which it lemmatizes them and uses a custom dictionary to convert them to symbolic numbers. This error was based on occurence of similar errors in UA-GEC.

corr	I saw two of my friends today at the meeting.
incorr	I saw 2 of my friends today at the meeting.

Table 4: Words highlighted in red is number error

Another module performs word inversions at random. The two words must be at most two words apart. While Ukrainian does not have rigid word order, sentences written outside of dominant word order can seem odd to the reader and are annotated by the creators of UA-GEC corpora as stylistic error. As such, we created an error to generate word inversion to address that.

corr	I saw two of my friends today at the meeting.
incorr	I saw two of friends my today at the meeting.

Table 5: Words highlighted in red have been rearranged

4.6 Round Translation

The final technique that we investigated was round translation, which is known to be effective for low-resource monolingual datasets (Ahmadnia and Dorr, 2019). We would start by tokenizing and translating a sentence from Ukrainian to Russian through the Marian UK-RU (Tiedemann) encoder and transformer. Then, the sentence would be translated back, but using the UK-RU (instead of RU-UK) tokenizer and the correct transformer. In such a way, errorful sentences would be obtained through the usage of an incorrect tokenizer, yielding sentences resembling a mix of Russian and Ukrainian, also known as surzhyk.

English	In the document, Britain confirms Ukraine's right to reach its own security agreements, including with future NATO membership.
corr	Крім того, у документі Британія підтверджує право України досягати власних домовленостей щодо безпеки, включо з майбутнім членством в НАТО.
incorr	Кроме того, у документі британци підтвердять право України доносити свої суворі угоди про безносності, в тому числі і будучому членство НАТО.

Table 6: Words highlighted in red have been modified with translation

5 Results

We train and evaluate more than 80 models to compare how well different model types, in conjunction with different synthetically generated data, perform for Ukrainian grammar correction. A full list of all models can be found in the appendix. We will provide the most important findings in this section. All models are trained using Google Colab GPUs.

We evaluate all our models on the UA-GEC development set (Syvokon and Nahorna, 2021) set using the M_2 scorer provided by the UNLP shared task.

5.1 Model comparison

In Table 7 we compare the performance of select models that we train. We train 10 different sequence tagging models on different combination of synthetic data and UA-GEC. We observe that models of this type do not benefit from synthetic data, and the baseline model trained on UA-GEC is the best model of this type by $F_{0.5}$ score. For comparison we also provide the seq2tag model trained on most data, which achieves highest recall of all seq2tag models.

The rule based models which we expected to augment neural-based models achieve very high rate of false positives and therefore are not appropriate to be used as additional layer of correction. In addition to that, we find that the true positives of rule-based models occur in the areas in which neural models already perform well (spelling and punctuation).

The NMT-based models performed better than both the rule-based models and seq2tag models on the evaluation test. The precision of those models especially is noticeable higher than the seq2tag models we train. We also find that models that use

Type	TP	FP	FN	Total P	Total R	Total F0.5
NMT (baseline)	685	302	1068	0.694	0.391	0.601
NMT (best)	691	241	1047	0.741	0.398	0.632
seq2tag (baseline)	399	753	953	0.346	0.295	0.335
seq2tag (most data)	461	1324	901	0.258	0.339	0.271
rule-based	104	1064	1194	0.089	0.080	0.087

Table 7: Comparison of precision, recall, and $F_{0.5}$ score between different models. The baseline is the UA-GEC dataset; $best$ is the best dataset we used.

a lot of synthetic data or do not use the core UA-GEC train corpus perform significantly worse than the baseline trained on UA-GEC only. This lead us to experiment with adding small amount (under 20k) of synthetically generated sentences to the core UA-GEC dataset to augment our model. Our best model is trained on such an augmented UA-GEC dataset. We discuss the impact of including different synthetic data in the next secion,.

5.2 Synthetic data comparison

To evaluate the impact of our synthetically generated data on model performance, we evaluate the results of the models trained with the UA-GEC train dataset and with several thousands of synthetically generated sentences of each error type mixed in. This allows us to determine if our synthetic data helped augment the performance in select target areas. The results for each category are presented below in Tables 8-12.

Model	Category TP	Precision	Recall	F0.5
ua-gec (baseline)	435	0.694	0.391	0.601
punct-assist (best)	488	0.742	0.375	0.620
ua-gec (baseline)	39	0.694	0.391	0.601
grammar-assist (best)	44	0.703	0.399	0.610
ua-gec (baseline)	153	0.694	0.391	0.601
spelling-assist (best)	83	0.738	0.317	0.583
ua-gec (baseline)	57	0.694	0.391	0.601
lexics-assist (best)	47	0.719	0.383	0.612
ua-gec (baseline)	57	0.694	0.391	0.601
fluency-assist (best)	41	0.712	0.365	0.599
ua-gec (baseline)	685	0.694	0.391	0.601
translation (best)	697	0.703	0.399	0.610

Table 8: Punct-assisted model is the model that was trained on synthetically generated punctuation errors. It achieves a higher target TP rate and overall $F_{0.5}$ score.

Model	Grammar TP	Precision	Recall	F0.5
ua-gec (baseline)	39	0.694	0.391	0.601
grammar-assist (best)	44	0.703	0.399	0.610

Table 9: Grammar-assisted model is the model that was trained on synthetically generated grammar errors. It achieves a higher target TP rate and overall $F_{0.5}$ score.

Model	Spelling TP	Precision	Recall	F0.5
ua-gec (baseline)	153	0.694	0.391	0.601
spelling-assist (best)	83	0.738	0.317	0.583

Table 10: Spelling-assisted model is the model that was trained on synthetically generated spelling errors. It achieves a lower TP rate and overall $F_{0.5}$ score.

Model	Lexics TP	Precision	Recall	F0.5
ua-gec (baseline)	57	0.694	0.391	0.601
lexics-assist (best)	47	0.719	0.383	0.612

Table 11: Lexics-assisted model is the model that was trained on synthetically generated lexical errors. It achieves a lower TP rate and overall $F_{0.5}$ score.

Model	Fluency TP	Precision	Recall	F0.5
ua-gec (baseline)	57	0.694	0.391	0.601
fluency-assist (best)	41	0.712	0.365	0.599

Table 12: Fluency-assisted model is the model that was trained on synthetically generated fluency errors (numerals-to-words, word order). It achieves a lower TP rate and overall $F_{0.5}$ score.

Model	Total TP	Precision	Recall	F0.5
ua-gec (baseline)	685	0.694	0.391	0.601
translation (best)	697	0.703	0.399	0.610

Table 13: Translation best model is the model that was trained on a synthetically generated back-translation (Ukrainian-Russian-Ukrainian). It achieves a higher TP rate and overall $F_{0.5}$ score.

We have determined that punctuation, grammar, and round-translation successfully augment the target category, and all but spelling and fluency successfully improve overall $F_{0.5}$ score. We believe that lexics and fluency errors did not improve the performance of the model due to intrinsic complexities of correction for those categories. The reason adding spelling failed to improve performance is a direction for future research.

5.3 The best model

When evaluating models augmented with synthetic data we noticed that the resulting F_{05} score some-

Model	TP	FP	FN	P	R	F0.5
ua-gec	685	302	1068	0.694	0.391	0.601
Dilute 20k	639	248	1074	0.720	0.373	0.607
Dilute 100k	544	184	1144	0.747	0.322	0.591

Table 14: Comparison of diluted model. Diluting increases precision at the cost of recall. Without extra data, the cost to recall is too high to justify

times went up compared to the baseline when the true positive rate for the target category actually went down. This increase in $F_0.5$ score can be accounted for by the increase in accuracy: adding that extra data did not make the model better at new type of errors, but made model already better at making fewer false positives.

Based on our previous research, it is found that the conventional approach of increasing the dataset size does not necessarily improve the model performance. Consequently, a decision was made to selectively generate mixed data to create a more diverse and representative dataset that could improve the performance of the model on a wider range of inputs.

For this study, the initial dataset UA-GEC (25k sentences) was selected, which contained a wide variety of errors. The objective was to achieve maximum accuracy in a specific category by artificially adding 10k sentences with only punctuation errors. As a result, the true positive (TP) rate increased by 53 points. Further tests were conducted by adding other inclusions, which resulted in an oversaturation of the model with only erroneous sentences. Therefore, around 5k absolutely correct sentences were added, resulting in a rapid decrease in the false negative (FN) rate.

The same process was repeated for other possible categories, but not as significant progress was made as before. This led to the conclusion that the type of error generation used in this study may be specific or simply not comparable to the test dataset.

It is important to note that mixing data in NLP can introduce new challenges, such as domain adaptation or language transfer issues. Therefore, it is essential to carefully evaluate the model's performance on a separate validation set to ensure that the mixing of data does not negatively impact its generalization ability.

6 Conclusions

We have investigated most of the state-of-the-art GEC approaches in the English language and tried to appropriate them for the Ukrainian language. We found that the most efficient GEC system can be developed using the NMT approach, however, seq2tag has a lot of room for research. Our best model gets the 0.632 $F_{0.5}$ score on the UA-GEC dataset, establishing the state-of-the-art benchmark.

Moreover, the results suggest that adding a mix of punctuation errors, russism errors, and clean data to the UA-GEC training data achieves the best results. Overall, we found that data quality is much more important than the amount of raw data. Therefore, we suggest that human-annotated GEC data is the most promising direction for future research.

Limitations

To the best of our knowledge, our paper is the first one outlining the application of the modern GEC techniques to the Ukrainian language, so it is bound by a lot of limitations.

1. We did not use neither Wiki Edits nor Lang8 datasets. Our initial overview observed that both of them included a lot of artifacts and grammatical mistakes in the “correct” options, so we concluded that cleaning up those datasets would take a lot of time and resources. Hence, both of them lay outside the scope of this paper.

2. Due to technical limitations of the resources we had, we did not have an option to test all available multi-language transformers. We know for a fact that there are multiple transformers, such as T5 and ELECTRA that can be adapted for both NMT and seq2tag architectures for the Ukrainian language. However, testing all of them was not technically feasible, therefore, this paper does not include that.

Model	TP	FP	FN	Precision	Recall	F0.5
ua-gec	685	302	1068	0.694	0.391	0.601
ua-gec + punct10k + dilute5k	644	221	1069	0.745	0.376	0.623
ua-gec + punct10k + dilule3.5k + lexics5k	691	241	1047	0.741	0.398	0.632

Table 15: Comparison of best NMT model data combinations

3. We found that the amount of data indeed scales well for the seq2tag architecture, however, the amount of truly error-free "clean" sentences was capped at 1 million. At the same time, most research for the English language used much more data, such as 9 million for GECToR, or even more for the mBART training. Therefore, the question of comparing seq2tag and NMT in the context of the Ukrainian language remains open.

4. For seq2tag, we realize that the total number of possible G-transformations is much higher than we have used, and that, despite covering most of the grammatical and punctuation errors, we did not cover everything. Therefore, there is still ample room for research of G-transformations.

5. Finally, this paper did not study back-translation models. All the ones that were used in the English language were trained on vast amounts of human-annotated data, while we have only UA-GEC. However, the usefulness of these models might be shown by future studies.

Acknowledgements

This work enjoyed selfless support from Anastasiia Teriokhina, Artur Zhdan, Maksym Matviievsky, Mykhailo Trushch, Nikita Masych, Tanya Melnyk, Taras Yaitskyi, and Vladyslav Verteletskyi. Thank you Veronika Kitsul for editing.

References

Benyamin Ahmadnia and Bonnie J. Dorr. 2019. Bilingual low-resource neural machine translation with round-tripping: The case of persian-spanish. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2019, Varna, Bulgaria, September 2-4, 2019*, pages 18–24. INCOMA Ltd.

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

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, BEA@ACL 2019, Florence, Italy, August 2, 2019*, pages 52–75. Association for Computational Linguistics.

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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 793–805. Association for Computational Linguistics.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2022. Grammatical error correction: A survey of the state of the art. *CoRR*, abs/2211.05166.

Solomija N. Buk and Andrij A. Rovenchak. 2003. The rank-frequency analysis for the functional style corpora in the ukrainian language. *CoRR*, cs.CL/0311033.

Manaal Faruqui, Ellie Pavlick, Ian Tenney, and Dipanjan Das. 2018. Wikiatomicedits: A multilingual corpus of wikipedia edits for modeling language and discourse.

Simon Flachs, Felix Stahlberg, and Shankar Kumar. 2021. Data strategies for low-resource grammatical error correction. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 117–122, Online. Association for Computational Linguistics.

Rayid Ghani, Rosie Jones, and Dunja Mladenić. 2001. Mining the web to create minority language corpora. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, page 279–286, New York, NY, USA. Association for Computing Machinery.

Jared Lichtarge, Christopher Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. *CoRR*, abs/1904.05780.

Bohdan Moskalevskyi. Mova institute.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. 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, BEA@ACL 2020, Online, July 10, 2020*, pages 163–170. Association for Computational Linguistics.

S. D. Pogorilyy and A. A. Kramov. 2020. Method of noun phrase detection in ukrainian texts. *CoRR*, abs/2010.11548.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 1715–1725.

Serge Sharoff. 2006. Creating general-purpose corpora using automated search engine queries. *WaCky*.

Robyn Speer. 2019. ftfy.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models.

Oleksiy Syvokon and Olena Nahorna. 2021. UA-GEC: grammatical error correction and fluency corpus for the ukrainian language. *CoRR*, abs/2103.16997.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *CoRR*, abs/2008.00401.

Jörg Tiedemann. Marianmt.

Oleksa Tyhyj. 2009. *Slovnyk movnyh pokruchiv*.

Zheng Yuan. 2017. Grammatical error correction in non-native English. Technical Report UCAM-CL-TR-904, University of Cambridge, Computer Laboratory.

A Appendix

Category	**TP**	**FP**	**FN**	**P**	**R**	**F0.5**
Total	691	241	1047	0.741	0.398	0.632
F/Calque	5	0	72	1.000	0.065	0.258
F/Collocation	4	0	20	1.000	0.167	0.500
F/PoorFlow	10	0	129	1.000	0.072	0.279
F/Repetition	3	0	31	1.000	0.088	0.326
F/Style	18	0	125	1.000	0.126	0.419
G/Aspect	0	0	2	1.000	0.000	0.000
G/Case	13	0	73	1.000	0.151	0.471
G/Comparison	0	0	3	1.000	0.000	0.000
G/Conjunction	4	0	11	1.000	0.267	0.645
G/Gender	3	0	13	1.000	0.188	0.536
G/Number	2	0	17	1.000	0.105	0.370
G/Other	2	0	3	1.000	0.400	0.769
G/PartVoice	0	0	4	1.000	0.000	0.000
G/Participle	0	0	1	1.000	0.000	0.000
G/Particle	1	0	2	1.000	0.333	0.714
G/Prep	3	0	21	1.000	0.125	0.417
G/Tense	0	0	12	1.000	0.000	0.000
G/UngrammaticalStructure	2	0	42	1.000	0.046	0.192
G/VerbAForm	7	0	4	1.000	0.636	0.897
G/VerbVoice	0	0	9	1.000	0.000	0.000
M:NOUN	0	1	0	0.000	1.000	0.000
M:OTHER	0	2	0	0.000	1.000	0.000
M:PUNCT	0	46	0	0.000	1.000	0.000
Other	3	0	8	1.000	0.273	0.652
Punctuation	488	0	172	1.000	0.739	0.934
R:DET	0	1	0	0.000	1.000	0.000
R:NOUN	0	43	0	0.000	1.000	0.000
R:ORTH	0	8	0	0.000	1.000	0.000
R:OTHER	0	14	0	0.000	1.000	0.000
R:PUNCT	0	39	0	0.000	1.000	0.000
R:SPELL	0	46	0	0.000	1.000	0.000
R:VERB	0	1	0	0.000	1.000	0.000
R:WO	0	1	0	0.000	1.000	0.000
Spelling	123	0	273	1.000	0.311	0.693
U:NOUN	0	19	0	0.000	1.000	0.000
U:OTHER	0	1	0	0.000	1.000	0.000
U:PUNCT	0	19	0	0.000	1.000	0.000

Table 16: Full breakdown of the $F_{0.5}$ score of our best model

Model	**ua gec**	**grammar**	**punct**	**spelling**	**lecxics**	**rus**	**dilute**	**backtranslation**	**RT**	**data**	**F0.5**
5k punct + 10k ua_gec	10k		5k							5k punct + 10k ua_gec	0.5061
5k-rt-punct	full		5k						5k	UA-GEC + rt5k + punct5k	0.6101
5k-rt-punct-3e	full		5k						5k	UA-GEC + 5k backtranslated + 5k punct	0.6117
5k-rt-punct-diluted10k	full		5k				10k		5k	UA-GEC + 5k backtranslated + 5k punct + 10k clean	0.6111
press_f_gram_dilute	full	5k					5k			"ua-gec, 5k gram, 5k dilute"	0.6128
press_f_gram_typos_punct_dilute	full	5k		5k			5k			"ua-gec, 5k gram, 5k dilute, 5k punct"	0.6128
borshch_25_future_8	full					10k				russified_borshch_0.25er_10k	0.6118
borshch_40_future_7	full					10k				russified_borshch_0.4er_10k + gec	0.5793
combined-assist	full	2.5k	2.5k							ua-gec + 5k combined errorifer data from borsch-combined	0.5782
dilute-100k	full						100k			UA-GEC + 100k pure	0.5913
dilute-20k	full						20k			ua-gec + 20k pure	0.6073
future_10_punct5k_rus5k	10k		5k			5k				punct5k + russified_gec 5k + ua_gec10k	0.5828
future_10_punct5k_rus5k	full		5k			5k				gec_punct5k_rus5k	0.5828
future_12_punct5k_gec10k_punct5k_gec	full		10k							punct5k(low er) + gec10k + punct5k(er) + gec	0.6128
future_13_2ep	full		10k				5k			Punct10k + ua_gec + 5k dilute	0.6225
future_15_3ep	full		5k				3k			Punct5k + ua_gec + 3k dilute (shuffle)	0.6192
future_16_3ep	full		10k		5k	5k	5k				0.6299
future_17_3ep	full		10k		5k		3.7k				0.6321
future_9_punct5k_rus10k			5k		10k					punct5k + russified_gec10k	0.5795
future3					10k	10k				surzh10k(from 450kdataset)+ rusified_gec10k	0.3669
maksym-unlp-1	full									ua-gec	0.6107
maksym-unlp-2	full									ua-gec	0.6110
maksym-unlp-3	full									ua-gec	0.6107
maksym-unlp-5	full									ua-gec	0.6008
maksym-unlp-6	full									ua-gec	0.6125
press_f	12k	5k	5k			5k	5k			"12k UA-GEC, 5k inflection-preposition-gender, 5k rus, 5k dilute, 5k punct"	0.5914
press_f_typos	12k			5k			13k			"12k ua -gec, 13k ua-gec correct, 5k typos"	0.5832
punct-assist-1	fukk		5k							ua-gec + 5k punct errorifier with eror_prob=0.01	0.6172
punct-assist-2	full		10k							ua-gec + 10k punct errorifier with eror_prob=0.01	0.6152
punct-assist-3	full		20k							ua-gec + 20k punct errorifier with eror_prob=0.01	0.6204
rus 5k + punct_5k + ua_gec	full		5k			5k				rus 5k + punct_5k + ua_gec	0.5828
rus_45_future5	-				20k					russified_gec20k(medium_error)	0.5854
rus_60_future6	-				20k					russified_gec20k(0.6_error)	0.5767
rus_future4	-				20k					rusified_gec20k	0.5669
uagec-5k-rt	full							5k		UA-GEC + punct10k + dil4k + rus5k + rt5k mixed	0.6087
uagec-rt5k-punct10k-dilute4k-rus10k-mixed	full		10k			5k	4k		5k		0.6165
uagec-rt5k-punct10k-diluted1k-rus5k-mixed	full		10k				1k		5k	UA-GEC + punct10k + dil1k + rt5k + rus5k mixed	0.6065
uagec-rt5k-punct10k-diluted2k-rus10k-mixed	full		10k			5k	2k		5k	UA-GEC + punct10k + dil2k + rt5k + rus5k mixed	0.6187
uagec-rt5k-punct10k-rus10k-diluted1k-mixed	5k									5k	0.6226
uagec-rt5k-punct10k-rus10k-mixed	full		10k						5k	UA-GEC + punct10k + rt5k mixed	0.6097

Table 17: All models we have trained with their respective data and $F_{0.5}$ scores.