Russian Generative Spelling, Punctuation and Capitalization Correction

Nikita Martynov[1], **Danil Astafurov**[2], **Ulyana Isaeva**[1], **Ivan Vasil'yevich Maksimov**[3]
Joqsan Azocar[3], **Dmitrii Kosenko**[3], **Alena Fenogenova**[1]

[1]SberAI, [2]ITMO University,[3]MIPT
Correspondence: alenush93@gmail.com

Abstract

This paper presents **SAGE**, an open-access framework that encloses a set of models specifically designed for the generative correction of spelling, punctuation, and capitalization errors in Russian. The release includes four models, featuring a Russian-English version and a distilled version for easy use and cost-effectiveness. The models are pre-trained using a sequence-to-sequence approach on artificial errors that mimic human mistakes and fine-tuned on annotated multi-domain texts. A set of carefully engineered auxiliary learning objectives is employed during pre-training to enrich the models with additional semantic and syntactic information. Evaluations indicate that SAGE models, despite having a small number of parameters, outperform top-tier multilingual and Russian-specific large language models, including both closed- and open-source options, and are considered state-of-the-art. We release the online demo powered by a single Nvidia A100 80GB GPU as a Web service, which allows to simultaneously test the most advanced SAGE model of 1.7B parameters, its distilled version and the Russian-English SAGE model.

Keywords: Generative LLMs, Spellchecking, Benchmark

1. Introduction

The automatic error correction (EC) task is fundamental to a wide range of text-based applications, with an example of recently emerged interactive smartphone keyboards (Liu et al., 2024). EC has been a longstanding focus within the field of natural language processing since the pioneering works of Damerau (1964) and Levenshtein (1966). However, despite being thoroughly studied, the task still poses a significant challenge even for state-of-the-art (SOTA) generative large language models (LLMs) (Fang et al., 2023). The difficulty of the task stems from the *differing possibilities for correction* and understanding of the *communicative intention* of the writer (see Bryant et al. (2023)), as well as the increasing variety of errors. In this regard, EC in Russian presents additional complexity due to its intricate syntactic structure, rich morphological system, and flexible word order (Sorokin et al., 2016; Shavrina et al., 2016).

To address these challenges, various solutions have been proposed, particularly with significant advancements in SOTA LLMs and prompting techniques designed to correct errors in input text (Wu et al., 2023; Fang et al., 2023; Coyne et al., 2023). While this prompting approach shows promise, it has two main limitations: (i) it is highly dependent on the clarity of the instructions provided (Liu et al., 2023), and (ii) it often overcorrects rare words (Fang et al., 2023), mistakenly identifying them as errors. Furthermore, the most effective open source LLMs have tens of billions of parameters, while the APIs for proprietary models are often limited in access and can exhibit performance fluctuations and raise privacy concerns (Chen et al., 2023, 2024). These factors significantly hinder the practical application of the prompting approach for error correction in daily usage.

To address these limitations, we present a family of models that jointly perform Spelling EC (SEC), Punctuation EC (PEC), and Capitalization EC (CEC) tasks in a plain sequence-to-sequence fashion, with input represented by corrupted text without no instruction. Our encoder-decoder solutions utilize the FRED-T5 (Zmitrovich et al., 2024) architecture for Russian and include its 95M, 820M, and 1.7B variants, alongside the mT5 model (Xue et al., 2021) for the multilingual version. These models are pre-trained and fine-tuned on diverse natural and synthetic error data, which allows them to effectively identify a wide range of mistakes.

To encourage the use of SAGE in environments with limited computing resources, we have distilled the best model for Russian down to just 95 million parameters. This distilled version performs comparably to GPT-4o [1] for EC in Russian and can be operated on a CPU.

The contributions of the presented work are summarized as follows:

- We open source the family of SAGE models[2], jointly providing spelling, punctuation, and capitalization normalization, as well as the training code for auxiliary tasks, pre-training, fine-tuning, and validation procedures. Both are being distributed under the MIT license.

[1] openai.com/index/hello-gpt-4o/
[2] HuggingFace: ai-forever/sage

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 9861–9872
11–16 May 2026.

- We conduct a comprehensive evaluation of the proposed solution. Although they are not computationally demanding, our models outperform open-source and proprietary multilingual and Russian-specific LLMs, alongside open-source tools with regard to SEC, PEC, and CEC tasks in Russian.
- As the SAGE release includes correction for capitalization and punctuation, we also expanded the open benchmark introduced in (Martynov et al., 2024) to account for punctuation and capitalization errors.
- The SAGE models are available for direct use from the library. We also introduce an open-access web demo service (see Figure 1) designed for user-friendly interaction. It features three models: the best Russian, multilingual, and distilled options. The Web demo is available at huggingface.co/spaces/ai-forever/sage and its interface is demonstrated in Figure 1.

2. Related Work

Research on SEC, PEC, and CEC in Russian is limited, with most of the work focusing on SEC (Panina et al., 2013; Sorokin, 2017). The SpellRuEval competition introduced RUSpellRU, a corpus of naturally misspelled texts, while Martynov et al. (2023) presented MultidomainGold, the first multidomain SEC dataset. Subsequent improvements by Martynov et al. (2024) incorporated two additional corpora, GitHubTypoCorpusRu and MedSpellCheck, achieving SOTA with M2M100 (Fan et al., 2021) and FRED-T5.

Recent approaches frame EC tasks as instruction-based problems, leveraging zero-shot LLM prompting. For Russian, options include open-source Russian-specific LLMs (e.g. T-lite [3]), open-source multilingual models like Llama (Touvron et al., 2023; Dubey et al., 2024) and Qwen (Bai et al., 2023; Yang et al., 2024), and proprietary models (GPT-4o, Claude [4], Gemini [5]).

Specialized tools for the Russian SEC include JamSpell, HunSpell and Yandex.Speller[6], while ORFO, Orfogrammka, and retext.ai[7] offer combined SEC, PEC, and CEC as proprietary services.

[3] T-lite-instruct-0.1

[4] anthropic.com/claude

[5] ai.google.dev/models/gemini

[6] yandex.ru/speller

[7] online.orfo.ru, orfogrammka.ru and retext.ai respectively

3. Datasets

We employ two respective types of datasets for pre-training and fine-tuning three FRED-T5 models and mT5 on synthetically generated and natural errors.

3.1. Pre-train Datasets

Following the work of Martynov et al. (2024), the Russian pre-training corpus for all models is assembled from equal parts of Russian Wikipedia and transcribed videos[8], comprising a total of 3.5 million samples each. Texts under 40 characters and those containing non-alphabetic symbols are removed. The remaining texts are then processed employing the Augmentex corruption algorithm proposed in Martynov et al. (2023) with the default parameters. The resulting pre-training dataset consists of 6.6 million pairs of corrupted source texts and their corresponding corrections. To train the Russian-English mT5 model, we additionally collected 6 GB of raw texts, equally split among the News section of the Leipzig Corpora[9] and an English Wikipedia dump, and then applied the same pre-processing steps and corruption algorithm used for the Russian portion. The final corpus for the mT5 model accounts for nearly 25 million text pairs. See Table 1 for additional dataset statistics.

3.2. Fine-tune Datasets

This work employs a combination of training splits of RUSpellRU and MultidomainGold datasets for the fine-tuning procedure of three Russian models, as these corpora represent all available manually annotated training sets for EC in Russian and contain mistakes from a total of six textual domains. In order to enable the model to correct punctuation and capitalization errors alongside common misspellings, both splits have been additionally labeled for PEC and CEC tasks.

RUSpellRU and MultidomainGold already contained error corrections, yet only for the SPELL category, i.e., without capitalization and punctuation corrections. To fulfill this gap, we conducted an additional annotation step. The task involved making corrections in cases of missed or misplaced punctuation marks or incorrect capitalization. The annotation was performed by 7 professional editors chosen during a task-specific qualification step, without expert overlap, i.e., with only one answer per sample.

[8] UrukHan/t5-russian-spell_I

[9] wortschatz.uni-leipzig.de/en/download/

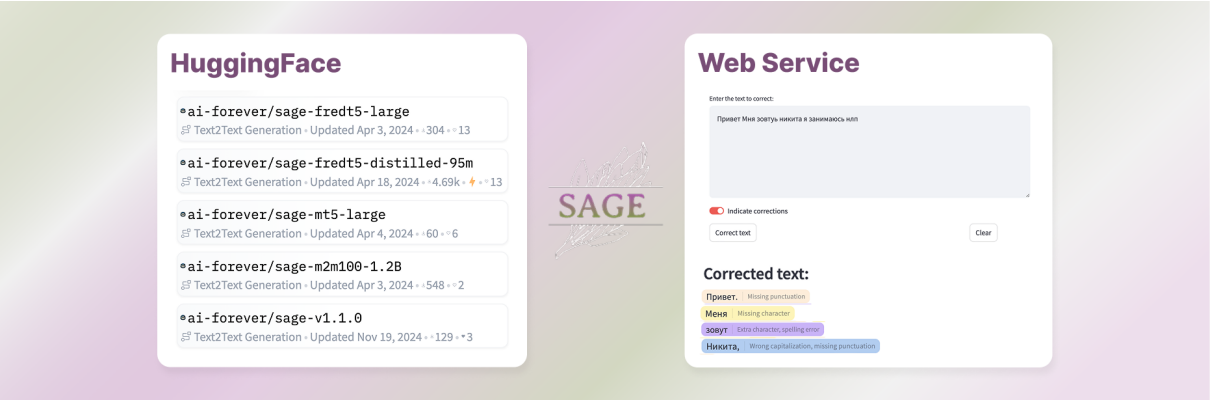

Figure 1: The models are available via Web demo and downloads from HuggingFace.

The experts annotated approximately 12K samples [10]. The annotation was conducted on the ABC Elementary[11] platform, and the task layout example is demonstrated in Figure 2.

In line with previous works on EC for English (Muralidhar Jayanthi et al., 2020), we do not fine-tune the mT5 model on English datasets containing natural errors and instead only employ it for testing.

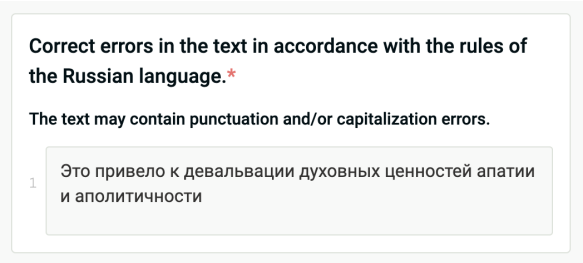

Figure 2: Annotation task layout.

3.3. Evaluation datasets

We use test splits of RUSpellRU and Multidomain-Gold datasets alongside annotated in Martynov et al. (2024) Russian parts of MedSpellCheck and GitHubTypoCorpusRu corpora to comprehensively evaluate the proposed solutions for Russian. To test the ability of our model to correct punctuation and capitalization errors, the samples in four datasets have undergone additional annotation for these types of mistakes via the same annotation pipeline as the fine-tuning datasets. We also employ BEA60K (Muralidhar Jayanthi et al., 2020) and JFLEG (Napoles et al., 2017) datasets for mT5 evaluation.

[10] in 211 hours, with a total annotation cost of $3149 (approximately $14.9 hourly compensation, which exceeds the minimum wage per hour in Russia)

[11] elementary.center/

4. Models Training

All models share the same training pipeline, except for mT5, which is not fine-tuned on data containing natural errors.

4.1. Models Selection

Generative SEC is shown (Bout et al., 2023; Rothe et al., 2021) to significantly outperform alternative approaches to SEC, including classification- and statistics-based algorithms. Considering this observation, we implemented the resulting solution in a generative sequence-to-sequence fashion with mT5 and the only Russian-specific encoder-decoder architecture FRED-T5. Although recent trend dictates employing decoder-based architectures and formulating the task in an instruction-following manner, we have observed that the decoder model fails to deliver the same level of performance as FRED-T5 for the Russian language. For the Russian-specific decoder we compared FRED-T5 with mGPT-1.3B (Shliazhko et al., 2024) of 1.3B parameters, which is closest comparable capacity that does not exceed that of our most advanced model (1.7B). We selected mT5-large (Xue et al., 2021) of 1.2B parameters for multilingual encoder-decoder given the architecture of FRED-T5 is based on that of T5 (Raffel et al., 2020) (see Zmitrovich et al. (2024)) and it has similar capacity. We evaluated mGPT-1.3B, mT5-large and three variants of FRED-T5 on four test datasets from Section 3.3, the results are available in Table 2.

4.2. Auxiliary Tasks

For the training pipeline, we employ two auxiliary objectives: (i) identifying types of errors in a source text and (ii) predicting the correct counterpart for the token in an original sentence, which are denoted further in the text as Error Type Recognition (ETR) and Correct Token Prediction (CTP), respectively. Both tasks induce cross-entropy loss function.

	Dataset	Size	Num. tokens	Avg. length	Num. domains	Avg. #errors	Manual annotation
	Pre-training (RU)	6.611.990	633.444.131	95	2	56	No
	Pre-training (EN)	24.769.202	1.277.508.738	52	2	31	No
SFT	RUSpellRU	2.000	37.373	18	1	3	Yes
SFT	MultidomainGold	3.569	170.617	47	5	8	Yes
Evaluation	RUSpellRU	2.008	37.953	19	1	6	Yes
Evaluation	MultidomainGold	4.106	187.836	46	7	6	Yes
Evaluation	MedSpellChecker	1.054	38.047	36	1	10	Yes
Evaluation	GitHubTypoCorpus	868	37.349	43	1	8	Yes
Evaluation	BEA60K	63044	1.549.778	25	4	1,11	Yes
Evaluation	JFLEG	1601	48.897	31	1	1,25	Yes

Table 1: Datasets statistics. **Size** is a number of samples in a corpus, **Num. tokens** reflects total number of tokens in a set for a FRED-T5-1.7B tokenizer and mT5-large tokenizer for English sets (only erroneous sentences), **Avg. length** and **Avg. #errors** correspond to average number of tokens and errors per text, respectively, **Num. domains** represents the number of source domains in a corpus, and **Manual annotation** represents whether the samples are manually labeled for SEC, PEC and CEC tasks. **SFT** and **Evaluation** aggregate datasets described in Sections 3.2 and 3.3 respectively.

		RUSpellRU						MultidomainGold						MedSpellcheck						GitHubTypoCorpusRu					
	Name	Pr. S	Rec. S	F1 S	Pr. P	Rec. P	F1 P	Pr. S	Rec. S	F1 S	Pr. P	Rec. P	F1 P	Pr. S	Rec. S	F1 S	Pr. P	Rec. P	F1 P	Pr. S	Rec. S	F1 S	Pr. P	Rec. P	F1 P
pre-train	Fred																								
pre-train	95M	52.6	62.0	56.9	83.7	21.4	34.1	35.4	45.6	39.9	29.6	22.6	25.6	26.7	46.0	33.8	12.4	6.3	8.4	39.7	43.0	41.3	27.0	18.3	21.8
pre-train	820M	65.5	65.9	65.7	84.4	23.8	37.1	47.8	50.6	49.2	33.5	22.4	26.8	46.1	52.5	49.1	36.2	16.2	22.4	52.0	47.6	49.7	29.3	16.1	20.8
pre-train	1.7B	71.8	66.5	69.0	85.0	30.1	44.5	49.6	51.4	50.5	34.6	24.6	28.8	38.6	51.9	44.3	23.7	11.5	15.5	52.9	50.0	51.4	32.7	15.7	21.2
pre-train	mgpt	59.6	67.7	63.4	83.8	36.8	51.1	38.3	41.9	40.0	18.4	14.2	16.0	28.0	50.3	36.0	17.0	8.8	11.6	43.8	48.2	45.9	21.4	15.8	18.2
pre-train	mt5	58.6	64.7	61.5	88.0	27.9	42.4	42.1	47.0	44.4	41.1	29.1	34.1	40.8	52.1	45.8	43.6	30.7	36.0	53.0	50.0	51.4	37.9	20.3	26.4
fine-tune	Fred																								
fine-tune	95M	80.0	71.0	75.2	85.5	78.4	81.8	75.5	67.4	71.3	65.9	61.3	63.5	49.2	63.5	55.4	74.6	62.2	67.8	51.6	46.0	48.6	41.2	34.2	37.3
fine-tune	820M	88.1	82.2	85.1	89.0	85.3	87.1	79.8	75.7	77.7	69.7	67.0	68.3	77.0	72.2	74.5	74.7	67.7	71.0	67.6	54.1	60.1	49.9	39.2	43.9
fine-tune	1.7B	90.0	85.5	87.7	89.7	86.7	88.2	80.5	77.5	79.0	70.4	67.3	68.8	74.8	73.8	74.3	74.9	69.0	71.8	69.8	57.0	62.8	50.2	37.0	42.6
fine-tune	mgpt	82.8	78.4	80.6	84.8	48.8	62.0	62.1	51.4	56.2	48.7	32.3	38.8	65.3	66.7	66.0	70.8	43.9	54.2	62.3	51.8	56.6	35.7	32.2	33.9
fine-tune	mt5	84.7	79.0	81.8	86.9	82.6	84.7	74.8	70.6	72.6	64.8	62.8	63.8	69.2	66.3	67.7	76.5	67.4	71.7	63.8	53.3	58.0	49.7	41.9	45.5

Table 2: Evaluation scores of models **pre-trained** and pre-trained then **fine-tuned** without auxiliary tasks.**RUSpellRU**, **MultidomainGold**, **MedSpellcheck** and **GitHubTypoCorpusRu** refer to the test splits of corresponding datasets. **Pr**ecision, **Rec**all and **F1** are described in Section 5.1 and separately estimate **S**pelling and **P**unctuation corrections. Best values in column with regard to segment (pre-train or fine-tune) are underlined. FRED-T5-1.7B dominates its counterparts in fine-tune segment and as soon as the pipeline of the final solution includes fine-tuning on natural error data, FRED-T5-1.7B is better option for the backbone model for **SAGE** service with regard to its **fine-tune** performance.

Error type recognition. Given a sequence of tokens $T = [t_1, ..., t_{N_T}]$, a finite and fixed (for all sequences T) set of all possible error types $E = \{e_1, ..., e_{N_E}\}$, and a non-injective correspondence $f_T : T \to E$, the task is to predict $f_T(t_i), t_i \in T$, which is formulated in the form of a multilabel classification loss function

$$L_{ETR}^{t_i} = -\sum_{e \in E} [y_e^{t_i} p_e^{t_i} + (1 - y_e^{t_i})(1 - p_e^{t_i})] \quad (1)$$

where $p_e^{t_i}$ is the probability of error type $e \in E$ being injected in token t_i, $y_e^{t_i} = \mathbb{1}_{\{e \in f_T(t_i)\}}$. The objective for the whole sequence T is given by $L_{ETR} = \sum L_{ETR}^{t_i}$ over $t_i \in T$.

Correct token prediction. With the source sentence $\mathbf{s}$ represented as a set $S = \{[t, i_l, i_r]_j\}_{j=1}^{N_T}$, i_l, i_r are the left and right indices of t in $\mathbf{s}$, the respective correction $\mathbf{c}$ is split into an ordered collection of tokens $C = [c_1, ..., c_{N_T}]$ and the following objective is minimized:

$$L_{CTP} = -\sum_{j=1}^{N_T} \mathbb{1}_{\{c_j \neq \varnothing\}} \log p_{c_j}(t_j) \quad (2)$$

, where $p_{c_j}(t_j)$ is a probability of c_j being a correct counterpart for the source token t_j.

4.3. Training

We first train FRED-T5-820M, FRED-T5-1.7B, and mT5-large on the pre-training dataset described in Section 3.1 with two auxiliary objectives from Section 4.2 on 8 Nvidia H100 for *five epochs* and 95, 192, and 293 GPU hours, respectively. Following (Zmitrovich et al., 2024) and (Xue et al., 2021), we keep the *Adafactor optimizer* (Shazeer and Stern, 2018) and constant *learning rate* of $1 \cdot 10^{-3}$ throughout the pre-training with the total *batch size* of 1024 samples. The fine-tuning stage for FRED-T5-820M and FRED-T5-1.7B involves corpora from Section 3.2 and is performed with the total *batch size* of 32 and *linear scheduler* that decreases learning rate from $1 \cdot 10^{-3}$ to zero. At the pre-training stage, we optimize the total objective of $L_{final} = L_{CTP} + L_{ETR} + L_{seq2seq}$, where $L_{seq2seq}$ is a cross-entropy loss. During fine-tuning, only $L_{seq2seq}$ is minimized.

4.4. Distillation

To prepare the student, we reproduced language modeling pre-training of FRED-T5-1.7B from Zmitrovich et al. (2024) for flan-t5-small[12] (Chung et al., 2024) architecture, for which we employed original FRED-T5-1.7B's tokenizer and embedding layer.

FRED-T5-95M has been obtained specifically for the purposes of this work. The architecture is analogous to flan-t5-small and consists of 8 transformer layers with a hidden size of 1024. A byte-level Byte-Pair Encoding (BPE) tokenizer is utilized, featuring a vocabulary size of 50,257 tokens supplemented by 107 special tokens. These special tokens include task-specific prefixes such as `<LM>`, `<SC1>`, up to `<SC6>`, which guide the model during different training tasks.

The training corpus comprises a substantial 300 GB of Russian text. Training was conducted using a mixture of seven denoising tasks akin to the UL2 (Tay et al., 2023) framework, but with several modifications tailored to the model's objectives. Notably, during the first half of the training process, the model was trained on a limited subset of the dataset (approximately 1%, equating to 3 GB) and without incorporating task-specific prefixes. The model has been pre-trained for three epochs with batch size of 512 samples, Adafactor optimizer (Shazeer and Stern, 2018), learning rate of 0.01 and sequence length of 512 tokens for both inputs and generated texts. The total training duration spanned approximately 7 days, leveraging the computational power of 8 Nvidia A100 80GB GPUs.

During the pre-training and fine-tuning of the student model on error data, we used datasets from Sections 3.1 and 3.2 respectively, auxiliary tasks described in Section 4.2, and similar hyperparameters to those employed while pre-training and fine-tuning of FRED-T5-1.7B, with the only exception of the total *batch size* being 8912 samples for pre-training and 64 for fine-tuning. The additional objective of Kullback-Leibler divergence between the logits of the last layers of the teacher and student models was added, resulting in the total loss function $L_{distill} = L_{CTP} + L_{ETR} + L_{seq2seq} + L_{KL}$, which the student model minimized at each stage.

4.5. Model Optimization

The throughput of the trained FRED-T5-1.7B, which is the largest model in the family, was additionally increased to 317 tokens per second (11.4 times) through a native PyTorch implementation and the use of `torch.compile`. The model preparation process involves allocating memory for caching, capturing and compiling the computational graphs of both encoder and decoder, and warming up the model to force Just-In-Time (JIT) compilation before performing any inference tasks.

5. Evaluation

5.1. Metrics

The metrics used in the experiments calculate precision, recall and F1 score between predicted and gold corrections. They are extracted from a tuple ‹*source*, *correction*, *gold*›, where predicted and gold corrections are a list of *source* modifications in *correction* and *gold*, respectively. We use two metrics that differ in the algorithm for the extraction of the modifications.

SpellRuEval (Sorokin et al., 2016) extracts word-level corrections from a given pair of texts in a single list, thus the overall metric produces global precision, recall, and F1 scores. The SpellRuEval is case- and yofication-insensitive[13] (i.e. does not annotate such corrections) and does not extract punctuation modifications.

RuERRANT[14] (Katinskaia et al., 2022) is a Russian-language adaptation of the original grammatical error annotation toolkit ERRANT (Bryant et al., 2017). We further developed this algorithm to account for finer-grained mistakes common for Russian, which include yofication, punctuation, and capitalization errors.

5.2. Human evaluation

Additionally, we evaluated the quality of FRED-T5-1.7B corrections through expert assessment by professional philologists and linguists. Experts assigned binary scores: 1 for corrections requiring no manual amendments and 0 otherwise. This task emulates end-user experience with text correction tools embedded in messaging platforms and web applications. Annotation was conducted with three-way overlap following the instructions provided in Appendix B.

5.3. Evaluation details

All SEC scores reported in this paper are calculated using **SpellRuEval**, so the reported results are comparable with those reported in Martynov et al. (2024). Punctuation and capitalization corrections are extracted with modified **RuERRANT**. For the evaluation of LLMs, we report only results

[12] google/flan-t5-small

[13] Yofication (rus. ёфикация) refers to optional use of the letter "ё" instead of "е"; in Russian both spellings are acceptable, but some style guides require "ё". This is a separate metric from orthography, since writing "е" for "ё" is not an orthographic error.

[14] `Askinkaty/errant`

	Model	RUSpellRU			MultidomainGold			MedSpellcheck			GitHubTypoCorpusRu		
		Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Open API	**Yandex.Speller**	83.0	59.8	69.5	52.9	51.4	52.2	**80.6**	47.8	60.0	67.7	37.5	48.3
	Hunspell	31.3	34.9	33.0	16.2	40.1	23.0	10.3	40.2	16.4	28.5	30.7	29.6
	JamSpell	42.1	32.8	36.9	25.7	30.6	28.0	24.6	29.7	26.9	49.5	29.9	37.3
Proprietary Russian	**GigaChat Max**												
	Spelling	56.6	75.7	64.8	46.6	70.5	56.1	25.1	55.8	34.6	41.6	51.9	46.2
	Punctuation	73.4	75.3	74.3	49.4	52.4	50.9	41.7	45.9	43.7	33.8	27.9	30.5
	Capitalization	88.4	88.6	88.6	63.4	75.9	69.1	37.6	56.7	45.2	32.6	**56.6**	41.4
	GigaChat Pro												
	Spelling	49.1	71.9	58.4	41.7	69.8	52.2	30.8	60.3	40.8	33.2	50.2	39.9
	Punctuation	77.5	77.2	77.4	46.7	51.3	48.9	59.0	58.0	58.5	29.6	29.8	29.7
	Capitalization	88.4	84.4	86.4	61.0	73.6	66.7	64.0	79.7	71.0	29.9	54.7	38.7
	YandexGPT-4 Pro												
	Spelling	67.0	82.7	74.0	54.8	76.0	63.7	50.2	73.8	59.7	58.0	56.6	57.3
	Punctuation	62.7	69.3	65.8	36.5	51.7	42.8	47.0	43.9	45.4	42.0	37.2	39.5
	Capitalization	88.0	91.0	89.5	63.8	74.7	68.8	70.6	82.8	76.2	42.9	**56.6**	**48.8**
Open-source Russian	**T-lite-instruct-0.1**												
	Spelling	10.9	44.5	17.5	7.0	42.8	12.0	2.6	25.4	4.7	7.7	34.3	12.6
	Punctuation	47.4	59.9	52.9	10.0	31.2	15.2	31.3	45.2	37.0	6.7	18.9	9.9
	Capitalization	63.4	23.1	33.9	18.7	32.2	23.6	20.0	33.0	24.9	6.9	26.4	10.9
	Vikhr-8B-instruct												
	Spelling	13.1	38.5	19.5	16.6	46.7	24.5	5.7	30.7	9.6	15.1	37.4	21.5
	Punctuation	60.6	65.7	63.0	24.3	45.1	31.6	45.0	50.3	47.5	12.1	18.8	14.7
	Capitalization	70.8	19.5	30.6	58.0	47.7	52.4	51.0	29.9	37.7	19.7	22.6	21.0
	Vikhr-12B-instruct												
	Spelling	13.3	38.9	19.8	16.7	46.9	24.6	5.7	30.7	9.6	15.3	37.5	21.7
	Punctuation	61.7	66.5	64.0	24.4	45.3	31.8	45.4	50.4	47.8	12.2	18.9	14.9
	Capitalization	70.7	19.4	30.4	58.3	47.8	52.5	54.5	30.3	38.9	21.1	24.5	22.7
	Saiga-Llama3-70B												
	Spelling	54.4	65.4	59.4	54.8	63.8	58.9	42.5	57.1	48.7	52.0	52.6	52.3
	Punctuation	76.8	73.0	74.9	61.1	50.3	55.2	76.4	59.0	66.6	43.0	24.7	31.4
	Capitalization	90.4	80.5	85.2	74.3	67.6	70.8	70.2	46.0	55.6	31.4	30.2	30.8
Open-source Multilanguage	**Llama-3.1-405B**												
	Spelling	55.1	76.1	63.9	45.2	71.8	55.5	30.7	65.4	41.8	48.6	55.4	51.8
	Punctuation	67.4	72.6	69.9	43.1	55.0	48.4	56.1	59.5	57.7	44.7	31.6	37.0
	Capitalization	89.1	87.3	88.2	68.7	67.3	68.0	67.3	57.5	62.0	32.8	37.7	35.1
	Llama-3.1-8B												
	Spelling	24.9	46.1	32.3	27.8	48.5	35.4	16.0	43.5	23.4	25.5	38.9	30.8
	Punctuation	64.4	56.0	59.9	37.0	40.4	38.6	60.0	53.2	56.4	23.2	25.7	24.4
	Capitalization	85.6	64.1	73.3	65.9	61.2	63.5	55.0	48.7	51.6	25.0	34.9	29.1
	Llama-3.2-1B												
	Spelling	3.4	6.8	4.5	11.0	15.2	12.7	4.1	10.3	5.9	5.8	8.7	7.0
	Punctuation	31.7	31.7	31.7	20.2	20.0	20.1	50.1	36.3	42.1	7.1	13.4	9.3
	Capitalization	65.7	33.8	44.6	26.4	21.4	23.6	27.6	23.0	25.1	6.9	19.8	10.2
	Qwen-32B-instruct												
	Spelling	42.6	64.2	51.3	30.6	61.4	40.8	29.9	61.5	40.3	34.9	48.3	40.5
	Punctuation	67.3	71.0	69.1	49.2	49.9	49.6	62.8	51.6	56.7	30.9	23.4	26.7
	Capitalization	89.2	84.9	87.0	70.5	62.5	66.3	70.4	67.4	68.9	32.1	34.0	33.0
	Gemma-2-27B												
	Spelling	54.4	71.6	61.8	48.2	68.5	56.6	46.7	65.9	54.7	50.8	54.4	52.5
	Punctuation	78.8	73.8	76.2	39.1	51.6	44.5	60.2	52.6	56.1	32.9	33.9	33.4
	Capitalization	89.7	94.5	92.0	69.5	74.9	72.1	64.8	74.7	69.4	39.0	50.0	43.8
Open AI	**GPT-4o**												
	Spelling	74.5	**87.8**	80.6	65.9	**79.1**	71.9	53.2	77.3	63.0	61.2	**58.6**	59.9
	Punctuation	71.6	74.9	73.2	55.8	58.2	57.0	73.4	66.7	69.8	42.4	26.2	32.4
	Capitalization	92.0	92.8	92.4	76.2	70.9	73.5	73.1	75.1	74.1	**43.0**	46.2	44.6
SAGE	**FRED-T5-1.7B**												
	Spelling	**91.1**	87.0	**89.0**	**82.1**	77.7	**79.8**	73.8	73.1	73.5	**70.8**	55.2	**62.1**
	Punctuation	**89.5**	**86.8**	**88.1**	**70.4**	**68.0**	**69.2**	74.8	**69.5**	**72.0**	50.3	**39.6**	**44.3**
	Capitalization	95.2	**95.9**	**95.6**	**80.5**	**80.5**	**80.5**	**80.9**	72.8	76.6	32.9	45.3	38.1
	FRED-T5-820M												
	Spelling	89.0	81.7	85.2	81.1	74.9	77.9	72.1	69.1	70.6	68.3	53.2	59.8
	Punctuation	88.4	84.6	86.5	69.3	67.0	68.2	73.5	65.9	69.5	**52.1**	38.0	43.9
	Capitalization	**95.5**	94.0	94.7	78.6	80.0	79.3	79.3	**85.1**	**82.1**	37.3	50.0	42.7
	FRED-T5-95M-distilled												
	Spelling	83.5	74.8	78.9	77.2	69.9	73.4	65.1	64.8	64.9	57.8	48.5	52.7
	Punctuation	86.8	80.6	83.6	66.8	63.4	65.0	**78.6**	63.1	70.0	45.2	39.5	42.1
	Capitalization	94.4	92.5	93.5	76.8	79.1	77.9	63.5	74.7	68.7	29.9	46.2	36.3
	mT5-large												
	Spelling	88.4	71.6	79.1	65.3	62.7	63.9	77.7	**77.5**	**77.6**	69.5	46.0	55.3

Table 3: Evaluation results of error correction systems. Solutions that are able to perform SEC task only are evaluated only for spelling errors (see Open API section). The best result in each column is highlighted in bold, the second best result is underscored. The best results are ranked within the respective type of errors. Within the SAGE family, FRED-T5-820M and mT5-large (Russian-English) are included; mT5-large was not fine-tuned on natural punctuation and capitalization errors.

Model	BEA60K			JFLEG		
	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4o	70.4	88.8	78.6	75.9	88.6	81.8
mT5-large	64.7	83.8	73.0	74.9	88.4	81.1

Table 4: Evaluation metrics of mT5-large. GPT-4o is provided for reference. Although mT5-large is of only 1.2B parameters, it matches the results of GPT-4o on JFLEG and still maintains decent performance on BEA60K dataset.

that correspond to the best-performing prompts derived by experimentation. For human evaluation, we employed a sample of 300 error-containing examples evenly distributed across the test datasets listed in Section 3.3).

6. Results and Discussion

We provide the evaluation results in Table 3. Within each column of the table, bold indicates the best result and underlining indicates the second best, ranked separately for Spelling, Punctuation, and Capitalization rows. Additional results are shown in Table 4.

FRED-T5-1.7B achieves the best results among SAGE models and outperforms both open- and closed-source LLMs and open-access services for EC in Russian with respect to most of the metrics and scopes of errors (spelling, punctuation, and capitalization), additionally producing corrections that the end-user does not need to amend in 65% of the cases, according to the results of experts evaluation (see Section 5.2). The distilled version (FRED-T5-95M-distilled) is offered for resource-constrained deployment; its metrics match those of GPT-4o. Although the remarkable performance of our solutions on the RUSpellRU and MultidomainGold test sets can be attributed to the use of the training splits of these corpora during the fine-tuning stage, SAGE models still exceed nearly all counterparts on the unseen test parts of the MedSpellCheck and GitHubTypoCorpusRu datasets. In addition, FRED-T5-95M-distilled can be launched on CPU and FRED-T5-1.7B can be run locally with the cost of 0.82$ per 1.16M tokens on a single Nvidia A100 80GB[15], which is cheaper than the API of proprietary GPT-4o, YandexGPT-4 Pro and GigaChat models.

[15]Assuming average throughput of 317 tokens per second for the optimized FRED-T5-1.7B model, an hour of uninterrupted work results in $3660 \times 317 = 1.160.220 \approx 1.1$M tokens, and one hour of renting Nvidia A100 80GB costs 0.82$ (see `vast.ai`)

7. Demo Serving

System requirements We use a single Nvidia A100 80GB GPU to deploy SAGE models. To manage inputs longer than 1,000 symbols (approximately 256 tokens), we segment the initial text into consecutive blocks of 1,000 symbols or less. These blocks are then processed sequentially by the model. During inference, we utilize greedy decoding with the temperature set to 1.0. We have found that constraining SAGE models in terms of generation diversity tends to result in copying the input.

User experience The Web demo includes a user-friendly form for selecting models, entering text, or choosing from random samples, along with an error indication mode that highlights and categorizes mistakes. Additionally, the user interface includes a 'Copy' button that enables users to save the corrected text without any error markup. Both options provide three models to choose from: the advanced model, its distilled version, and the Russian-English model.

8. Conclusion

We introduced SAGE, an efficient open-source framework for the joint correction of spelling, punctuation, and capitalization errors in Russian and English texts. SAGE addresses the limitations of LLM-based approaches while avoiding the computational and privacy costs associated with proprietary models. Our 95M Russian model matches the performance of GPT-4o on CPU, and evaluations demonstrate superiority over existing tools and LLMs across all error correction tasks. The open release includes a family of models, a public training code, and an expanded benchmark. Additionally, SAGE can be utilized in a library, as a system demo providing practical solutions for error correction in resource-constrained scenarios.

Limitations

Length Although our models achieve strong results and demonstrates SOTA performance in spelling error correction tasks, the limited context window size of 1024 tokens restricts its application in long-context scenarios. The evaluation datasets vary in length but are primarily designed for sentences rather than longer texts. This approach is not correlated with overall quality since any text can be divided into smaller pieces. While the training dataset includes various lengths, they are constrained to fit within the model's context.

API limitations The current solution aims to strike a balance between size and quality. The demo operates in a synchronous manner since it is not a high-load service. The API provided is intended for demonstration purposes and will be scaled further for production use.

Optimization approach and architecture limitations In Section 4.1, we outlined the rationale for selecting the encoder-decoder architecture among open Russian models. However, other architectures beyond transformers merit further investigation, which we plan to explore in future research.

This study did not employ optimization methods that are incompatible with the T5 architecture. For example, TensorRT-LLM (trt, 2024) supports various quantization techniques for decoder-only models, which can be used alongside kernel optimization. However, for models like T5, TensorRT-LLM only provides support for kernel optimization, and this is restricted to models that use the Hugging Face Transformers API (Wolf et al., 2020).

At the time of this work, the vLLM framework (Kwon et al., 2023) provided KV cache management, quantization, and kernel optimization methods exclusively for decoder-only models (iss, 2023). The framework's API made it challenging to extend support to encoder-decoder architectures, as managing KV caches for hidden states in cross-attention layers is necessary, a feature not present in decoder-only models (Vaswani et al., 2023). Although this limitation was later addressed, and vLLM now supports encoder-decoder models, it still lacks full compatibility with T5-like models (iss, 2024) due to the custom position bias used in these architectures.

Few-shot evaluation All competing LLMs were evaluated in a zero-shot setting to emulate real-world usage patterns for spelling and punctuation correction. Although few-shot evaluation is expected to improve the performance of competing models, the breadth of Russian orthography and syntax substantially limits the generalizability of few-shot prompting; therefore, the results are not expected to deviate significantly from those obtained in the zero-shot setting.

Ethics Statement

Bias and Fairness LLMs can reflect and reinforce the biases present in their training data. Spelling correction issues can unintentionally result in stereotypes, particularly when correcting words or names associated with underrepresented or marginalized groups. For example, the API might prioritize standard Russian spellings and incorrectly mark culturally significant names or terms as errors. To mitigate this issue, we ensure that our training data is diverse and representative, making a concerted effort to include various domains in the data and test the demo on a multi-domain benchmark. However, it is essential to note that we do not aim to cover dialects or specific language variations within minority groups, as the demo is designed to utilize standard Russian.

Privacy and Security Spelling error correction APIs typically process user-inputted text, which may contain sensitive or personal information. We do not store or misuse the data it processes in the current API.

Misuse and Overreliance The usage of the spelling API improves spelling accuracy but may cause users to over-rely on it, ignoring learning language rules or critically assessing corrections. We understand these risks and encourage users to avoid misusing the API.

9. Bibliographical References

2023. vLLM Issue 187 - Adding support for encoder-decoder models, like T5 or BART.

2024. Optimizing inference on large language models with nvidia tensorrt-llm, now publicly available.

2024. vLLM Issue 7366 - [RFC]: Encoder/decoder models & feature compatibility.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report.

Andrey Bout, Alexander Podolskiy, Sergey Nikolenko, and Irina Piontkovskaya. 2023. Efficient grammatical error correction via multi-task training and optimized training schedule. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5800–5816, Singapore. Association for Computational Linguistics.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2023. Grammatical error correction: A survey of the state of the art. *Computational Linguistics*, 49(3):643–701.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. How is chatgpt's behavior changing over time?

Xiang Chen, Chaoyang Gao, Chunyang Chen, Guangbei Zhang, and Yong Liu. 2024. An empirical study on challenges for llm developers. *arXiv preprint arXiv:2408.05002*.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, volume 235 of *Proceedings of Machine Learning Research*, pages 8359–8388. PMLR.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Steven Coyne, Keisuke Sakaguchi, Diana Galvan-Sosa, Michael Zock, and Kentaro Inui. 2023. Analyzing the performance of gpt-3.5 and gpt-4 in grammatical error correction. *arXiv preprint arXiv:2303.14342*.

Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Tao Fang, Shu Yang, Kaixin Lan, Derek F Wong, Jinpeng Hu, Lidia S Chao, and Yue Zhang. 2023. Is chatgpt a highly fluent grammatical error correction system? a comprehensive evaluation. *arXiv preprint arXiv:2304.01746*.

Alena Fenogenova, Artem Chervyakov, Nikita Martynov, Anastasia Kozlova, Maria Tikhonova, Albina Akhmetgareeva, Anton Emelyanov, Denis Shevelev, Pavel Lebedev, Leonid Sinev, Ulyana Isaeva, Katerina Kolomeytseva, Daniil Moskovskiy, Elizaveta Goncharova, Nikita Savushkin, Polina Mikhailova, Anastasia Minaeva, Denis Dimitrov, Alexander Panchenko, and Sergey Markov. 2024. MERA: A comprehensive LLM evaluation in Russian. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9920–9948, Bangkok, Thailand. Association for Computational Linguistics.

Team Gemma, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention.

V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Renjie Liu, Yanxiang Zhang, Yun Zhu, Haicheng Sun, Yuanbo Zhang, Michael Xuelin Huang, Shanqing Cai, Lei Meng, and Shumin Zhai. 2024. Proofread: Fixes all errors with one tap. *arXiv preprint arXiv:2406.04523*.

Nikita Martynov, Mark Baushenko, Alexander Abramov, and Alena Fenogenova. 2023. Augmentation methods for spelling corruptions. In *Proceedings of the International Conference "Dialogue"*, volume 2023.

Nikita Martynov, Mark Baushenko, Anastasia Kozlova, Katerina Kolomeytseva, Aleksandr Abramov, and Alena Fenogenova. 2024. A methodology for generative spelling correction via natural spelling errors emulation across multiple domains and languages. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 138–155, St. Julian's, Malta. Association for Computational Linguistics.

Aleksandr Nikolich, Konstantin Korolev, Sergei Bratchikov, Igor Kiselev, and Artem Shelmanov. 2026. Vikhr: The family of open-source instruction-tuned large language models for russian.

Marina Panina, Alexey Baitin, and Irina Galinskaya. 2013. Context-independent autocorrection of query spelling errors.[avtomaticheskoe ispravlenie opechatok v poiskovykh zaprosakh bez ucheta konteksta]. In *In Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue"*, pages 556–568.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Oleh Shliazhko, Alena Fenogenova, Maria Tikhonova, Anastasia Kozlova, Vladislav Mikhailov, and Tatiana Shavrina. 2024. mgpt: Few-shot learners go multilingual. *Transactions of the Association for Computational Linguistics*, 12:58–79.

Alexey Sorokin. 2017. Spelling correction for morphologically rich language: a case study of russian. In *Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing*, pages 45–53.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, et al. 2023. Ul2: Unifying language learning paradigms. In *International Conference on Learning Representations (ICLR 2023)*.

Mikhail Tikhomirov and Daniil Chernyshev. 2023. Impact of tokenization on llama russian adaptation. In *2023 Ivannikov Ispras Open Conference (ISPRAS)*, pages 163–168. IEEE.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Haoran Wu, Wenxuan Wang, Yuxuan Wan, Wenxiang Jiao, and Michael Lyu. 2023. Chatgpt or grammarly? evaluating chatgpt on grammatical error correction benchmark. *arXiv preprint arXiv:2303.13648*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2021)*, pages 483–498, Online. Association for Computational Linguistics.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al.

2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Dmitry Zmitrovich, Aleksandr Abramov, Andrey Kalmykov, Vitaly Kadulin, Maria Tikhonova, Ekaterina Taktasheva, Danil Astafurov, Mark Baushenko, Artem Snegirev, Tatiana Shavrina, Sergei S. Markov, Vladislav Mikhailov, and Alena Fenogenova. 2024. A family of pre-trained transformer language models for Russian. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 507–524, Torino, Italia. ELRA and ICCL.

10. Language Resource References

CJ Bryant, Mariano Felice, and Edward 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 (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Anisia Katinskaia, Maria Lebedeva, Jue Hou, and Roman Yangarber. 2022. Semi-automatically annotated learner corpus for russian. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 832–839.

Nikita Martynov, Mark Baushenko, Anastasia Kozlova, Katerina Kolomeytseva, Aleksandr Abramov, and Alena Fenogenova. 2024. A methodology for generative spelling correction via natural spelling errors emulation across multiple domains and languages. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 138–155, St. Julian's, Malta. Association for Computational Linguistics.

Sai Muralidhar Jayanthi, Danish Pruthi, and Graham Neubig. 2020. Neuspell: A neural spelling correction toolkit. *arXiv e-prints*, pages arXiv–2010.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017), Volume 2: Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

Tatiana Shavrina, Alexey Sorokin, Alexey Baitin, Irina Galinskaya, and Elena Rykunova. 2016. Spellrueval : the first competition on automatic spelling correction for russian. In *Proceedings of the Annual International Conference "Dialogue"*, volume 15.

Alexey Sorokin, Alexey Baytin, Irina Galinskaya, and Tatiana Shavrina. 2016. Spellrueval: The first competition on automatic spelling correction for russian. In *Proceedings of the Annual International Conference "Dialogue"*, volume 15.

A. List of Models and Services

In this section, we provide a list of LLMs and services that we employed for evaluation alongside the SAGE models on the test splits of datasets described in Section 3.3. The models are selected based on their performance on Chatbot Arena[16] (Chiang et al., 2024), Russian LLM Arena[17] and MERA leaderboard[18] (Fenogenova et al., 2024).

General information on the models and services is given below. Table 5 lists the evaluated LLMs and spelling services with their parameters, context length, and references.

B. Human Evaluation Instruction

This section provides an instruction employed for expert annotation procedure explained in Section 5.2. The instruction reads as follows:

You will be provided with text pairs: an original Russian sentence that may contain errors, and a corrected sentence. The corrected sentence was obtained through automatic correction of the original sentence. The correction was generated using the SAGE generative model. The task is to evaluate how well SAGE performs error correction.
The original sentence may contain spelling, grammatical, punctuation errors, and/or errors related to incorrect case usage ("london" instead of "London").
SAGE should correct all such errors when present, while considering the text style when making correction decisions. In other words, if the text represents social media correspondence using vocabulary characteristic of conversational style, SAGE ***should not*** *correct it to dictionary equivalents.*
For example, if the original sentence is "Ya poshel domoy i taaaaaaaaak dolgo zhdal poka lift priedit", the ideal correction we expect from SAGE would be "Ya poshyol domoy i taaaaaaaaak dolgo zhdal, poka lift priedet.".

[16] https://lmarena.ai
[17] https://llmarena.ru/
[18] https://mera.a-ai.ru/en

	Model	Parameters	Context length	Link to source	Citation
Large Language Models	GigaChat Max GigaChat Pro GigaChat Plus	— — —	32k 32k 32k	https://giga.chat	—
	YandexGPT-4 Pro	—	32k	ya.ru/ai/gpt-4	—
	T-lite-instruct-0.1	7B	8k	T-lite-instruct-0.1	—
	Vikhr-8B-instruct Vikhr-12B-instruct	8B 12B	128k 128k	Vikhr-Llama3.1-8B-Instruct Vikhr-Nemo-12B-Instruct	Nikolich et al. (2026)
	Saiga-Llama3-70B	11.3B	8k	saiga_llama3_70b_sft	—
	Llama-3.1-405B Llama-3.1-70B Llama-3.1-8B	405B 70B 8B	128k	meta-llama/Llama-3.1-405B-Instruct meta-llama/Llama-3.1-70B-Instruct meta-llama/Llama-3.1-8B-Instruct	Dubey et al. (2024)
	Qwen-32B-instruct	32B	8k	msu-rcc-lair/RuadaptQwen-32B-instruct	Tikhomirov and Chernyshev (2023)
	Gemma-2-27B	27B	8k	google/gemma-2-27b-it	Gemma et al. (2024)
	GPT-4o	—	128k	openai.com/index/hello-gpt-4o/	—
Services	Yandex.Speller Hunspell JamSpell	— — —	— — —	yandex.ru/dev/speller hunspell.github.io github.com/bakwc/JamSpell	— — —

Table 5: General information about LLMs and services employed for the evaluation.

This example demonstrates the ideal case of SAGE's performance and what we ultimately expect from it. For all text pairs shown to you, we have ideal corrections, and we can automatically verify how many errors SAGE corrects and how many it leaves uncorrected. However, the goal of the current task is to measure user satisfaction with SAGE's performance. In other words, we aim to understand in how many cases SAGE will produce a correction that the user (i.e., annotator in this case) ***would not need to manually revise*** *again. Thus, the primary objective of this annotation task is to measure how "acceptable" SAGE's proposed corrections are from the end-user perspective, rather than from the standpoint of automatic metrics.*