DOI: 10.15514/ISPRAS-2026-38(3)-11

Subword-Level Grammatical Error Correction: A Universal Approach

[1] *I.A. Khabutdinov, ORCID: 0009-0002-8876-1746 <khabutdinov.ia@phystech.edu>*
[1,2] *A.V. Grabovoy, ORCID: 0000-0002-4031-0025 <grabovoy@ipu.ru>*
[2] *Yu.V. Chekhovich, ORCID: 0000-0002-5204-5484 <chekhovich@ipu.ru>*
[2] *A.S. Kildyakov, ORCID: 0009-0005-9591-0983 <kildyakov@ipu.ru>*
[2] *A.A. Ivakhnenko, ORCID: 0009-0001-2843-6848 <ivakhnenko@ipu.ru>*

[1] *Moscow Institute of Physics and Technology,*
9 Institutskiy per., Dolgoprudny, Moscow Region, 141701, Russian Federation.

[2] *V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences,*
65 Profsoyuznaya street, Moscow 117997, Russia.

Abstract. In this study, we propose a fully automatic methodology for data generation, correction rule vocabulary construction, and Sequence Tagging model training that specifically targets Grammatical Error Correction. Our approach operates at the SentencePiece subword level, using basic transformations – *keep*, *append*, *replace* and *delete* – that are universally applicable across languages, thereby eliminating the need for grammar-specific operations. By using the Levenshtein algorithm to generate ground truth corrections and editorial prescriptions, we obtained a completely invariant and language-independent dataset generation process. We applied our method to the Sequence Tagging model GECToR and achieved comparable quality results for English with $F_{0.5}$ scores of 62.4 on the CoNLL-2014 (test set) and 61.9 on the BEA-2019 (test set), without manual rule design or manual annotation of error spans/types. The results indicate that subword-level universal edits can provide a practical alternative to grammar-specific operations, while requiring only parallel correction data.

Keywords: grammar error corrections; neural language processing; transformers; machine learning.

For citation: Khabutdinov I.A., Grabovoy A.V., Chekhovich Yu.V., Kildyakov A.S., Ivakhnenko A.A. Subword-Level Grammatical Error Correction: A Universal Approach. Trudy ISP RAN/Proc. ISP RAS, vol. 38, issue 3, part 1, 2026, pp. 187-196. DOI: 10.15514/ISPRAS-2026-38(3)-11.

Acknowledgements. The authors acknowledge the support of this research by the Ministry of Science and Higher Education of the Russian Federation under agreement No. 075-03-2025-662/13 (October 29, 2025), associated with the project» Applied Research on the Implementation of Artificial Intelligence Technologies in Higher Education".

Исправление грамматических ошибок на уровне подслов: универсальный подход

[1] *И.А. Хабутдинов, ORCID: 0009-0002-8876-1746 <khabutdinov.ia@phystech.edu>*
[1,2] *А.В. Грабовой, ORCID: 0000-0002-4031-0025 <grabovoy@ipu.ru>*
[2] *Ю.В. Чехович, ORCID: 0000-0002-5204-5484 <chekhovich@ipu.ru>*
[2] *А.С. Кильдяков, ORCID:0009-0005-9591-0983 <kildyakov@ipu.ru>*
[2] *А.А. Ивахненко, ORCID: 0009-0001-2843-6848 <ivakhnenko@ipu.ru>*

[1] *НИУ Московский физико-технический институт,*
Россия, 141701, Московская область, г. Долгопрудный, Институтский пер., 9.

[2] *Институт проблем управления им. В. А. Трапезникова Российской академии наук,*
Россия, 117997, Москва, ул. Профсоюзная, 65.

Аннотация. В данном исследовании мы предлагаем полностью автоматическую методологию генерации данных, построения словаря правил исправления и обучения модели разметки последовательностей, специально ориентированную на исправление грамматических ошибок. Наш подход работает на уровне подслов SentencePiece, используя базовые преобразования – сохранение, добавление, замену и удаление, которые универсально применимы во всех языках, тем самым устраняя необходимость в грамматически-специфичных операциях. Используя алгоритм Левенштейна для генерации истинных исправлений и редакционных предписаний, мы получили полностью неконтролируемый и независимый от языка процесс генерации наборов данных. Мы применили наш метод к модели разметки последовательностей GECToR и достигли сопоставимых результатов качества для английского языка с оценками $F_{0.5}$ 62,4 на CoNLL-2014 (тестовый набор) и 61,9 на BEA-2019 (тестовый набор), без ручного конструирования правил и аннотации типов ошибок. Это показывает, что универсальные правила на уровне подслов могут стать альтернативой грамматическим операциям, при этом требуя только параллельные тексты – с ошибками и без ошибок.

Ключевые слова: исправление грамматических ошибок; обработка естественного языка; трансформеры; машинное обучение.

Для цитирования: Хабутдинов И.А., Грабовой А.В., Чехович Ю.В., Кильдяков А.С., Ивахненко А.А. Исправление грамматических ошибок на уровне подслов: универсальный подход. Труды ИСП РАН, том 38, вып. 3, часть 1, 2026 г., стр. 187–196 (на английском языке). DOI: 10.15514/ISPRAS–2026–38(3)-11.

Благодарности. Исследования проведены в рамках соглашения с Минобрнауки России от 29.10.2025 №075-03-2025-662/13, тема проекта: «Прикладные исследования по внедрению технологий искусственного интеллекта в высшем образовании».

1. Introduction

Grammatical Error Correction (GEC) is an essential task [1-4] as it ensures clarity, precision, and credibility in written communication. Grammatical mistakes can lead to misunderstandings and weaken the overall message impact. Therefore, correcting grammatical errors is crucial for effective communication and ensuring that the intended message is received accurately. Consequently, GEC plays a vital role in applications such as: checking essays [5], correcting chat messages [6], and correcting texts written by foreign students [7]. The majority of the GEC study was conducted in English [8], a field that has already shown positive outcomes. This issue is understudied in many other languages. The main reason for this is the lack of annotated data.

At the moment, the two most studied approaches are Sequence Tagging [9-11] and Machine Translation [12-13]. The Sequence Tagging (ST) GECToR [9] model shows a high quality of work for the English language [14]. ST models are efficient because they do not require a lot of training data and are easy to interpret. However, ST models suffer from other problems – the need for annotated data and the creation of a rule vocabulary to perform corrections.

In this study, we propose a fully automatic methodology for data generation, construction of a correction rule vocabulary and training of ST models. Our approach is based on moving to the SentencePiece [15] subword level in GEC. We used only "basic transformations" common to all languages – *keep*, *append*, *replace* and *delete*. Thus, we got rid of grammatical dependencies completely. For our experiments, we applied our approach to the ST model GECToR [9], which shows high results for English. We have shown that such a method produces results of comparable quality while being completely automatic. Also, this approach can be adapted to any language, as it does not require any knowledge of grammatical rules or manual annotations.

2. Subword-level motivation

Tokenization plays a critical role in the computational pipeline [16, 17], primarily due to its ability to reduce vocabulary size and improve model performance. Various tokenization schemes were employed depending on the model. For instance, Byte Pair Encoding (BPE) was utilized for RoBERTa [18], WordPiece for BERT [19], and SentencePiece for XLNet [20]. In this study, "subword" refers to a unit obtained in the process of character sequence partitioning by a tokenizer of the corresponding transformer-based model.

In the context of the ST models, the correction rule vocabulary has been built on the word level. This means that in the training/inference phase, several subwords within a word are matched by a single correction rule from the vocabulary.

In [9] there were "basic transformations" and "g-transformations". Basic transformations perform the whole-word editing operations – *delete*, *keep*, *replace* and *append*. G-transformations perform task-specific operations – change verb form, merge/split words, change noun number, case words. Thus, to develop "g-transformations" for an arbitrary language we need to know the grammatical rules. It also requires annotated data and additional functionality to map the assessor labels to the correction rule vocabulary.

At the subword level, we are not required to develop task-specific operations, since each word can be corrected in a finite number of "basic transformations" without changing the word completely. It's a crucial point, since the task becomes completely automatic:

1) the requirement for annotated data is obviated by the ability to automatically generate markup at the subword level utilizing the backtracking in the Levenshtein algorithm [21] employing only "basic transformations", as it supports all four operations — *delete*, *keep*, *replace*, *append*. All that is needed is parallel data — sentences with errors and sentences without errors;
2) the development of a model vocabulary is rendered unnecessary since all correction rules can be automatically derived from "basic transformations" at the subword level.

3. Problem statement

Let us denote the set of source sequences $X = \{s_i | s_i = [x_1, x_2, \dots, x_{n_i}]\}^N{}_{i=0}$, where each source sequence s_i, partitioned with a tokenizer, is represented as a sequence of subwords of length n_i, N is the total number of sentences.

Let the set of target sequences with a tokenization of length m_i be similarly given $Y = \{t_i | t_i = [y_1, y_2, \dots, y_{m_i}]\}^N{}_{i=0}$.

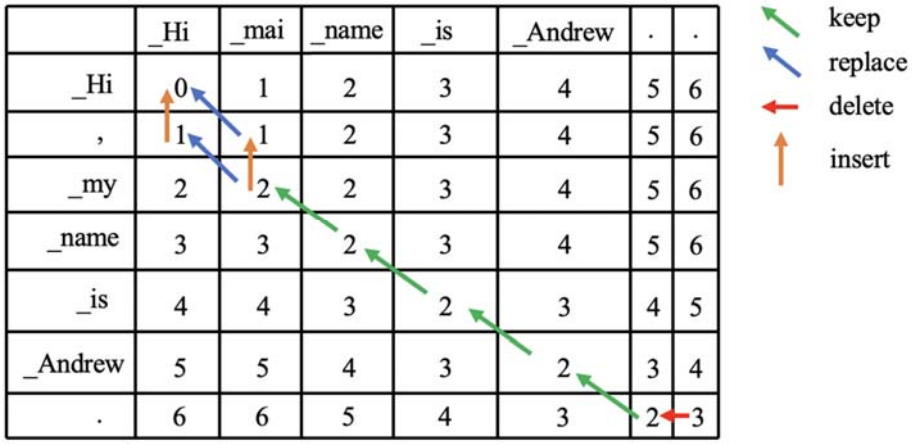
Let a correction rule vocabulary W with the rules *append*, *keep*, *replace* and *delete* be given. We need to find the set of all possible sequences of corrective transformations with minimum number of insertion, deletion, replacement operations $F = \{w_{ij} | w_{ij} \circ s_i \rightarrow t_i\}^N{}_{i=0}$, where sequence of correction rules from the vocabulary W, where $w_{ij} = \{w^*{}_1, w^*{}_2, \dots, w^*{}_{n_i}\}_{j \in \{0,\dots,o_i\}}$, where o_i –

number of correction rule sequences of the minimum length to transform s_i into t_i. The $\circ$ operation denotes the element-by-element application of correction rules to subwords.

4. Algorithm description

In this section, we describe the synthetic annotated data generation process. The source sequence $s_k = [x_1, x_2, \dots, x_{n_k}]$ is a tokenized k-th sentence with errors, the target sequence is a tokenized k-th sentence $t_k = [y_1, y_2, \dots, y_{m_k}]$ without errors, where n_k and m_k are the lengths of the source and target sequences respectively.

In our formulation, the problem can be reduced to the problem of retrieving an editorial prescription. We use the Levenshtein [21] distance, which is defined as the minimum number of *append*, *delete* and *replace* operations required to convert one sequence to another.



	_Hi	_mai	_name	_is	_Andrew	.	.
_Hi	0	1	2	3	4	5	6
,	1	1	2	3	4	5	6
_my	2	2	2	3	4	5	6
_name	3	3	2	3	4	5	6
_is	4	4	3	2	3	4	5
_Andrew	5	5	4	3	2	3	4
.	6	6	5	4	3	2	3

Fig. 1. The Levenshtein matrix and editorial prescriptions between source and target sequences. We can observe that there may be more than one prescription.

The Levenshtein distance between the source sequence and the target can be efficiently computed using a two-dimensional matrix, which is filled using a dynamic programming algorithm. Let D_{ij} be an edit distance between prefixes $s_k[0..i]$ and $t_k[0..j]$ of lengths i and j, where $D_{0j} = j$ and $D_{i0} = i$. The other values are determined by a recurring relationship:

$$D_{ij} = D_{i-1,j-1} if\ s_k[i] = t_k[j]\ and\ 1 + min(D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1})\ otherwise.$$

For example, Fig. 1 shows the Levenshtein distance matrix for the tokenized source sentence "Hi mai name is Andrew.." and the tokenized target sequence "Hi, my name is Andrew.". The edit distance between the source and target sequences is in the bottom-right corner of the matrix and is equal to 3.

An edit prescription is a sequence of rules (keep, replace, append, delete) that describes how to transform a source sequence into a target sequence using a minimum number of edit operations. In order to find ground truth corrections, we need to find an editorial prescription. As shown in Fig. 1, an editorial prescription can be obtained by backtracking on the calculated distance table. Starting at the bottom-right corner of the table, build paths to all adjacent cells corresponding to smaller subtasks and smaller editing distances. Moving up – inserting a subword into the source sequence (*append*), moving left – deleting a subword (*delete*), moving diagonally – replacing a subword (replace) or keeping unchanged (*keep*) in case of a match of subwords.

The number of editorial prescriptions is equal to the number of paths in the graph of subtasks that have a minimum cost. As we can observe for our example, there are two editorial prescriptions — {*delete* **.** ; *append* **_my** ; *replace*_**mai** with **,**} and {*delete* **.** ; *append***,** ; *replace* **_mai** with **_my**}. Thus we need to find the most optimal one. For brevity, the *keep* rules have been omitted.

We denote by $EP_k = \{e_1, e_2, \dots, e_{o_k}\}$ the set of editorial prescriptions for a pair of sequences (s_k, t_k), where o_k – the number of editorial prescriptions for the k-th pair.

In order to find the most optimal editorial prescription for the k-th pair, we take source and target subwords similarity into account when applying the *replace* subword rule. Let $R_l \in e_l, l \in \{1,2, \dots, o_k\}$ be the set of all *replace* rules with cardinality p_l for an arbitrary editorial prescription e_l:_

$$R_l = \{replace\_t_{1i}\_t_{2i}\}^{p_l}_{i=0},$$

where source subword $t_{1i} \in s_k$, target subword $t_{2i} \in t_k$.

Let us introduce subword similarity:

$$\sigma_l = \sum_{i=0}^{p_l} LevenshteinDist(t_{1i}, t_{2i}),$$

where $LevenshteinDist$ is the function calculating Levenshtein distance between t_{1i} and t_{2i} at the level of symbols within subwords. The editorial prescription $e^*_l: l = arg\ min\{\sigma_1, \sigma_2, \dots, \sigma_{o_k}\}$ is the most optimal. These are gold corrections for the pair (s_k, t_k).

5. Experiments

In this section, we describe the pipeline and the problems we tackled while building it. In order to evaluate the quality of our approach we use the Sequence Tagging architecture GECToR which shows high performance in the GEC problem. For a valid comparison, we utilized the same data and repeated the training process with the same parameters.

5.1 Data description

Table 1 describes the data statistics that were used in [9] and in our research. As we can see from the table, five datasets were employed – PIE-synthetic [22], Lang-8 [23, 12], NUCLE [24], FCE [25] and W\&I+LOCNESS [26].

5.1.1 Algorithm constraint

Our backtracking procedure enumerates all minimum-cost edit paths to construct the set of editorial prescriptions, which may lead to a combinatorial blow-up for long sequences with many differences. In such cases, we cannot reliably recover gold alignments within resource limits, and those sentence pairs are excluded from training. Concretely, compared to the data statistics reported in [9], we were able to use 887,338 non-synthetic sentence pairs out of 1,073,096 (≈ 82.7%). The excluded pairs are primarily those with ambiguous or excessively large numbers of optimal paths during backtracking.

5.1.2 Training data

As shown in Table 1, for the first stage of training, as in [9], we used 9M sentences from PIE-synthetic. However, due to limitations, only 6.7M sentences match the [9]. Towards the objective of obtaining a total of 9M we employed the other PIE-synthetic sentences.

Regarding the non-synthetic data, we can observe from Table 1 that we used only 78% of the total. The non-synthetic data were employed for fine-tuning in training stage II and III.

Table 1. Training datasets at each stage with the corresponding number of sentences in [9] (word-level) and in our research (subword-level).

Dataset	#sents		Training stage
	Subword-level	Word-level	
PIE-synthetic	9,000,000	9,000,000	I
Lang-8	787,613	947,344	II
NUCLE	51,929	56,958	II
FCE	25,968	34,490	II
W&I+LOCNESS	21,828	34,304	II, III

5.1.3 Evaluation data

To compare the quality of the model's performance, we used the same test data – CoNLL-2014 [8] and BEA-2019 [27] test sets. For the evaluation, we utilized scorers – M^2 [21] and ERRANT [27], corresponding to the tasks.

5.2 Model training

The GECToR architecture consists of a transformer-based encoder and two linear classifiers. The first classifier predicts the presence of an error in a subword – error detector. The second classifier predicts a specific rule from the vocabulary to be applied to correct the error or to leave it unchanged if the most likely rule is *keep* – error tagger.

If the maximum probability among error detector predictions within a single sentence is less than a certain threshold, then that sentence is not subject to correction. Otherwise, error tagger predictions are used for correction. In [9], the threshold is equal to 0.66, in our work we empirically found the best value to be 0.05.

5.2.1 Word-level correction

In [9], the correction rule vocabulary has been compiled at the word-level. This means that in the training/inference phase, several subwords within a word are matched by a single correction rule from the vocabulary.

In [9] there were “basic transformations” and “g-transformations”. Basic transformations perform the whole word edit operations – *delete*, *keep*, *replace*, *append*. G-transformations perform task-specific operations – change verb form, merge/split words, change noun number, case words. Thus, to develop “g-transformations” for an arbitrary language we need to know the grammatical rules. It also requires to have annotated data and additional functionality to map the assessor labels to the correction rule vocabulary. The vocabulary consists of 5000 rules, of which 4971 are “basic transformations” and 29 are “g-transformations”.

5.2.2 Subword-level correction

In our approach, “g-transformations” are completely absent, thus we are spared from manually composing correction rules. This is a crucial moment, as generalising the grammar of a language using manual rules is a non-trivial task for the languages with rich morphology.

We obtained our vocabulary at the subword-level from the training data using only basic transformations – *delete*, *keep*, *replace* and *append*. So, this process is completely automatic, all that is needed is parallel data. The size of the resulting vocabulary is 25714 rules.

The induced rule vocabulary at the subword level is larger than in the word-level setting (25,714 vs. 5,000 in [9]). While every rule is observed in parallel data and therefore corresponds to a valid SentencePiece token sequence, some subword-level replacements may look linguistically unintuitive when viewed in isolation. In practice, plausibility is enforced by the model's context-dependent predictions and by decoding constraints of the tokenizer.

5.2.3 Training parameters

We used XLNet as the encoder since it showed the best results in [9]. We provided the training process with the same parameters except for the batch size. Table 2 shows the set parameters for the experiments.

The authors of [9] set batch size=256 for the I stage and batch size=128 for II, III stages. We used sizes 32 and 16 respectively. Table 2 shows the best epochs for each training stage of the XLNet model at the subword and word levels.

Table 2. The best epochs at each training stage of the XLNet model [9] (word-level) and in our research (subword-level) with appropriate batch sizes.

Training stage	Word-level		Subword-level	
	Epoch	Batch size	Epoch	Batch size
I	20	256	20	32
II	9	128	7	16
III	4	128	1	16

5.3 Results

Table 3 shows the results obtained on benchmarks CoNLL-2014 (test) and BEA-2019 (test) with the related utilities – M^2-scorer and ERRANT. On CoNLL-2014 benchmark, our subword-level model shows competitive $F_{0,5}$ = 62.5 compared to $F_{0,5}$ = 65.3 for word-level in [9]. Even our subword-level model slightly outperforms in terms of recall metric R = 40.4 compared to R = 40.1 or word-level. On BEA-2019 our model could not come close to the results of the original model, but still shows a strong $F_{0,5}$ = 61.9. As we can see the model has a relatively low recall. This can be explained by the fact that we did not use all synthetic data. However, we have been able to show that it is possible to achieve competitive results with a completely automatic approach. It is important to note that this approach is language-independent and can be adapted to any language with a subword vocabulary.

6. Conclusion

In this study, we have demonstrated a completely automatic approach for training Sequence Tagging models, generating data and constructing a correction rule vocabulary.

We have shown that in this approach we do not need to develop grammar-specific operations. The basic transformations – delete, keep, replace and append – are sufficient for error correction, since any word-level error can be represented as subword-level basic transformations. We presented a way to generate ground truth corrections using the Levenshtein algorithm and retrieving the editorial prescriptions.

Table 3. Comparison of subword-level models with word-level models. The M^2 score for CoNLL-2014 (test) and ERRANT for the BEA-2019 (test) are reported.

Model	CoNLL-2014 (test)			BEA-2019 (test)		
	P	R	$F_{0,5}$	P	R	$F_{0,5}$
GECToR (subword-level + XLNet)	72.3	40.4	62.4	70.5	41.6	61.9
GECToR (word-level + BERT)	72.1	**42.0**	63.0	71.5	**55.7**	67.6
GECToR (word-level + RoBERTa)	73.9	41.5	64.0	77.2	55.1	71.5
GECToR (word-level + XLNet)	**77.5**	40.1	65.3	**79.2**	53.9	**72.4**

We have shown how we can take into account the subword similarity to obtain optimal ground truth corrections. Thus, the dataset generation is completely automatic. This is especially important for low-resource and morphologically complex languages.

We applied our approach to the Sequence Tagging model GECToR and demonstrated that our approach shows comparable quality results for the English language while being completely automatic – $F_{0,5}$ of 62.4 on CoNLL-2014 (test) and $F_{0,5}$ of 61.9 on BEA-2019 (test).

In future works, we want to conduct experiments on adapting this approach for other languages. We also want to see if the quality of the models depends on the used transformer-based encoders with corresponding tokenizers.

References

[1]. Rozovskaya A., Roth D. Grammatical Error Correction: Machine Translation and Classifiers. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, 2016, pp. 2205-2215.

[2]. Yuan Z., Stahlberg F., Rei M., Byrne B., Yannakoudakis H. Neural and FST-based approaches to grammatical error correction. Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, Florence, Italy, 2019, pp. 228-239.

[3]. Bryant C., Ng H. How Far are We from Fully Automatic High Quality Grammatical Error Correction? Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 2015, pp. 697-707.

[4]. Patel H., Rajput D., Gadekallu T., Iwendi C., Bashir A., Jo H. A review on classification of imbalanced data for wireless sensor networks. International Journal of Distributed Sensor Networks, vol. 16, 2020.

[5]. Flickinger D., Yu J. Toward More Precision in Correction of Grammatical Errors. Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, Sofia, Bulgaria, 2013, pp. 68-73.

[6]. Yuan X., Pham D., Davidson S., Yu Z. ErAConD: Error Annotated Conversational Dialog Dataset for Grammatical Error Correction. Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, United States, 2022, pp. 76-84.

[7]. Lee J., Seneff S. An analysis of grammatical errors in non-native speech in english. 2008 IEEE Spoken Language Technology Workshop, 2008, pp. 89-92.

[8]. Ng H., Wu S., Briscoe T., Hadiwinoto C., Susanto R. Bryant C. Encode, The CoNLL-2014 Shared Task on Grammatical Error Correction. Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task, Baltimore, Maryland, 2014, pp. 1-14.

[9]. Omelianchuk K., Atrasevych V., Chernodub A., Skurzhanskyi O. GECToR – Grammatical Error Correction: Tag, Not Rewrite. Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, Seattle, WA, USA, 2020, pp. 163-170.

[10]. Khabutdinov I., Chashchin A., Grabovoy A., Kildyakov A., Chekhovich Yu. RuGECToR: Rule-Based Neural Network Model for Russian Language Grammatical Error Correction. Program. Comput. Softw., vol. 50, issue 4, 2024, pp. 315-321.

[11]. Malmi E., Krause S., Rothe S., Mirylenka D., Severyn A. Encode, Tag, Realize: High-Precision Text Editing. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 2019, pp. 5054-5065.

[12]. Rothe S., Mallinson J., Malmi E., Krause E., Severyn A. A Simple Recipe for Multilingual Grammatical Error Correction. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, 2021, pp. 702-707.

[13]. Grundkiewicz R., Junczys-Dowmunt M., Heafield K. Neural Grammatical Error Correction Systems with Unsupervised Pre-training on Synthetic Data. Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, Florence, Italy, 2019, pp. 252-263.

[14]. Rozovskaya A., Roth D. Grammar Error Correction in Morphologically Rich Languages: The Case of Russian. Transactions of the Association for Computational Linguistics, vol. 7, 2019, pp. 1--17.

[15]. Kudo T., Richardson J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Brussels, Belgium, 2018, pp. 66-71.

[16]. Limisiewicz T., Balhar J., Marecek D. Tokenization Impacts Multilingual Language Modeling: Assessing Vocabulary Allocation and Overlap Across Languages. Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, 2023, pp. 5661-5681.

[17]. Asvarov A., Grabovoy A. The Impact of Multilinguality and Tokenization on Statistical Machine Translation. 35th Conference of Open Innovations Association (FRUCT), Tampere, Finland, 2024, pp. 149-157.

[18]. Liu Y., Ott M., et. al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019. Cite arxiv:1907.11692.

[19]. Devlin J., Chang M., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, Minnesota, 2019, pp. 4171-4186.

[20]. Yang Z., Dai Z., Yang Y., Carbonell J., Salakhutdinov R., Le Q. XLNet: generalized autoregressive pretraining for language understanding. Proceedings of the 33rd International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 2019.

[21]. Dahlmeier D., Ng H. Better Evaluation for Grammatical Error Correction. Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Montreal, Canada, 2012, pp. 568-572.

[22]. Awasthi A., Sarawagi S., et. al. 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), Hong Kong, China, 2019, pp. 4259-4269.

[23]. Mizumoto T., Komachi M., Nagata M., Matsumoto J. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. Proc. of 5th International Joint Conference on Natural Language Processing, 2011, pp. 147-155.

[24]. Dahlmeier D., Ng H., Wu S. Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English. Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications, Atlanta, Georgia, 2013, pp. 22-31.

[25]. Yannakoudakis H., Briscoe T., Medlock B. A New Dataset and Method for Automatically Grading ESOL Texts. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, 2011, pp. 180-189.

[26]. Bryant C., Felice M., et. al. The BEA-2019 Shared Task on Grammatical Error Correction. Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, Florence, Italy, 2019, pp. 52-75.

[27]. Bryant C., Felice M., Briscoe T. Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics,Vancouver, Canada, 2017, pp. 793-805.

Информация об авторах / Information about authors

Ильдар Айратович ХАБУТДИНОВ — аспирант Московского физико-технического института (МФТИ), старший разработчик GigaChat в ПАО «Сбербанк». Сфера научных интересов: обработка естественного языка, большие языковые модели, методы обучения и дообучения нейросетевых моделей, методы выравнивания языковых моделей, распределённое обучение.

Ildar Airatovich KHABUTDINOV — PhD student at the Moscow Institute of Physics and Technology (MIPT), Senior Developer in the GigaChat team at Sber. Research interests: natural language processing, large language models, neural network training and fine-tuning methods, language model alignment methods, distributed training.

Андрей Валериевич ГРАБОВОЙ – кандидат физико-математических наук, старший научных сотрудник лаборатории № 42 «Интеллектуального анализа данных» Института проблем управления имени В. А. Трапезникова РАН с 2025 года, доцент кафедры «Интеллектуальных систем» МФТИ (НИУ). Сфера научных интересов включает выбор моделей глубокого обучения, априорные распределения гиперпараметров, дистилляцию знаний, обработку естественного языка и статистические методы обработки информации.

Andrey Valerievich GRABOVOY – Cand. Sci. (Phys.-Math,), senior scientist of the Laboratory 42 of Data Mining at the V.A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences since 2025, Associate Professor at the Department of Intelligent Systems, Moscow Institute of Physics and Technology (National Research University). Research Interests: include pattern recognition, prior distributions of model parameters, model distillations, nlp.

Юрий Викторович ЧЕХОВИЧ – кандидат физико-математических наук, заведующий лабораторией № 42 «Интеллектуального анализа данных» Института проблем управления им. В.А. Трапезникова РАН с 2025 года. Сфера научных интересов: машинное обучение, искусственный интеллект, обработка естественных языков, распознавание образов, высоконагруженные прикладные системы.

Yury Viktorovich CHEKHOVICH – Cand. Sci (Math). Head of the Laboratory 42 of Data Mining at the V.A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences since 2025. Research Interest: machine learning, artificial intelligence, natural language processing, pattern recognition, high-load applications.

Александр Сергеевич КИЛЬДЯКОВ – научный сотрудник лаборатории №42 «Интеллектуального анализа данных» Института проблем управления им. В.А. Трапезникова РАН с 2025 года. Сфера научных интересов: машинное обучение, искусственный интеллект, обработка естественных языков, распознавание образов, высоконагруженные прикладные системы.

Alexandr Sergeevich KILDYAKOV – Scientist of the Laboratory 42 of Data Mining at the V.A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences since 2025. Research Interest: machine learning, artificial intelligence, natural language processing, pattern recognition, high-load applications.

Андрей Александрович ИВАХНЕНКО – кандидат физико-математических наук, младший научный сотрудник лаборатории №42 «Интеллектуального анализа данных» Института проблем управления им. В.А. Трапезникова РАН с 2025 года. Сфера научных интересов: снижение сложности информационных систем, машинное обучение, искусственный интеллект, обработка естественных языков, распознавание образов, высоконагруженные прикладные системы.

Andrey Alexandrovich IVAKHNENKO – Cand. Sci. (Phys.-Math,), Scientist of the Laboratory 42 of Data Mining at the V.A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences since 2025. Research interests: optimization of information systems architecture, machine learning, artificial intelligence, natural language processing, pattern recognition, high-load applications..