




Article

Identification and Correction of Grammatical Errors in Ukrainian Texts Based on Machine Learning Technology

Vasyl Lytvyn ¹, Petro Pukach ², Victoria Vysotska ^{1,3}, Myroslava Vovk ^{2,*} and Nataliia Kholodna ¹¹ Information Systems and Networks Department, Lviv Polytechnic National University, 12 Bandera Str., 79013 Lviv, Ukraine² Institute of Applied Mathematics and Fundamental Sciences, Lviv Polytechnic National University, 12 Bandera Str., 79013 Lviv, Ukraine³ Institute of Computer Science, Osnabrück University, 1 Friedrich-Janssen-Str., 49076 Osnabrück, Germany

* Correspondence: myroslava.i.vovk@lpnu.ua

Abstract: A machine learning model for correcting errors in Ukrainian texts has been developed. It was established that the neural network has the ability to correct simple sentences written in Ukrainian; however, the development of a full-fledged system requires the use of spell-checking using dictionaries and the checking of rules, both simple and those based on the result of parsing dependencies or other features. In order to save computing resources, a pre-trained BERT (Bidirectional Encoder Representations from Transformer) type neural network was used. Such neural networks have half as many parameters as other pre-trained models and show satisfactory results in correcting grammatical and stylistic errors. Among the ready-made neural network models, the pre-trained neural network model mT5 (a multilingual variant of T5 or Text-to-Text Transfer Transformer) showed the best performance according to the BLEU (bilingual evaluation understudy) and METEOR (metric for evaluation of translation with explicit ordering) metrics.

Keywords: NLP; text pre-processing; error correction; grammatical error correction; machine learning; deep learning; text analysis; text classification; neural network

MSC: 68Q55; 68T50; 68U15; 68U35

Citation: Lytvyn, V.; Pukach, P.; Vysotska, V.; Vovk, M.; Kholodna, N. Identification and Correction of Grammatical Errors in Ukrainian Texts Based on Machine Learning Technology. *Mathematics* **2023**, *11*, 904. <https://doi.org/10.3390/math11040904>

Academic Editor: Zhao Kang

Received: 4 January 2023

Revised: 27 January 2023

Accepted: 8 February 2023

Published: 10 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

GEC (grammatical error correction) is the goal of automatic error identification and correction in the input text. GEC is used in various fields: the correction of search queries, machine translation from one language to another language (MT, machine translation), spell checking in browsers and word processors, etc. GEC methods are divided into the following methods: rule-based; based on syntactic analysis of sentences; and based on machine learning methods, in particular deep learning.

Rule-based checking relies on a set of predefined patterns of possible errors in the text. Such rules are usually developed manually. A text is erroneous if it meets one of the rules [1]. The advantages of the approach are speed of action, interpretation of results, and possibilities of iterative development. However, the method has drawbacks: the complexity increases as different types of errors appear, and the creation of rules is resource-consuming and requires expert knowledge of linguistics. A huge number of rules are needed to cover all possible errors in the text.

Syntax-based checking performs a complete analysis of the morphology and syntax of the text. This requires a lexical database (DB), as well as morphological and syntactic analyzers (parsers). Depending on the grammar of the language, the syntactic parser sets the syntactic structure of each sentence in the form of a tree. If the full analysis was not successful, then the text is erroneous [1]. The disadvantage of the syntactic approach is

the need to develop additional rules for clarifying corrections in sentences [2]. These rules should cover all possible error options.

Machine learning models, in particular deep learning ones [1], learn to predict corrections for each word/tag that determines the action on a certain token that needs to be performed to correct the sentence (sequence labeling). Most often, recurrent neural networks (RNN) are used to process natural language texts. A large number of sentences (parallel corpora) are used to build an RNN model and to train it. However, RNN perceives tokens sequentially, which slows down the learning and prediction time and makes parallel data processing impossible. In the case of commercial applications, latency is critical. Therefore, scientific research is aimed at applying another (non-RNN) architecture of neural networks, which is called ‘transformer’.

Transformer is a deep learning model that replaces recurrence with an attention mechanism [3,4]. This mechanism enables preserving the context of the text for any position of the token in the input word sequence. This, in turn, makes it possible to parallelize processes compared to RNN, thus reducing the duration of training [3]. Transformers quickly became the dominant architecture for NLP [4], especially for tasks that involve language understanding (classification, translation, generalization of text, machine translation) and language generation. Pre-training the model allows training transformers on large open unannotated corpora. Later, they can be adjusted to specific tasks, which results in a high-quality system. The BERT model is one such transformer [5]. This model can be customized for a specific task by just adding a single layer of neurons without significant modifications to the internal architecture. BERT is a neural network trained to solve two tasks: predicting a certain word in a sentence and determining whether the next sentence is a logical continuation of the previous one.

It should be added that for the English language, a significant increase in the accuracy of English grammar correction has been achieved due to the construction of GEC systems. Unfortunately, almost no research has been conducted on the Ukrainian language. Ukrainian belongs to morphologically complex languages. A large amount of parallel or manually labeled data is required to build a good machine-learning model for correcting grammatical/stylistic errors in texts of morphologically complex languages. Currently, because of the war, a large amount of disinformation and fakes appear in the information space of Ukraine, usually written by non-native speakers. One of the methods of identifying potentially false information is the presence of grammatical and stylistic errors in such news, which are usually present when the text is automatically translated from another language or when texts are written by non-native speakers without correction by professional editors. Therefore, research towards determining the unique features of the identification of grammatical errors in the Ukrainian language is an urgent task at present. According to research results, to obtain the best GEC accuracy using a neural network, you need to use pre-trained deep learning models that support the Ukrainian language.

The aim of the research was to develop directions for building a GEC system for texts written in Ukrainian using deep learning methods (transformers). In order to achieve this aim, the following tasks must be solved:

- Research of the Ukrainian text corpora;
- Comparison of state-of-the-art methods for GEC;
- Research of using neural networks with different architectures;
- Choice of the most optimum model of a GEC system for texts written in Ukrainian.

2. Related Works

GEC methods that are based on rules, syntactic analysis, or statistical modeling are described in studies for various languages: Danish [6], Greek [7], Latvian [8], Slavic [9,10], Punjabi [11], Filipino [12], Arabic [13], English [14], and Ukrainian [15].

GEC systems for the English language are constantly developing in the direction of using deep learning methods. Two applications are known: Grammarly and LanguageTool. Grammarly is a commercial online platform from Grammarly Inc. that not only checks

and corrects grammatical errors but also offers recommendations for clarity (conciseness and lucidity), catchiness (vocabulary and variety), and message tone (formality, politeness, and confidence) [16,17]. The platform was included in The Time's rating 'Time100 Most Influential Companies of 2022' and FastCompany's rating 'The 10 most innovative companies in artificial intelligence of 2022', as well as in Forbes' 'Cloud 100' list and The Software Report's 'Top 100 Software Companies' [18]. Currently, Grammarly supports only the English language and its dialects: American, British, Canadian, and Australian [19].

LanguageTool is an open-source GEC program that uses rule-based validation. In total, the system supports more than 30 languages, including Ukrainian [20]. LanguageTool has been developing since 2003 and currently has 5415 rules for English and 1022 for Ukrainian [21]. Every day, users check more than 20 million texts using this platform [22]. To check the correctness of written Ukrainian texts, rules of the following categories are available: barbarisms (for example, приймати участь—брати), capital letters, grammar, logical errors (for example, wrong date), spelling, design, punctuation, style, typography [21]. The disadvantage of LanguageTool for checking Ukrainian texts is the small number of rules that cannot cover all possible grammatical errors. Therefore, research on the use of deep learning methods to build GEC systems for Ukrainian texts is promising.

The researchers Oleksiy Syvokon and Olena Nahorna have presented a set of data that is a text corpus [15] that is professionally labeled for GEC and free editing in Ukrainian. The researchers have collected texts with errors (20,715 sentences—328,779 tokens) by various authors, including native speakers. The data cover a wide range of styles, from chats and essays to formal writing. Professional proofreaders corrected and annotated the corpus for errors related to fluency, grammar, punctuation, and spelling. Researchers [1,15] believe that this corpus can be used for the development and evaluation of GEC systems in the Ukrainian language and the study of morphologically rich languages. The biggest problem is the need for high-quality training corpora containing a large number of training examples [1].

The 'pymorphy2' morphological parser for the Ukrainian language was developed in [23]. 'pymorphy2' analyzes the part of speech, number, case, tense, the lemma and stem of a given word. The morphological parser is based on OpenCorpora dictionaries converted to XML format. Users can also add their own words and rules. This enables performing morphological analysis of texts of a certain subject area (SA) without changing the source code of 'pymorphy2' and adapting 'pymorphy2' to work with other languages.

In [24], researchers developed a system for text pre-processing (TPP) for morphological and syntactic analysis of Ukrainian texts. To tokenize, split into sentences, and search for email addresses, the researchers used the NLTK library of the Python programming language and regular expressions to remove stop words and search for named entities. The 'pymorphy2' library, which supports the Ukrainian language, was used for morphological analysis of words, their lemmatization, or stemming. In addition, the researchers implemented the graphical interface of the application using the PyQt library.

At present, many scientific schools and specialists continue research into Ukrainian texts based on NLP methods, including the studies covered in [23–30]. However, they are mostly focused on the specificities of processing the Ukrainian language but not on applying machine learning to optimize the solution of NLP problems. In this work, the first step towards error correction in Ukrainian texts using machine learning methods was made, which includes pre-processing of text content, selection and generation of features, and the selected algorithms—all this in conditions of a small annotated data corpora.

The level of correct noun identification in [31] for Polish text is 87% nominative, 65% genitive, 77% dative, 84% locative, 79% instrumental, and 66% accusative. Experimental results verify that based on a morphological analyzer using a neural network in [32] level of error identification for Polish text comes up to 93.3–99.9%. In [33], the author demonstrated the obtained accuracy of up to 28% to identify errors among the words being in common use and 7% for the unknown Polish words.

To analyze sentences in Byelorussian [34], authors used models on trees of universal dependencies of two close Slavic languages (Ukrainian and Russian). Due to the obtained annotations in Byelorussian identification level of grammatical errors is up to 83.4%.

In [10], there is a proposed approach of grammatical correctness checking of Russian-language texts based on ‘minimal supervision’. The sense of this approach is to apply small annotated data set with the addition of artificial errors to news corpora, literature, etc. (18 million words total). Research has developed classifiers for some common types of grammatical errors: prepositions, noun case, verb form, and agreements with the verb (division by number and gender). Additionally, the authors used the neural MT method to the text, including errors, and noticed that its accuracy is too low because of insufficient incoming data amount. The proposed ‘minimal supervision’ method also increases classifiers’ accuracy and MT quality. It was experimentally noticed that the MT method demonstrates unsatisfactory qualitative metrics (F-score = 10.6) while using data sets of the corresponding texts containing approximately 200 thousand manually annotated and corrected words. This MT system is significantly surpassed by the ML approach proposed in [10] (a variation of the study with a teacher using unmarked training data; usually, there is a small marked data set and a large unmarked data set). In [35], there is an evolved system of grammatical error correction based on the approach that uses sequence annotation via a classical machine learning algorithm (namely, the Naive Bayes Classifier). Five models were used corresponding to present in corpora grammatical error types.

There is no base established metrics to estimate GEC quality, respectively, to [36–42]. Therefore, to estimate such systems in general, MT-quality metrics are used. However, some disadvantages also are present. BLEU (bilingual evaluation understudy) [43] is an algorithm to estimate MT quality. Estimations are calculated for separated translated segments (usually sentences), comparing them with high-quality translation sets. These results are averaged over all corpora to obtain the general estimation of translation quality. BLEU estimation is always a number from 0 to 1. This number means how suchlike is text-candidate to standard text. Numbers close to 1 define more similar texts. BLEU value is calculated on common sets of N-gram in the initial sentence and its translation.

METEOR (metric for evaluation of translation with explicit ordering) [44] is one estimation MT-quality metric. It uses N-gram and is oriented on statistics and exact estimation of the initial text. Unlike BLUE metrics, this metric uses functions of comparison between synonyms and exact word correspondence. It was developed to solve BLUE problems and also to correlate better with expert estimations on phrases or sentences level.

3. Materials and Methods

The morphological complexity of the Ukrainian language necessitates the combination of different GEC methods. Training a neural network ‘from scratch’ using randomly initialized weights requires large parallel corpora. At the time of writing, the only Ukrainian-language data set contains just about a thousand texts, which is critically small for training a model for processing morphologically rich languages, such as Ukrainian.

Therefore, it is proposed to take an already-trained model as a basis. There are several pre-configured models that support the Ukrainian language: RoBERTa from YouScan [36] and MT-models (from Google—MT5 [37], from Facebook—M2M100 [38], and mBART-50 [39]). There are no basic or set metrics for assessing GEC quality [40–42]. MT-quality metrics (BLEU (bilingual evaluation understudy) [43]; METEOR (Metric for Evaluation of Translation with Explicit ORdering [44])) are generally used to evaluate such systems, but they also have certain drawbacks [37–45].

The step sequence of interaction with the checking system of text grammatical correctness is presented by two diagrams of the sequence corresponding to individual user applications and for NLP-application using text correction as one of the components in their own pipeline (pipeline Figure 1).

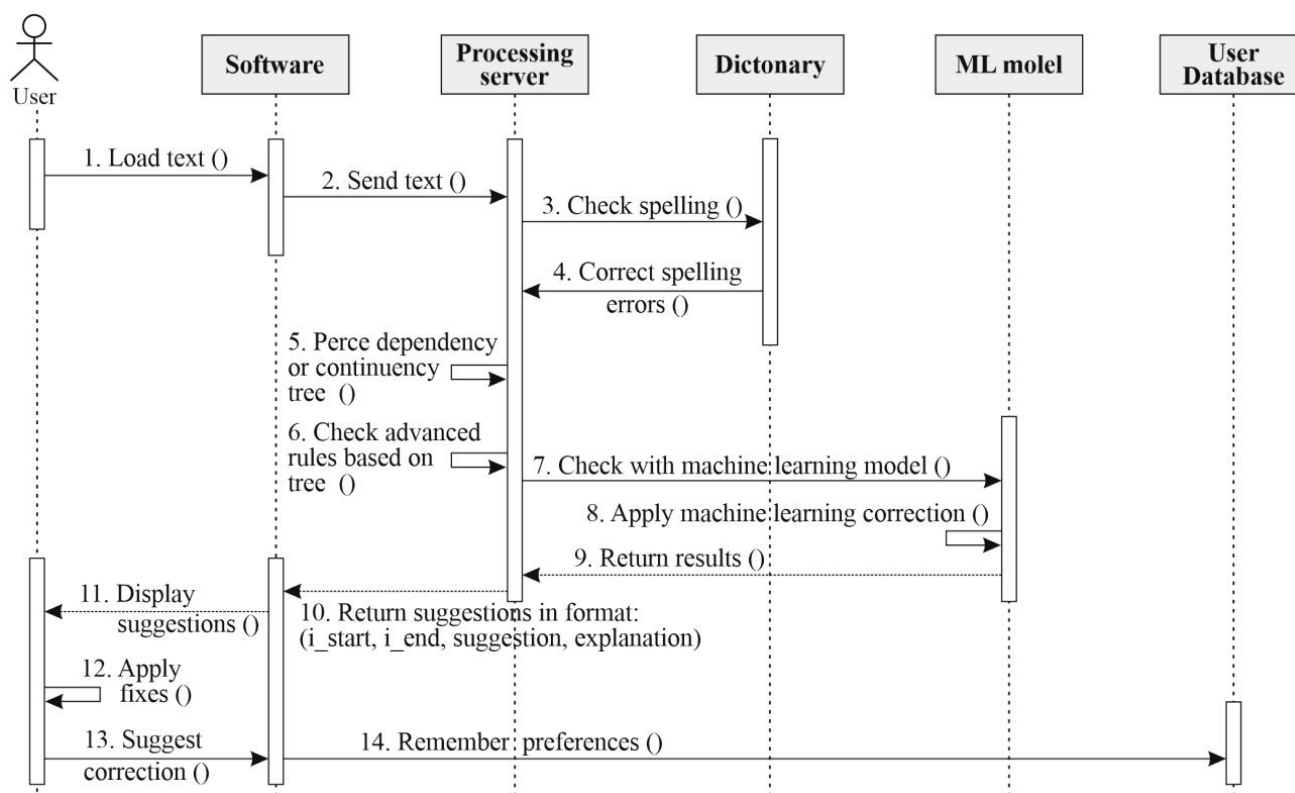


Figure 1. Sequence diagram for the first regime of operating system.

Figure 1 presents a sequence diagram for the user's text checking. There are registration and authorization steps, also creating and saving documents. This database is used to save preferences/samples of correctness for further personification of proposed by system correctness. At first, the user chooses the version of data loading by text insert, file opening, or creating and filling a new document.

Personal installed software or available in online format sends data to the server for analysis. Only one possible approach to constructing a grammatical correctness system is demonstrated on the diagram. It combines checking spelling, rules, syntactic parsers, and using machine learning algorithms. After checking spelling with the necessary language dictionary system needs to check basic rules without syntactic parsers (for example, rotation «y-B», the spelling of «пів-хатів», and use of apostrophes).

Using of rules checking and syntactic parsers are explained by two factors:

- Absence of sufficiently large annotated /parallel Ukrainian language corpora to study entirely automatic models based on deeper study algorithms;
- The insufficient capability of the language models of artificial intelligence to generalize rules.

While all possible spelling rules are not realized in the system, including those based on the syntactic parser, the additional use of machine learning methods can be considered to predict tags defining necessary action over tokens or to 'translate' text with errors into the correct version. Then, the software visualizes and applies the proposed corrections and saves the document in the database if needed.

The second regime of system application is to analyze the grammatical correctness of the obtained via API requests (Figure 2). The following diagram shows the steps of sending, obtaining, and processing requests from the outside NLP application.

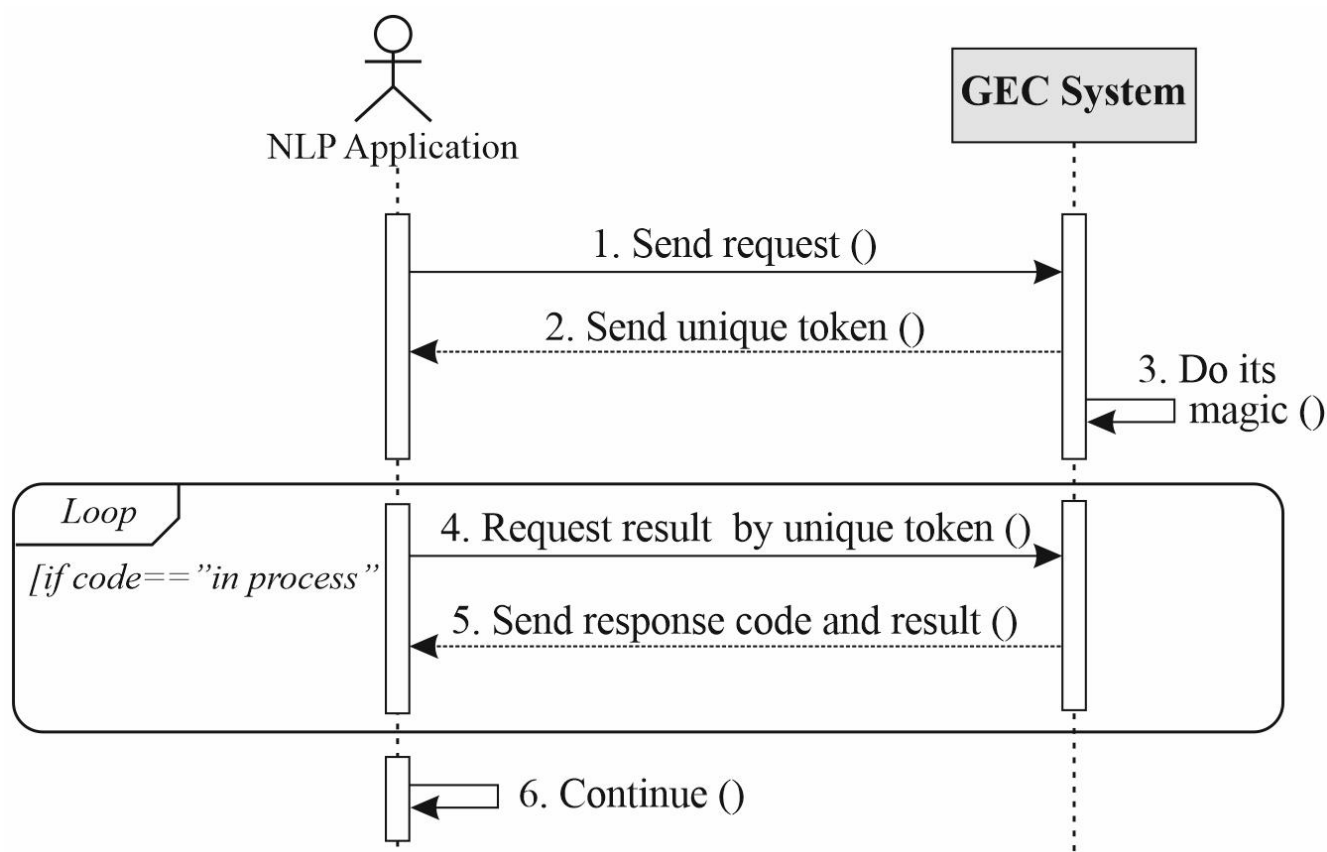


Figure 2. Sequence diagram for the second regime of the operating system.

GEC System defines the total system of grammatical correctness together with all components, including graphic interface, server for processing, and previously studied machine learning models.

GEC System proposes a unique token for each request; due to this token, one can receive results when request processing ends. NLP application must send a new request for results obtaining with some time interval taking into account system load. NLP application continues to work after obtaining corrected texts.

The diagram of the processing activity (Figure 3) shows one of the possible approaches to developing a system of automatic grammatical correctness.

After checking and spelling corrections, it is important to check the main grammar rules. Ukrainian examples: rotation «y-b», the spelling of «пів-напів», use of an apostrophe, etc.). Depending on the availability of syntactic parsers, words groups, parts of speech, etc., one can use checking of the additional rules basing on the obtained results.

If interactive development of the system is theoretically possible and accepted rules explain all probable errors of some language, then using methods and algorithms is not necessary.

When checking the additional rules is not sufficient (especially in morphologically rich languages) to correct remaining errors, it is possible to apply machine learning methods. When there is a lack of training databases, or they are unavailable to obtain parallel/annotated corpora, one can apply methods of augmentation and data generation.

The choice of the corresponding approaches and models evidently depends on whether databases are annotated or not and also on the presence of parallel texts. In cases of sufficiently large and qualitative non-annotated corpora, it is also possible to develop a language model based on statistical methods.

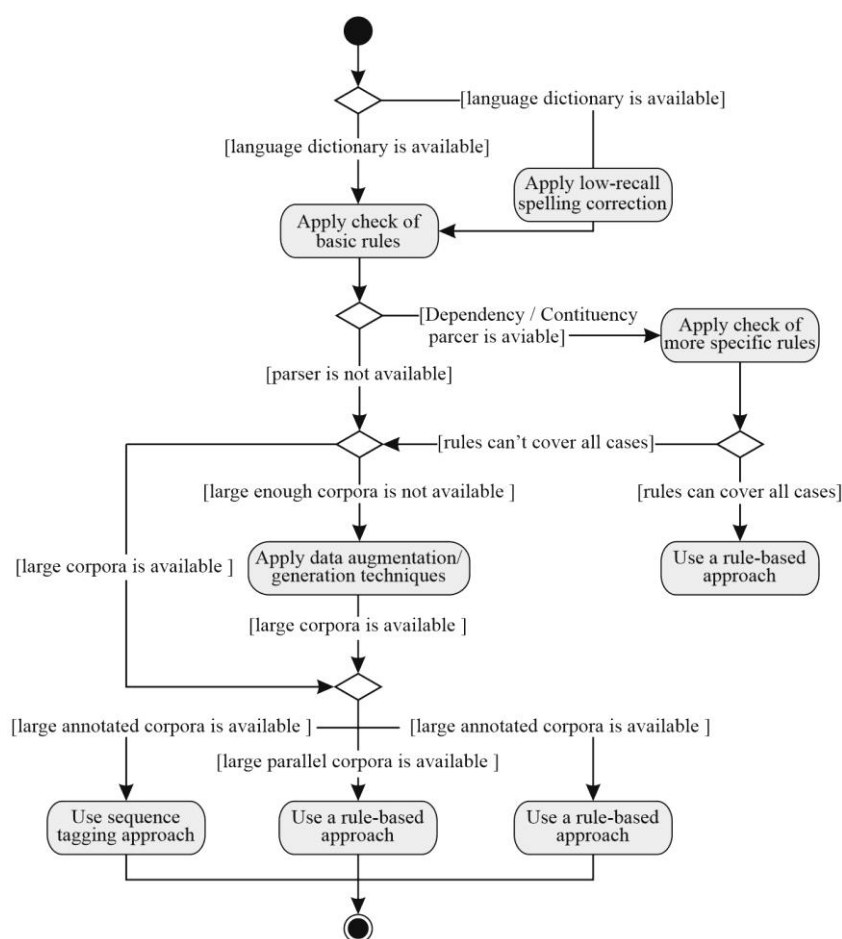


Figure 3. Diagram of processing activity of automatic grammatical correctness system.

4. Results

4.1. Setting the RoBERTa Pre-Trained MT Model

The RoBERTa pre-trained neural network can be considered an improved analog of BERT [40]. Its architecture is similar to that of BERT. The main difference is selecting hyperparameters during pre-training, increasing the volume of the training corpus, and applying dynamic masking of tokens for predicting a word in a sentence. The word that must be predicted in a given sentence changes with each epoch. However, there is no prediction of whether the next sentence is a logical continuation of the previous one.

Encoder-decoder neural networks, such as BERT, generate context vector representations of words in a sentence of fixed length regardless of the length of the input message. They generate tokens sequentially. Each subsequent token depends on the previous ones. Application of the previously trained model ‘Ukrainian Roberta’ is possible by analogy with the approach proposed in [41]. When initializing BERT as a decoder, a cross-attention layer with randomly generated weights must be added to predict the next word based on the context vector embedding of input tokens. LM Head (a layer or layers of fully connected neurons, where the number of output neurons corresponds to the number of tokens in the dictionary) is used to predict the distribution of probabilities of using the next word. LM Head weights are initialized with the BERT W_{emb} vector embedding layer weights. Moreover, the bidirectional mechanism of attention in the BERT model must be replaced by a unidirectional one to generate a token that would depend only on previous tokens [40,45]. The dictionary of tokens that corresponds to the previously trained model ‘Ukrainian Roberta’ contains 52 thousand elements, where the first entries are special tokens, punctuation marks, and most common words (Figure 4). Since token dictionaries are unique for each pre-trained model, TPP also has its own peculiarities.

After the tokenization of texts and sentences in the training sample, the distribution of the number of tokens is the following (Figure 5). The maximum sentence length is 100 tokens, and longer sentences are truncated.

```
sorted(tokenizer.vocab.items(), key=lambda x: x[1])

[('<s>', 0),
 ('<pad>', 1),
 ('</s>', 2),
 ('<unk>', 3),
 ('<mask>', 4),
 ('!', 5),
 ('"', 6),
 ('#', 7),
 ('$', 8),
 ('%', 9),
 ('&', 10),
 ('"', 11),
 ('(', 12),
 (')', 13),
 ('*', 14),
 ('+', 15),
```

Figure 4. Token dictionary of ‘Ukrainian Roberta’.

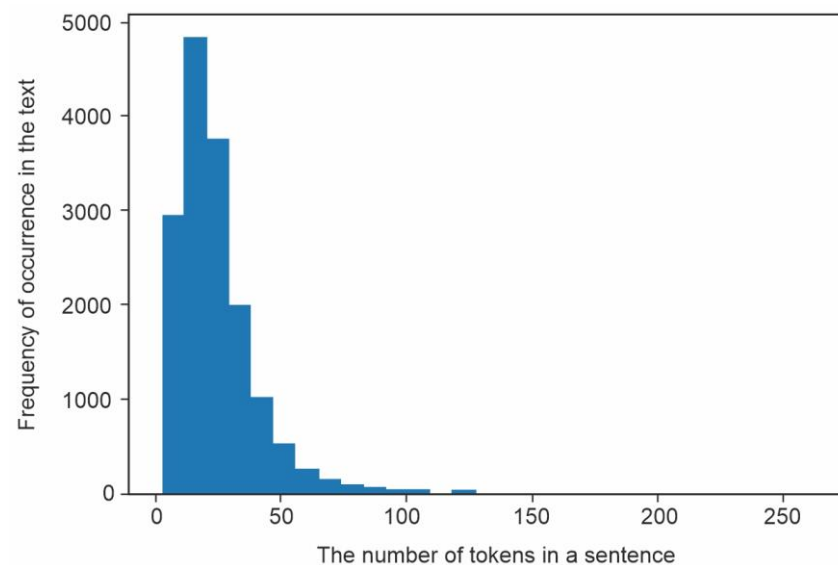


Figure 5. Distribution of the number of tokens in the training sample.

The UA-GEC data set [15], which contains 850 and 160 texts in the training and test samples, respectively, was chosen for the research. According to the authors, the data set contains a total of 20,715 sentences (both with and without errors). At the time of writing, the built-in iteration methods on the UA-GEC corpus only allow you to retrieve complete documents, not individual sentences. The text is annotated in the following format:

I {like=>likes::error_type=Grammar} turtles.

To process and split texts, the Regex library was used [46–48]. Regular expressions for splitting texts into sentences and detecting errors have the following format:

split_pattern = r'\n+'

additional_split_pattern = r'(?<=[^A-Z].?[!\(\.\.\.])+(?=[A-Z"'])'

error_pattern = r'\{(((^[\{]*\(*\))*=>([^\{]*\(*\))*::error_type=([^\{]*)\}\}'

Correct pairs are not deleted, but sentences with a number of tokens greater than four are selected for further processing (two of them are special tokens for the beginning and end of the sentence).

The token dictionary, corresponding to the previously studied model «Ukrainian Roberta» [36], contains 52 thousand elements, where the first notes are the specific tokens, punctuation characters, and the most certain words.

The neural network was trained for 15 epochs with the following hyperparameters:

- $n_epochs = 15$;
- $batch_size = 8$;
- $lr = 0.00001$.

After the 15th epoch, the value of the loss function on the test sample increases (Figure 3), so the 15th epoch is the most optimal value in this case (Table 1). In total, as a result of the distribution of texts according to the above-mentioned expressions, we obtain 15,599 and 2205 sentences in training (train in Figure 6) and test (validation in Figure 6) samples, respectively.

Table 1. The categorical cross-entropy loss (CCEL) values.

Epochs	CCEL Validation (Test)	CCEL Train (Train)
14	0.28125	0.09309
15	0.28119	0.09299
16	0.28139	0.09375

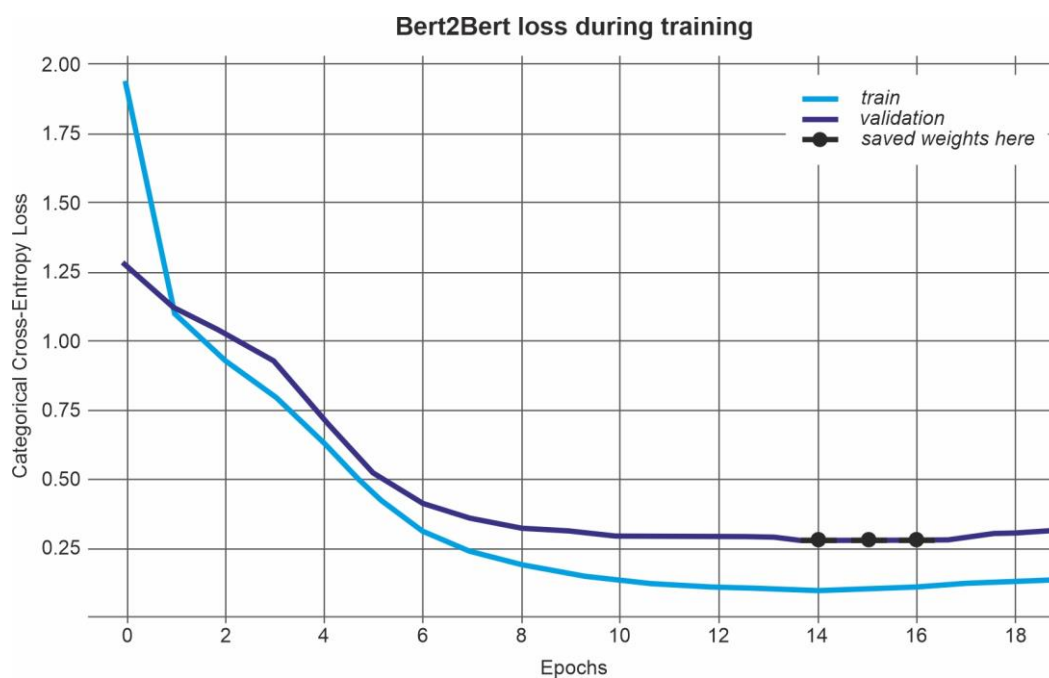


Figure 6. The curve of the loss function during neural network learning.

4.2. Setting the Pre-Trained mT5 MT Model

MT5 [37] is based on the transformer architecture and is pre-trained using the Common Crawl corpus. In total, the neural network is pre-trained on texts written in 101 languages, including Ukrainian.

Due to limited available computing resources, the largest length of the investigated sentences was 20 tokens, with a batch size of two records. It is worth noting that since the neural network is pre-trained using texts written in different languages, the corresponding token dictionary contains parts of Ukrainian words rather than full words. Therefore,

one word is coded with more tokens, and in the context of limited sentence length, this can lead to an incorrect presentation of training examples and their meaning, which is not entirely complete. The neural network was trained for 10 epochs, the lowest value of the loss function on the test sample was reached at the 4th epoch, and the weights of the neural network were also preserved.

The main libraries used in implementing the neural network were PyTorch, Hugging-Face Transformers, Regex, and Pandas [49,50].

Data sets may be used morphologically close to Ukrainian languages. Their simultaneous using in training neural networks can involve the identification of the errors caused by local dialects. There is much local speech in Ukraine, especially in border-line territories. The limits of the paper training neural network are based on the Ukrainian morphology database only. The Ukrainian language was studied because we obviously collaborate with linguists to use text data sets for it.

When taking into account restrictions on available computing resources, the longest sentence must be 20 tokens, and the batch size is two notes. It should be noted that since the neural network is previously trained on the different languages texts (local dialect may be present, and words/phrases and even sentences also belong to another language), the corresponding token dictionary contains rather parts of Ukrainian words than the whole words. That is why one word is coded by a larger number of tokens, so the limited sentence length leads to an incorrect presentation of the training examples and their sense.

The maximum number of tokens is limited by computing resources. (Figure 7). Incorrect presentation of the training examples also is possible in this case.

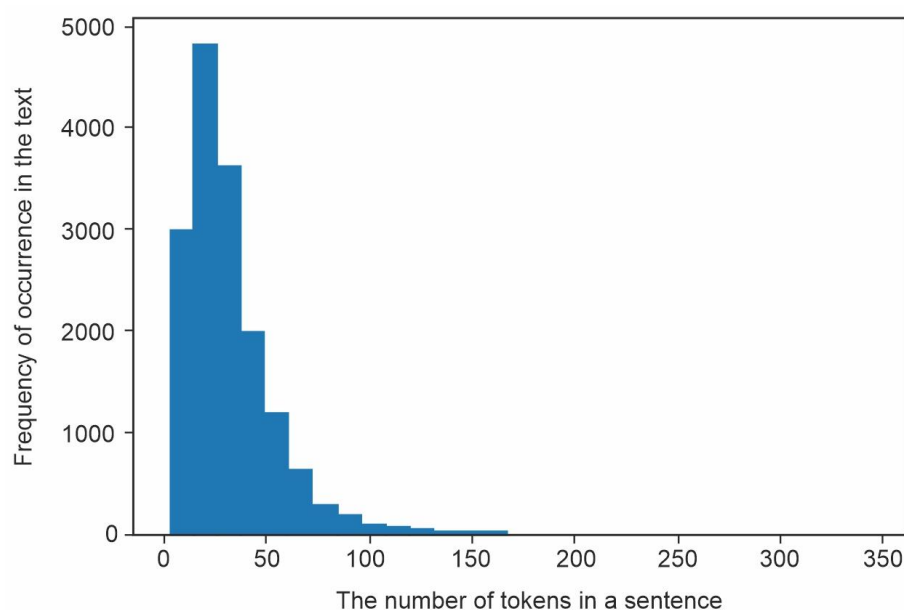


Figure 7. Distribution of token number in training sample.

A neural network was studied during 10 epochs; however, a minimum value of loss function in the test sample was reached during the second epoch (Figure 8).

A neural network study was provided during 10 epochs for comparison, and the scales were the same after the last epoch. It is noticed that the quality of text generation depends on parameters, namely:

- Hyper-parameters values;
- Amount of studied sampling;
- The number of training epochs;
- Architecture and parameters number of neural network;
- Token number of input text;
- Studied sampling (validation data were used or was used all corpora).

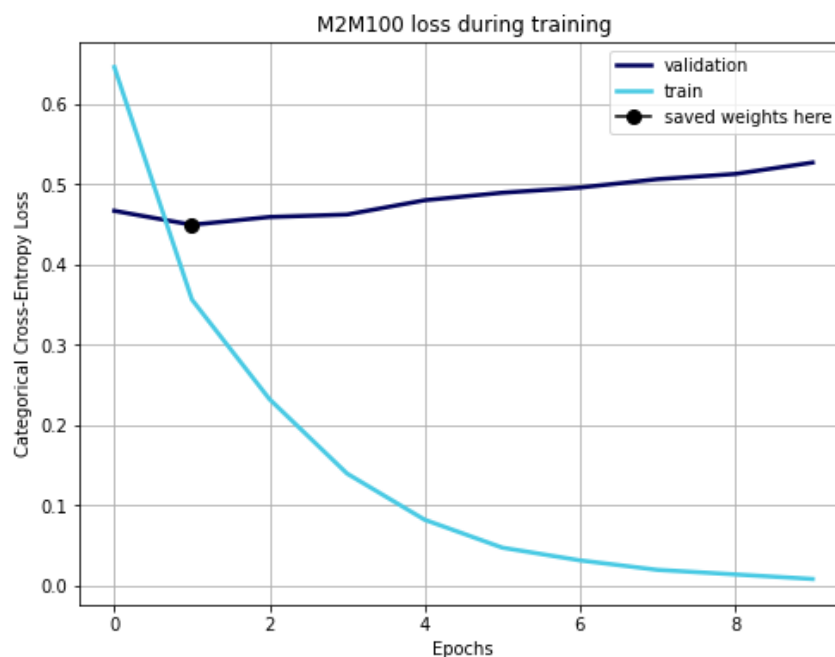


Figure 8. Curve of loss function during training HM.

To demonstrate the work of neural networks, we used our own examples and tasks of the External Independent Assessment (standardized testing, the passing of which is mandatory for all graduates of Ukrainian schools who wish to enter higher educational institutions of Ukraine). The results were obtained for the ‘Ukrainian Roberta’ encoder-decoder. The neural network is good at placing punctuation marks in a simple sentence (Figure 9).

```
predict('я й не думав що лінгвітіа це легкоо')
```

```
tensor([[ 0, 0, 848, 462, 355, 18554, 16, 402, 17134, 14428,
          341, 622, 537, 4222, 4222, 2]], device='cuda:0')
['я й не думав, що лінгвістика – це легко легко']
```

```
predict('Лиши біду, нехай щезає від тебе! - радив спокійно хлопець [...]– Нема з ким говорити!')
```

Input sentence
Лиши біду, нехай щезає від тебе! - радив спокійно хлопець [...]— Нема з ким говорити!

Result
Лиши біду, хай щезає від тебе! — радив спокійно хлопець... — Нема з ким говорити!

```
predict('Лиши біду, нехай щезає від тебе! - радив спокійно хлопець – Нема з ким говорити!')
```

Input sentence
Лиши біду, нехай щезає від тебе! - радив спокійно хлопець — Нема з ким говорити!

Result
Лиши біду, хай щезає від тебе! — радив спокійно хлопець. — Нема з ким говорити!

Figure 9. Placing punctuation marks.

Next example: the neural network corrected the punctuation marks correctly but changed the sentence independently using word combinations known to it (Figure 10). Some-

times the neural network ‘does not understand’ the essence of the task (Figure 11—leave the correct phrase in its initial form). Sometimes the neural network does not ‘understand’ the essence of the task at all (Figure 12). However, in this case, the sentence has a logical structure. In Figure 13, the neural network correctly changed the case of the noun from nominative to vocative, but *гуаш—фарба*, a feminine noun, is not correctly coordinated here, probably because there are no similar examples in the training data set.

```
predict('Я цієї пісні раніше не чув, – сказав студентові Василь: – Ви її всю знаєте?')
```

Input sentence

Я цієї пісні раніше не чув, – сказав студентові Василь: – Ви її всю знаєте?

Result

Я цієї пісні раніше не чув, – сказав студент Василь Васильєв. – Ви її всю історію знаєте?

```
predict('«Це ж хто сказав, що одна ластівка не робить весни?» – переможно вигукнув Таран.')
```

Input sentence

«Це ж хто сказав, що одна ластівка не робить весни?» – переможно вигукнув Таран.

Result

«Це ж хто сказав, що одна лапа не робить весни?» — вигукнув Таран.

Figure 10. Sentence changing by neural network.

```
predict('кришталеве джерело')
```

Input sentence

кришталеве джерело

Result

Решталеве джерело — джерело джерело для того призначення

Figure 11. Neural network fails to understand the task.

In Figure 14, the neural network corrected the numeral correctly but completed the sentence in its own ‘creative’ way. In examples containing only words or phrases, the neural network ‘tries’ to complete the idea by completing sentences on its own. This is a direct consequence of the fact that examples in the training set contain only sentences but not phrases.

However, other methods are generally used to check words and phrases: dictionary check and simple rule check (apostrophe, alternation of the *y-v* sounds, the spelling of *пів-*, *напів-*, etc.). Preliminary results of testing of the neural network show that it defines the essence of the task incorrectly (error correction while preserving the original meaning of the sentence/phrase), which may be the result of the absence of individual words and phrases in the training data set. In contrast, the neural network corrects sentences with patterns from the training data set, which, on the contrary, may signal its overtraining. In addition, the architecture of the RoBERTa model may be the cause of such errors. In Figure 15, the neural network replaced *чим ... тим* with *що ... то*. Some linguists believe that this conjunction is not actually Ukrainian but Surzhyk. Simple sentences are usually a simple task for a neural network (Figure 16). A neural network suggests several results

sorted by higher probability. However, sometimes the most likely sentences contain errors, for example, as in Figure 17a. Additionally, among those proposed in Figure 17b, there is a correct sentence.

predict('експонат в музеї')

Input sentence
експонат в музеї

Result
Експонент в музеї "Фантом", де ви побували на музеї, де й музей "спався на музеї".

predict('далека путь')

Input sentence
далека путь

Result
далека путь — це довгий шлях

Figure 12. Incorrect changing of a sentence.

predict('Слухай-но, Остап, чи не міг би ти купити мені червоний гуаш?')

Input sentence
Слухай-но, Остап, чи не міг би ти купити мені червоний гуаш?

Result
Слухай-но, Остапе, чи не міг би ти купити мені червоний гуаш?

Figure 13. Change in the case of a noun.

predict('п'ятнадцять')

Input sentence
п'ятнадцять

Result
П'ятнадцять років — це п'ятнадцять.

Figure 14. Creativity of the neural network.

Input sentence
Один мудрець вдало підмітив: Чим більше пізнаєш людей, тим більше починаєш любити тварин.

Result
Один мудрець вдало підмітив: що більше пізнаєш людей, то більше починаєш любити тварин.

Figure 15. Correction of Surzhyk by the neural network.

Input sentence вона хотіла би навчатися малювання. Result Вона хотіла би навчатися малювати.	Input sentence Концерт тривав майже півтора години. Result 1 – Концерт тривав майже півтори години. 2 – Конгрес тривав майже півтори години. 3 – Концерт тривав майже півтора години.
Input sentence учора був чудове день. Result Учора був чудовий день.	Input sentence Водій автобуса сказав пасажирам щоб вони оплати проїзд. Result 1 – Водій автобуса сказав пасажирам, щоб вони оплати проїзд. 2 – Водій автобуса сказав пасажирам, щоб вони оплати проїзду. 3 – Водій автобуса сказав пасажирам, щоб вони оплатили проїзд.
Input sentence Хочу подякувати учасників. Result Хочу подякувати учасникам.	Input sentence він дбайливий по відношенню до майна. Result 1 – Він турботливий по відношенню до майна. 2 – Він турботливий за відношенню до майна. 3 – Він турботливий до майна.

Figure 16. Correction of simple sentences.

Input sentence учора був чудове день народження. Result Учора був чудове день народження.	Input sentence учора був чудове день народження. Result 1 – Учора був чудове день народження. 2 – Учора був чудовий день народження. 3 – учора був чудове день народження.
--	---

Figure 17. Wrong correction and variants of correction.

In Figure 18a, the neural network correctly replaced the phrase за професії, but the word архітектор is replaced with a word that is closest in meaning. Similarly, the most likely correct answer (number 1 in the list of suggestions) may have an incorrect sentence (Figure 18b), but the correct variant is among the suggestions. In general, correct sentences in the first position, i.e., with the highest probability, occur 51.6% of the time in test sentences.

Input sentence він за професії архітектор. Result 1 – Він за фахом інженер. 2 – Він за професії інженер. 3 – Він за спеціальністю інженер.	Input sentence Нажал я не знала про це Result 1 – Нажал я не знала про це. 2 – На жаль, я не знала про це. 3 – На мою думку, я не знала про це.
---	--

Figure 18. Right correction.

In some sentences, mT5 offers more correct corrections (Figure 19). However, for the final assessment and comparison of the neural networks, a sufficiently larger test corpus is required.

It is clear from the above examples that mT5 is better at correcting grammar without changing the original meaning of the sentence. However, this model has twice as many parameters as the Roberta-based encoder-decoder (580 million versus 250), which requires significant computational resources for training the neural network and obtaining predictions. The M2M100 neural network corrects grammatical errors much worse than the two previous models (Figure 20).

Roberta		mT5	
Input sentence він за професії архітектор. Result 1 – Він за фахом інженер. 2 – Він за професії інженер. 3 – Він за спеціальністю інженер.		Input sentence він за професії архітектор. Result 1 – Він за професії архітектор. 2 – Він за професією архітектор. 3 – Він – за професії архітектор.	
Input sentence Хочу подякувати учасників. Result Хочу подякувати учасникам.		Input sentence Хочу подякувати учасників. Result 1 – Хочу подякувати учасників. 2 – Хочу подякувати учасників. 3 – Хочу подякувати учасників. – Хочу подякувати учасників.	
Input sentence це сама гірша ситуація. Result 1 – Це сама гірша ситуація. 2 – Я сама гірша ситуація. 3 – – Це сама гірша ситуація.		Input sentence це сама гірша ситуація. Result 1 – Це найгірша ситуація. 2 – Це сама гірша ситуація. 3 – Це сама гірша ситуація.	
Input sentence Ще б ми стільки прочитали сказала Васирина. Result 1 – Ще б ми стільки прочитали, сказала Кирилина. 2 – Ще б ми стільки прочитали, сказала Михайлина. 3 – Ще б ми так прочитали, сказала Кирилина.		Input sentence Ще б ми стільки прочитали сказала Васирина. Result 1 – Ще б ми стільки прочитали, сказала Васирина. 2 – Ще б ми стільки прочитали, – сказала Васирина. 3 – Ще б ми стільки прочитали сказала Васирина.	
Input sentence Стояла ніч красива мов Кармен, червоні й чорні міряла троянди. Result 1 – Стояла ніч, красива Кармен, червоні й чорні міряла троянди. 2 – Стояла ніч, красива Кармен, червоні й білі міряла троянди. 3 – Стояла ніч, красива Кармен, білі й чорні міряла троянди.		Input sentence Стояла ніч красива мов Кармен, червоні й чорні міряла троянди. Result 1 – Стояла ніч красива, мов Кармен, червоні й чорні міряла троянди. 2 – Стояла ніч красива мов Кармен, червоні й чорні міряла троянди. 3 – Стояла ніч красива, мов Кармен, червон та чорні міряла троянди.	
Input sentence Хоча снігу не має про/те надворі мороз пробирає поспині. Result 1 – Хоча снігу немає про це надворі, мороз мороз пробирає поспині. 2 – Хоча снігу не має про це надворі, мороз мороз пробирає поспині. 3 – Хоча снігу не має бути надворі, мороз морози пробирає поспині.		Input sentence Хоча снігу не має про/те надворі мороз пробирає поспині. Result 1 – Хоча снігу немає про те надворі мороз пробирає по спині. 2 – Хоча снігу не має про те надворі мороз пробирає поспині. 3 – Хоча снігу не має про те надворі мороз пробирає по спині.	

Figure 19. Variants of correction by different neural network models.

Input sentence він за професії архітектор. Result 1 – Він за професії архітектору. 2 – Він за професії архітекторе. 3 – Він за професії архітектор.	Input sentence Водій автобуса сказав пасажирам щоб вони оплати проїзд. Result 1 – Водій автобуса сказав пасажирам, щоб вони оплати проїзд. 2 – Водій автобуси сказав пасажирам, щоб вони оплати проїзд. 3 – Водій автобусу сказав пасажирам, щоб вони оплати проїзд.
Input sentence Хочу подякувати учасників. Result 1 – Хочу подякувати учасників. 2 – Хоча подякувати учасників. 3 – Хочу підякувати учасників.	Input sentence Концерт тривав майже півтора години. Result 1 – Концерт тривав майже півтора години. 2 – Конгрес тривав майже пів року. 3 – Концерт тривав майже півтори години.

Figure 20. Work of the M2M100 neural network.

5. Discussion

There are currently no standard metrics for evaluating the quality of functioning of GEC systems, so the c [43] and METEOR [44] metrics are usually used.

In order to estimate these systems in general, metrics of machine translation quality are used, but there are some disadvantages.

A disadvantage of BLEU can be considered comparing strings, resulting in the possibility of several methods ignoring the correction of some grammatical errors. Additionally, GEC quality methods cannot be compared to some other translation systems since, in the first case, the major number of N-gram in the sentences coincided (or entirely coincided if grammatical errors were absent and the sentence was not changed).

The following results were obtained for three artificial neural networks (Figure 21):

- ‘Ukrainian Roberta’ Encoder-Decoder—0.697;
- Google’s mT5—0.908;

- Facebook’s M2M100—0.847.

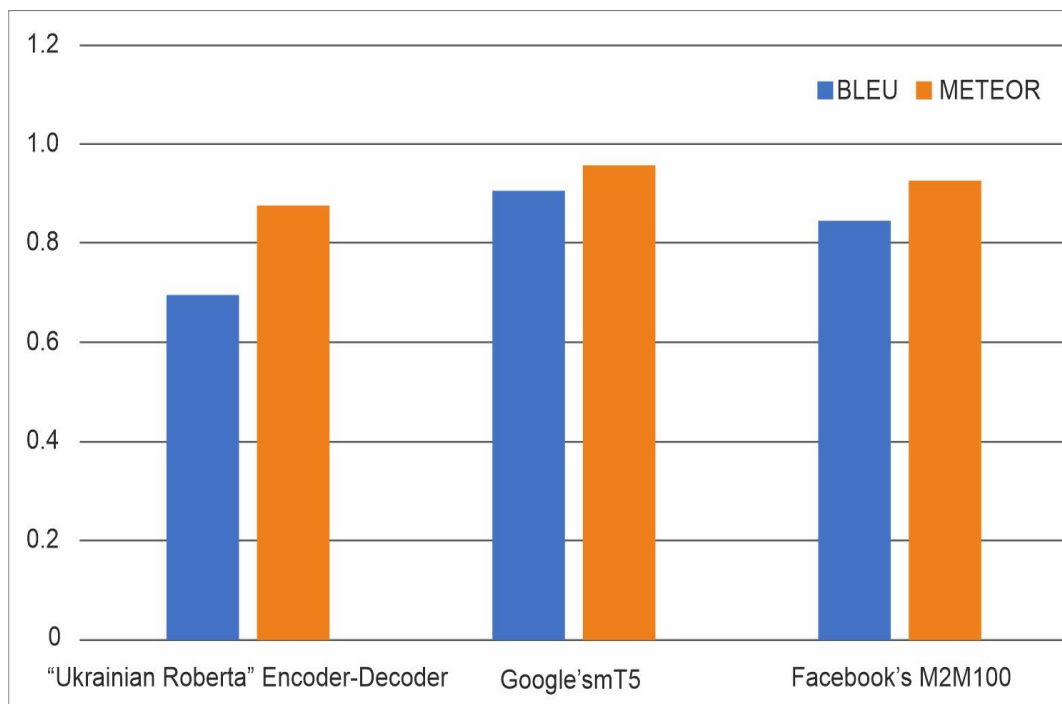


Figure 21. The results of applying the neural network.

Similarly, as in BLUE-metrics, the main unit to estimate in METEOR-metrics is a sentence. As a result of metrics processing on the phase level, the correlation with the human decision, according to [44], is 0.964, while the correlation with BLUE is 0.817 for the same input data. The maximum correlation with experts' estimation is 0.403 on the sentence level. The algorithm first realizes text leveling between two sentences, a string of standard translations, and a string of input estimating text. This metric uses several steps to establish correspondence between words of machine translation and standard translation to compare two strings:

1. The exact establishment of correspondence means to determine strings that are identical in standard and machine translation;
2. The establishment of stem correspondence is called 'steaming', which is determined by words with the same stem in standard and machine translation;
3. The establishment of synonyms correspondence means determining words being synonyms corresponding to WordNet.

There are obtained the following results for three artificial neural networks (Figure 21):

- 'Ukrainian Roberta' Encoder-Decoder—0.876;
- Google's mT5—0.956;
- Facebook's M2M100—0.925.

6. Conclusions

- Building a good machine learning model for GEC in morphologically complex texts requires a large amount of parallel or manually labeled data. Manual data annotation requires much effort by professional linguists, which makes the creation of text corpora a time- and resource-consuming process.
- Solving the task of automatic detection and correction of errors in Ukrainian texts requires further research due to the small number of works focusing on the study of the Ukrainian language. In addition, according to the results of the study of S.D. Pohorily and Kramova A.A. [27], the methods used to study the English language cannot be

used for Ukrainian since the latter is a much more complex and morphologically richer language.

- The appearance of the new transformer deep learning architecture in 2017 [3], which includes an attention mechanism, allowed for significant simplification of the development of language models. A disadvantage of this approach for developing models for different languages is the need for large corpora of annotated or parallel data. The only (at the time of writing) Ukrainian-language data set [15] contains only tens of thousands of sentences, and such a number of training samples is not enough to create an automatic intelligent system for identifying and correcting grammatical errors for Ukrainian texts.
- The development of a quality system for checking the grammatical correctness of sentences in Ukrainian texts requires a combination of machine learning algorithms with several different types of methods, in particular, the application of expert knowledge in computer linguistics.
- The best value for both BLEU and METEOR is obtained for the mT5 model. The results are consistent with the analysis of our own examples, in which the most accurate error corrections without changing the initial sentence were obtained for this neural network. The results of applying the neural network at the phrase level are the following: the correlation with the human decision was 0.964, while the correlation with BLUE was 0.817 on the same set of input data. At the sentence level, the maximum correlation with the experts' assessment was 0.403. Calculated metrics allow only partial comparison of the models since most of the words and phrases in the original and corrected sentences match. The best value of both BLEU (0.908) and METEOR (0.956) was obtained for mT5. Mt5 has a higher BLEU score than the 'Ukrainian Roberta' encoder–decoder (0.697); however, subjectively evaluating the results of correcting examples, Mt5 performs much worse. Mt5 also has a higher METEOR score than the 'Ukrainian Roberta' encoder–decoder (0.876). A cross-validation procedure can be the forthcoming step of our research.

Author Contributions: Conceptualization, V.L. and V.V.; methodology, P.P. and M.V.; software, N.K.; validation, V.L., V.V. and P.P.; formal analysis, M.V.; investigation, V.V.; resources, N.K.; writing—original draft preparation, V.L. and V.V.; writing—review and editing, P.P. and M.V.; visualization, N.K.; supervision, V.V.; project administration, V.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Naghshnejad, M.; Joshi, T.; Nair, V.N. Recent Trends in the Use of Deep Learning Models for Grammar Error Handling. *arXiv* **2020**, arXiv:2009.02358.
2. Leacock, C.; Chodorow, M.; Gamon, M.; Tetreault, J. *Automated Grammatical Error Detection for Language Learners. Synthesis Lectures on Human Language Technologies*, 2nd ed.; Springer: Berlin, Germany, 2014. [\[CrossRef\]](#)
3. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**. [\[CrossRef\]](#)
4. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations, ACL Anthology, Online, 16–20 November 2020; pp. 38–45. [\[CrossRef\]](#)
5. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. [\[CrossRef\]](#)

6. Bick, E. DanProof: Pedagogical Spell and Grammar Checking for Danish. In Proceedings of the International Conference Recent Advances in Natural Language Processing, Hissar, Bulgaria, 7–9 September 2015; pp. 55–62.
7. Gakis, P.; Panagiotakopoulos, C.T.; Sgarbas, K.N.; Tsalidis, C.; Verykios, V.S. Design and construction of the Greek grammar checker. *Digit. Scholarsh. Humanit.* **2017**, *32*, 554–576. [CrossRef]
8. Deksne, D. A New Phase in the Development of a Grammar Checker for Latvian. *Frontiers in Artificial Intelligence and Applications* **2016**, *289*, 147–152. [CrossRef]
9. Sorokin, A. Spelling Correction for Morphologically Rich Language: A Case Study of Russian. In Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing, Valencia, Spain, 3–4 April 2017; pp. 45–53. [CrossRef]
10. Rozovskaya, A.; Roth, D. Grammar Error Correction in Morphologically Rich Languages. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 1–17. [CrossRef]
11. Gill, M.S.; Lehal, G.S. A Grammar Checking System for Punjabi. In Proceedings of the Companion volume: Demonstrations, Manchester, UK, 18–22 August 2008; pp. 149–152.
12. Go, M.P.; Borra, A. Developing an Unsupervised Grammar Checker for Filipino Using Hybrid N-grams as Grammar Rules. In Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Oral Papers, Seoul, Republic of Korea, 28–30 October 2016; pp. 105–113.
13. Shaalan, K.F. Arabic GramCheck: A grammar checker for Arabic. *Softw. Pract. Exp.* **2005**, *35*, 643–665. [CrossRef]
14. Wang, Y.; Wang, Y.; Liu, J.; Liu, Z. A Comprehensive Survey of Grammar Error Correction. *arXiv* **2020**, arXiv:2005.06600.
15. Syvokon, O.; Nahorna, O. UA-GEC: Grammatical Error Correction and Fluency Corpus for the Ukrainian Language. *arXiv* **2021**, arXiv:2103.16997.
16. Lardinois, F. Grammarly Goes Beyond Grammar. TechCrunch 2019. Available online: <https://techcrunch.com/2019/07/16/grammarly-goes-beyond-grammar/> (accessed on 1 December 2022).
17. Lardinois, F. Grammarly Gets a Tone Detector to Keep You Out of Email Trouble. TechCrunch 2019. Available online: <https://techcrunch.com/2019/09/24/grammarly-gets-a-tone-detector-to-keep-you-out-of-email-trouble/> (accessed on 1 December 2022).
18. Grammarly Inc. About Us. Available online: <https://www.grammarly.com/about> (accessed on 1 December 2022).
19. Grammarly Inc. Does Grammarly Support Languages Other than English? Available online: <https://support.grammarly.com/hc/en-us/articles/115000090971-Does-Grammarly-support-languages-other-than-English-> (accessed on 1 December 2022).
20. LanguageTool. Languages. Available online: <https://dev.languagetool.org/languages> (accessed on 29 December 2022).
21. LanguageTool. Error Rules for LanguageTool. Available online: https://community.languagetool.org/rule/list?offset=0&max=10&lang=uk&filter=&categoryFilter=&_action_list=%D0%A4%D1%96%D0%BB%D1%8C%D1%82%D1%80 (accessed on 13 December 2022).
22. LanguageTool. About. Available online: <https://languagetool.org/about> (accessed on 15 December 2022).
23. Korobov, M. Morphological Analyzer and Generator for Russian and Ukrainian Languages. *arXiv* **2015**. [CrossRef]
24. Tmienova, N.; Sus, B. System of Intellectual Ukrainian Language Processing. In Proceedings of the XIX International Conference on Information Technologies and Security, Kyiv, Ukraine, 28 November 2019; pp. 199–209.
25. Pogorilyy, S.; Kramov, A.A. Method of noun phrase detection in Ukrainian texts. *arXiv* **2020**, arXiv:2010.11548. [CrossRef]
26. Glybovets, A.; Tochytskyi, V. Tokenization and stemming algorithms for the Ukrainian language. *NaUKMA Res. Papers. Comput. Sci.* **2017**, *198*, 4–8.
27. Kholodna, N.; Vysotska, V.; Markiv, O.; Chyrun, S. Machine Learning Model for Paraphrases Detection Based on Text Content Pair Binary Classification. *CEUR Workshop Proc.* **2022**, *3312*, 283–306.
28. Kholodna, N.; Vysotska, V.; Albota, S. A Machine Learning Model for Automatic Emotion Detection from Speech. *CEUR Workshop Proc.* **2021**, *2917*, 699–713.
29. Abbasi, A.; Javed, A.R.; Iqbal, F.; Kryvinska, N.; Jalil, Z. Deep learning for religious and continent-based toxic content detection and classification. *Sci. Rep.* **2022**, *12*, 17478. [CrossRef]
30. Bashir, M.F.; Arshad, H.; Javed, A.R.; Kryvinska, N.; Band, S.S. Subjective Answers Evaluation Using Machine Learning and Natural Language Processing. *IEEE Access* **2021**, *9*, 158972–158983. [CrossRef]
31. Kapłan, T.; Mazurkiewicz, J. The Method of Inflection Errors Correction in Texts Composed in Polish Language—A Concept. *Lect. Notes Comput. Sci.* **2005**, *3697*, 853–858. [CrossRef]
32. Jędrzejowicz, P.; Strykowski, J. A Neural Network Based Morphological Analyser of the Natural Language. *Adv. Soft Comput.* **2005**, *31*, 199–208. [CrossRef]
33. Wróbel, K. KRNNT: Polish recurrent neural network tagger. In Proceedings of the 8th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics, Poznań, Poland, 17–19 November 2017; pp. 386–391.
34. Shishkina, Y.; Lyashevskaya, O. Sculpting Enhanced Dependencies for Belarusian. *Lect. Notes Comput. Sci.* **2022**, *13217*, 137–147. [CrossRef]
35. Rozovskaya, A.; Chang, K.-W.; Sammons, M.; Roth, D. The University of Illinois System in the CoNLL-2013 Shared Task. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, Sofia, Bulgaria, 8–9 August 2013; Association for Computational Linguistics. pp. 13–19. Available online: <https://aclanthology.org/W13-3602/> (accessed on 1 December 2022).

36. Radchenko, V. Ukrainian Roberta. Available online: <https://github.com/youscan/language-models> (accessed on 17 December 2022).
37. Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; Raffel, C. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 July 2021; pp. 483–498. [CrossRef]
38. Fan, A.; Bhosale, S.; Schwenk, H.; Ma, Z.; El-Kishky, A.; Goyal, S.; Baines, M.; Celebi, O.; Wenzek, G.; Chaudhary, V.; et al. Beyond English-Centric Multilingual Machine Translation. *arXiv* **2020**. [CrossRef]
39. Tang, Y.; Tran, C.; Li, X.; Chen, P.-J.; Goyal, N.; Chaudhary, V.; Gu, J.; Fan, A. Multilingual Translation with Extensible Multilingual Pretraining and Finetuning. *arXiv* **2020**, arXiv:2008.00401.
40. Platen, V.P. Leveraging Pre-trained Language Model Checkpoints for Encoder-Decoder Models. Available online: <https://huggingface.co/blog/warm-starting-encoder-decoder> (accessed on 19 December 2022).
41. Rothe, S.; Narayan, S.; Severyn, A. Leveraging Pre-trained Checkpoints for Sequence Generation Tasks. *arXiv* **2019**, arXiv:1907.12461. [CrossRef]
42. Napoles, C.; Sakaguchi, K.; Post, M.; Tetreault, J. Ground truth for grammatical error correction metrics. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; pp. 588–593. [CrossRef]
43. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002; pp. 311–318. [CrossRef]
44. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 29 June 2005; pp. 65–72.
45. Platen, V.P. Encoder-Decoder Models Don’t Need Costly Pre-Training to Yield State-of-the-Art Results on seq2seq Tasks. Available online: <https://twitter.com/patrickplaten/status/1325844244095971328> (accessed on 20 December 2022).
46. RegEx. Available online: <https://regex101.com/r/F8dY80/3> (accessed on 18 December 2022).
47. RegEx. Available online: <https://www.guru99.com/python-regular-expressions-complete-tutorial.html> (accessed on 20 December 2022).
48. RegExr. Available online: <https://regexpr.com/> (accessed on 21 December 2022).
49. Bengfort, B.; Bilbro, R.; Ojeda, T. *Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning*; O’Reilly Media, Inc.: Boston, MA, USA, 2018.
50. Transformers—Hugging Face. Available online: <https://huggingface.co/docs/transformers/main/en/index> (accessed on 21 December 2022).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.