RuWikiBench: Evaluating Large Language Models through replication of encyclopedia articles

D. A. Grigoriev[1][2]***, D. I. Chernyshev***[1][3]

In light of the growing interest in using large language models (LLMs) as tools for generating scientific texts, the evaluation of their ability to produce encyclopedic content is becoming increasingly relevant. However, for Russian-language materials this issue has not been sufficiently studied, and existing benchmarks do not cover key aspects of analytical work with sources. This paper presents RuWikiBench - an open benchmark based on Ruwiki for evaluating the ability of large language models to reproduce Wikipedia-style articles, built around three tasks: selection of relevant sources, article structuring, and section generation. The results of testing popular open-source LLMs show that even under ideal conditions, the best models do not always follow the expert logic of composing encyclopedic content: even with a perfect source retrieval system, the models cannot reproduce the reference outline, and the quality of section generation shows almost no dependence on the number of model parameters.

Keywords: benchmark, Wikipedia, Ruwiki, large language model.

Introduction

Modern large language models demonstrate impressive results in generating texts of various styles and themes. However, their capabilities for working with scientific and encyclopedic materials remain understudied, particularly for Russian-language texts. Existing methods for evaluating model capabilities predominantly focus on standard linguistic tasks, without paying sufficient attention to analytical abilities when working with scientific texts. For the Russian language, this problem is especially relevant due to the limited availability of specialized evaluation tools.

There are many benchmarks covering various linguistic tasks for the Russian language. RussianSuperGlue [1] evaluates general language understanding and basic natural language processing tasks. MERA [2] provides unified testing conditions for models by compiling generation instructions for each task; however, the tasks themselves are oriented towards testing general comprehension. LIBRA [3] focuses on testing a model's ability to retain and retrieve information from a large context but is centered on short answers that do not require deep reasoning. Ru Arena General [4] focuses on pairwise model comparison rather than overall answer quality. Ping-Pong [5] evaluates the dialog abilities of models, which is important for interactive systems, but is not suitable for assessing the ability to conduct research and write coherent scientific-encyclopedic texts. At the same time, an entire class of tasks related to deep text analysis remains uncovered: creating detailed, structured, and factually accurate texts supported by a large number of sources.

The recent development of new agent capabilities, such as the emergence of the "Deep Research" function by OpenAI [6] or the development of the universal Storm algorithm [7], indicates a growing interest in conducting scientific research using large language models. This highlights the need to create new approaches for objectively evaluating the analytical capabilities of models. Existing benchmarks only partially address aspects critical for generating scientific-encyclopedic texts, such as the ability to extract relevant information from a set of documents, plan the structure of a future text, maintain text coherence and fluency, as well as ensure the factual consistency.

[1]Lomonosov Moscow State University, Moscow, Russia
[2]E-mail: dagrig14@yandex.ru
[3]E-mail: chdanorbis@yandex.ru

One of the closest studies in this area is the ResearchArena [8] benchmark, which formalizes the construction of an academic review; however, it is more aimed at testing the models' ability to select and organize relevant information and does not address their ability to generate coherent scientific-encyclopedic texts.

This paper proposes an approach aimed at creating tools to test how well large language models can work with scientific-encyclopedic texts. Our main contributions:

1. We collect "Ruwiki", a labeled dataset for generation of wikipedia-style articles in Russian language;
2. We propose new benchmark, RuWikiBench that measures large language model analytical capabilities in Russian language;
3. We evaluate the abilities of popular open-source large language models to generate Wikipedia-style articles.

The code and data of this work have been made publicly available[4].

1. Dataset collection

To measure analytical capabilities of large language models we need a labeled corpus of texts that would contain various sources, the respective analysis of those sources and analysis-source mapping for factuality evaluation of claims. Wikipedia articles are the best candidates for such data because this genre simultaneously requires factual accuracy, completeness of analysis, and understanding of context.

We utilize articles from Russian online encyclopedia "RuWiki" which distinguishing features are a large number of references to Russian-language sources, as well as stricter text filtering. These qualities ensure that the articles can be reliably verified and reproduced by human experts and therefore replicated by LLMs.

The data acquisition process included the following steps:

1. **Article Selection**: Articles on diverse topics containing a sufficient number of references to external sources were manually selected;
2. **Source Downloading**: For each article, the available sources it references were web-scraped;
3. **Splitting into Snippets**: To reproduce real Retrieval Augmented Generation (RAG) conditions, all texts were split into small fragments of approximately ≈ 600 words in length.

Fig. 1 shows a brief schematic of the source text extraction process. The sources were downloaded using the Python module `newspaper3k`[5]. A subset of "Ruwiki" articles B is taken as the initial corpus. To extract article's HTML code we use BeautifulSoup4[6]. The obtained text is structured by splitting it into fragments corresponding to nested headings (H1, H2, H3, etc.), which preserves both the substantive part of the article and its hierarchical organization. Next, all external references cited in the "Notes" section are automatically extracted. Invalid links (e.g., 404 error) are excluded from further processing, and the text associated with them is removed, leaving only those sources that are actually accessible.

Fig. 2 illustrates the schematic breakdown of an article[7] into key entities used in subsequent processing. During the data processing stage, text filtering is performed to ensure its correct

[4] `https://github.com/Nejimaki-Tori/WikiBench`
[5] `https://github.com/codelucas/newspaper`
[6] `https://beautiful-soup-4.readthedocs.io/en/latest/`
[7] `https://ru.ruwiki.ru/wiki/Python`

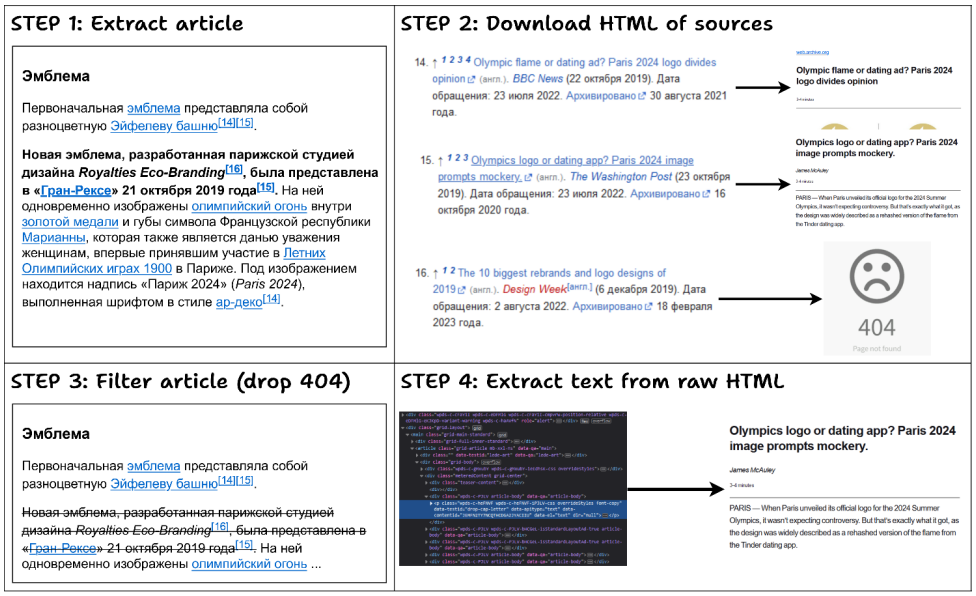

Figure 1. Source extract

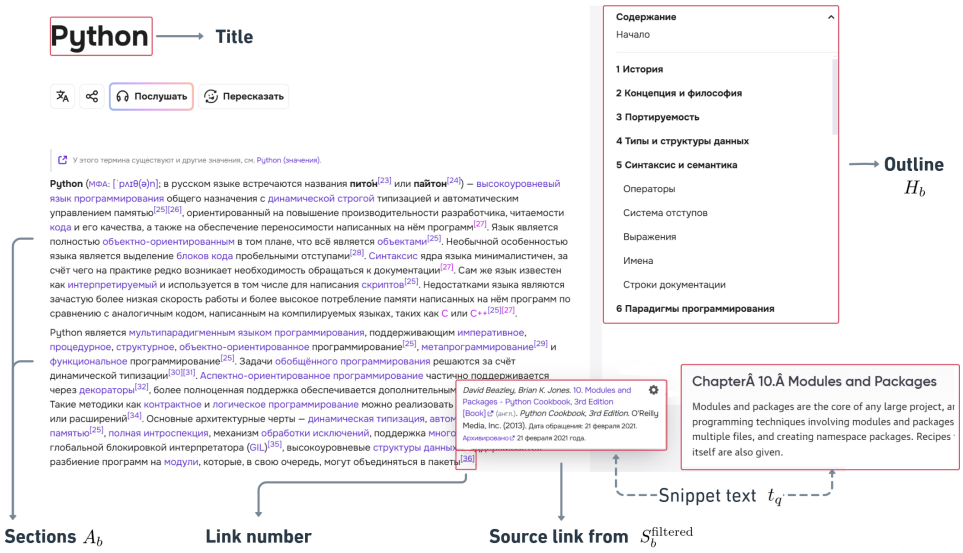

Figure 2. Main article entities

interpretation by the model. Each footnote (e.g., [1], [2]) is matched to a specific link corresponding to one of the available sources. This allows for precise identification of the link's position in the article text and its use for filtering.

Table 1. Key characteristics of the collected dataset

Metric	RuWikiBench	ResearchArena
Number of articles	285	7,952
Number of sources	15,686	12,034,505
Total number of snippets	36,860	-
Average outline size (headings)	37	-
Average section size (words)	112	-

Based on the valid links S_b^{filtered}, filtered sets of paragraphs A_b^{filtered} and headings H_b^{filtered} are formed. That is, only content supported by the extracted sources is retained; everything else is removed. Only sources for which a text t_q of at least 1500 characters could be retrieved are kept, to filter out "noisy" responses from HTML pages such as errors (e.g., error 404) or blocking messages. A_b^{filtered} retains only those paragraphs that contain at least one link to a source for which text was successfully retrieved. Similarly, H_b^{filtered} is formed-only those headings under which at least one paragraph remains. The characteristics of the collected corpus are presented in Tab. 1.

2. Evaluation Methodology

To objectively assess the ability of language models to generate scientific and encyclopedic texts, it is necessary to replicate the real process of preparing encyclopedic content:

1. **Selection of relevant sources**: The model is given the article title and a set of snippets, among which it must identify and rank materials relevant to the topic by their significance;
2. **Article structure construction**: Based on the topic and selected sources, the model creates an outline with main sections in the Wikipedia-style;
3. **Section generation**: Article materials are distributed across sections, after which a summary of the relevant materials is generated for each section.

Each stage is evaluated independently of the previous ones, allowing for a quantitative measurement of the quality of each specific subtask.

2.1. Selecting relevant sources

One of the most effective search strategies [9] is the preliminary generation of an expected result (description) based on the original query (article title) to create an expanded search query. In our pipeline the description is generated in both Russian and English, to address available source text language diversity. The queries in both languages are then combined into a single textual query for a BM25-based search system.

Experiments were conducted with two approaches to expanded query generation:

1. **Ground-truth query based on the title and second-level headings**: A ground-truth query which is used for direct evaluation of the models' ranking abilities. We use LLaMa 3 70b model [10] was used to generate the query;
2. **Query generated from the title by the evaluated model**: Similar to real-world conditions, the LLM is fully responsible for the quality of the results and independently decides which search query to generate for BM25.

Examples of generated descriptions are shown in Tab. 2.

The documents selected by the BM25 query are sequentially passed to the large language model, which must classify each snippet as relevant (answer "yes") or non-relevant (answer "no"). To obtain numerical scores, we compare related wikipedia article titles of retrieved documents to title of article we seek to generate. The logarithmic probability of the tokens in the model's response is taken: if the answer is positive, the probability $P(\text{yes})$ is used; if negative, $1 - P(\text{no})$ is used. This approach allows ranking the retrieved documents by the model's confidence in their relevance: the higher the probability, the higher the model's confidence in the response, and the higher the document is ranked in the results.

Table 2. Comparison of english-translated descriptions of the article “C++” in two variants

Query Variant	Text
Ground-truth	The article "C++" is an overview of the C++ programming language, its history, structure, and features. It covers the main aspects of the language, including its standard library, differences from the C language, and further development. In addition, the article contains examples of C++ programs, comparisons with alternative programming languages, as well as critical analysis and discussion of the influence of C++ on the development of programming and existing alternatives. The article is intended for readers interested in the C++ language and its role in modern programming.
Generation by title	The article "C++" may be dedicated to the C++ programming language, one of the most popular and widely used programming languages in the world. The article may cover the basics of the language, its history, syntax and features, as well as its applications in various fields such as operating systems development, games, and web applications. In addition, the article may include information about C++ standards and libraries, as well as its comparison with other programming languages. The article can be useful both for beginner programmers and for experienced professionals who want to deepen their knowledge of C++. It may also include code examples and practical tips on using C++ in real projects.

2.2. Article structure planning

First, each text fragment (snippet) from the reference article source is converted by embedding model. Then, the snippets are clustered into potential section contents. To guarantee reproducibility KMeans algorithm is applied with the number of clusters equal to the number of second-level headings in the reference outline, and the centroids are initialized with the vector representations of these headings.

Next, five snippets closest to the cluster center are selected. This is done to reduce the influence of less relevant snippets on the final outline. The generation of mini-outlines for sections is carried out taking into account two key parameters: the context window size (to account for references and the overall semantics of the document) and two generation modes - directly from the texts and through preliminary generation of a brief cluster description. These generation modes enable different levels of abstraction: the direct mode preserves details with raw data, while the mode via preliminary cluster description improves consistency and reduces information duplication. At the final stage, all mini-outlines are combined into the final structured article outline.

2.3. Section generation

For each article section, we extract all snippets that were indicated as sources for the reference text of the section. These snippets are again converted into embeddings, and a pairwise similarity matrix is constructed as the product $E \times E^{\top}$. Elements with a similarity value above the threshold of 0.8 (empirically determined) are considered semantically close and are grouped together to avoid redundant repetitions during generation (e.g., when different sources paraphrase the same information). For each such semantic group, a hierarchical representation is built: the first five

texts are taken, and a brief description is generated based on them. This description is then supplemented using the next five texts, and so on, until a complete compressed representation of the group is obtained. Thus, only a set of brief descriptions remains - the most important information without excessive repetition. After this, the text of the section is generated based on the obtained group descriptions using a hierarchical summarization method [11].

3. Benchmark parameters

Below is a description of all data, models, hyperparameters, and procedures used to ensure reproducibility and analysis.

3.1. Generation parameters

For all models, unless otherwise specified, the same generation parameters were used: temperature - 0.01, repetition penalty - 1.0, and top_p - 0.9.

3.2. Relevant source selection

Snippet indexing was performed using BM25[8] across the entire corpus of collected snippets without hyperparameter tuning (default values). For each relevant document, two non-relevant documents were selected (ratio 1:2) - this was done to improve evaluation robustness.

3.3. Article structure planning

To build snippet embeddings we used `sergeyzh/BERTA`[9] model. Two context window options were considered: either a zero window (only the snippet itself) or one neighboring snippet to the left and right to expand the context. Heading similarity with reference headings was compared using cosine similarity: semantic correspondence was prioritized over exact wording or heading level. The comparison was performed against the cleaned article structure: all headings whose sections consisted entirely of text without downloadable sources were removed from the preprocessed text.

4. Metrics

Within the benchmark, two groups of metrics are considered: ranking metrics, which assess how well the model selects relevant sources, and text similarity metrics, which measure how closely the generated content matches the reference.

4.1. Ranking Metrics

To evaluate the quality of the source list, we use NDCG@K [12] and R-Precision [13]:

$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}, \tag{1}$$

[8] `https://github.com/xhluca/bm25s`

[9] `https://huggingface.co/sergeyzh/BERTA`

$$\text{DCG@K} = \sum_{i=1}^{K} \frac{\text{rel}_i}{\log_2(i+1)}, \tag{2}$$

$$\text{IDCG@K} = \sum_{i=1}^{K} \frac{\text{rel}_i^{\text{IDEAL}}}{\log_2(i+1)}, \tag{3}$$

$$\text{R-Precision} = \frac{\sum_{i=1}^{R} \text{rel}_i}{R}, \tag{4}$$

where $\text{rel}_i \in \{0, 1\}$ is the indicator of relevance for the document at position i; $\text{rel}_i^{\text{IDEAL}}$ is the same quantity in the ideal (fully sorted) ranking; R is the total number of relevant documents for the given query.

4.2. Text Similarity Metrics

The quality of generated sections and headings is evaluated with **BERTScore** [14]:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^\top \hat{x}_j, \tag{5}$$

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i^\top \hat{x}_j, \tag{6}$$

$$F_{\text{BERT}} = \frac{2\, P_{\text{BERT}}\, R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}, \tag{7}$$

where x is the reference text, $\hat{x}$ is the generated text; each sentence is encoded using the model[10], after which cosine similarity is computed.

ROUGE-L and BLEU were also considered for evaluating section generations.

ROUGE-L [15] is based on the length of the longest common subsequence (LCS) between the generated summary S and the reference R:

$$\text{Precision} = \frac{\text{LCS}(S, R)}{|S|}, \tag{8}$$

$$\text{Recall} = \frac{\text{LCS}(S, R)}{|R|}, \tag{9}$$

$$\text{ROUGE-L} = \frac{2\ \text{Precision}\ \cdot\ \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{10}$$

BLEU [16] is an n-gram precision metric with a brevity penalty:

$$\text{BLEU}_N = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n \right), \tag{11}$$

where p_n is the precision for n-grams, w_n are the weights, and BP is the brevity penalty.

[10] `https://huggingface.co/sergeyzh/BERTA`

5. Experiments

5.1. Models

The experiments used the following large language models: RuadaptQwen2.5-7B-Lite-Beta [17], RuadaptQwen3-32B-Instruct-v2 [17], DeepSeek V3 [18], Qwen3-235B-A22B [19], tpro [20] and yagpt5lite [21]. In all tables, models are grouped by size, and the best results within each group are highlighted.

5.2. Results

Tab. 3 and Tab. 4 present the results of source selection evaluation. As the baseline we use BM25 retrieval without model-based reranking. In the first case (Tab. 3), where a pre-generated ground-truth search query was used for all models, the best results were achieved by DeepSeek V3, which indicates a strong ability to select relevant documents. In the second experiment (Tab. 4), where the query was based solely on the article title, tpro achieved the best results. The experiment showed that LLM query generation is not inferior in ranking quality to the reference setup. Presumably, this is because (as shown in Tab. 2) the queries turn out to be very similar to ground-truth: LLMs have knowledge of Wikipedia's typical article structure from training and can therefore connect relevant concepts in the required format.

Table 3. Results of pure ranking ability evaluation

Model	**NDCG**	**R-Pr**
baseline (bm25)	88.81	62.51
DeepSeek V3	**95.42**	**83.86**
Qwen3-235B-A22B	94.49	82.42
RuadaptQwen3-32B-Instruct-v2	95.25	81.81
tpro	95.42	83.53
RuadaptQwen2.5-7B-Lite-Beta	88.26	62.26
yagpt5lite	90.35	77.66

Table 4. Results of evaluating BM25 query-generation ability

Model	**BM25**		**Rerank**	
	NDCG	**R-Pr**	**NDCG**	**R-Pr**
DeepSeek V3	88.39	60.65	95.67	83.07
Qwen3-235B-A22B	89.17	62.98	94.90	81.96
RuadaptQwen3-32B-Instruct-v2	85.39	52.80	95.82	81.62
tpro	**90.61**	**65.07**	**96.06**	**83.37**
RuadaptQwen2.5-7B-Lite-Beta	88.81	62.51	88.23	60.96
yagpt5lite	86.59	57.98	90.27	77.65

Overall, the models show fairly high performance at this stage, which may be due to the fact that an article title reflects its content well. In the best cases, up to 80% of the documents in the sample are relevant, which implies there is significant room for for further improvement.

Table 5. Results of outline generation

Model	Mean BERTScore F1		
	Direct		Description
	no neighbors	one neighbor	
DeepSeek V3	**63.51**	**62.93**	**65.50**
Qwen3-235B-A22B	60.86	59.06	62.66
RuadaptQwen3-32B-Instruct-v2	60.12	60.04	62.91
tpro	60.32	59.09	60.75
RuadaptQwen2.5-7B-Lite-Beta	60.03	58.21	61.58
yagpt5lite	59.72	60.07	60.25

Tab. 5 presents the results of article structure planning. Direct - generation from the cluster snippets, with neighbor context as indicated; Description - generation via a preliminary description of all cluster elements. The results show that with preliminary description generation, all models consistently improve in quality. RuadaptQwen3 shows the largest gain, rising to second place and effectively matching the results of the larger model, Qwen3-235B-A22B. DeepSeek V3 remains the leader, showing a substantial margin over the others. At the bottom in quality are RuadaptQwen2.5-7B-Lite-Beta and yagpt5lite. At the same time, yagpt5lite, with only 8 billion parameters, delivers results comparable to a 32-billion-parameter model. Tab. 6 shows a comparison of a small excerpt of the reference and generated outlines. A common issue across all models was excessive heading hierarchy depth. On "Ruwiki", headings were rarely deeper than level three; however, the models often created fourth- and fifth-level headings, implying that all information belongs in one large section, even though it may differ somewhat in meaning and, in the original outline, would correspond to unrelated headings.

Table 6. Comparison of two englis-translated article outlines

Generated	Reference
# Introduction to Python	# Python
## Language Overview	## History
### History and Key Aspects	## Concept and Philosophy
#### Main Features and Implementations	## Portability
# Python Basics	## Data Types and Structures
## Syntax and Semantics	## Syntax and Semantics
### Data Types and Structures	### Indentation System
#### Numbers, Lists, Dictionaries	### Expressions
and Object-Oriented Programming	### Names
# Advanced Python Topics	### Documentation Strings
## Flow Control and Multithreading	## Programming Paradigms
...	...

Tab. 7 reports the evaluation of section-generation quality. The difference between model performance may be considered marginal, but this is due to the sensitivity of the metric used. Sections for which the algorithm did not select a single relevant snippet were excluded from the final metrics. The best overall results were demonstrated by Qwen3-235B-A22B; however, in terms

of ROUGE-L and BLEU, RuadaptQwen3-32BInstruct-v2 shows better structural consistency and greater phrase overlap with the reference. The yagpt5lite model performs above average, especially on BLEU, at a much smaller size, whereas tpro shows the lowest values across all metrics.

Table 7. Results of section generation

Model	Mean F1	Mean ROUGE-L	Mean BLEU
DeepSeek V3	53.48	14.34	2.81
Qwen3-235B-A22B	**53.74**	14.63	3.07
RuadaptQwen3-32B-Instruct-v2	53.21	**15.46**	**3.40**
tpro	53.15	13.58	2.27
RuadaptQwen2.5-7B-Lite-Beta	52.99	12.29	2.11
yagpt5lite	53.43	14.85	3.16

Table 8. Comparison of english-translated texts of two sections

Model	Text
DeepSeek V3	**COVID-19 is an infectious disease**, ... which **led to a global pandemic** that began in 2020. Initially presenting with respiratory symptoms such as cough, fever, and shortness of breath, the disease can cause severe complications, ... The virus is highly transmissible, is presumed to have a zoonotic origin, and spread rapidly worldwide. To control the pandemic, the WHO recommends vaccination, mask-wearing, social distancing, and hand hygiene, with vaccine effectiveness against the original strain reaching 85% or higher. Although COVID-19 in children more often has a mild course, severe cases are possible, including multisystem inflammatory syndrome.
yagpt5lite	**COVID-19 is a pandemic** caused by the novel coronavirus SARS-CoV-2. As of January 14, 2022, the WHO had confirmed about **318,648,834** cases of COVID-19 worldwide, including **5,518,343** deaths. The first COVID-19 vaccine was introduced in December 2020. On December 2, 2020, the United Kingdom became the first country to approve the Pfizer-BioNTech (BNT162) vaccine, which the WHO authorized for emergency use. SARS-CoV-2 is considered more contagious than SARS-CoV and quickly spread around the world after several infection cases in Wuhan, China. The pathogenesis of SARS-CoV-2 is associated with inflammatory responses that adversely affect the lungs and cause symptoms such as cough, fever, general malaise, shortness of breath, and respiratory failure.

Table 9. Comparison of texts of two sections

For a qualitative evaluation of section-generation quality, we consider the introductory parts of the article "COVID19" produced by DeepSeek V3 and yagpt5lite, respectively, shown in Tab. 9. Despite some semantic inaccuracies (for example, the statement "COVID-19 is a pandemic," whereas in reality it is a disease), yagpt5lite demonstrates a quite solid result. Its text falls short

of DeepSeek V3's version in terms of coverage and systematic exposition, but contains more numerical data and concrete facts. At the same time, the material generated by DeepSeek V3 tends to resemble an encyclopedic article, whereas the yagpt5lite version is closer in style to a technical report on the disease.

Conclusion

This paper proposes RuWikiBench benchmark for evaluating the analytical capabilities of large language models in generating scientific and encyclopedic texts in Russian. The core of the proposed evaluation system is a three-stage process, consisting of three independent systems that naturally arise when creating articles on a specific topic. Relying on a filtered "Ruwiki" corpus and a clearly defined evaluation methodology, the proposed benchmark establishes a foundation for further research in the application of language models to the task of generating scientific and encyclopedic text in Russian language.

Experiments showed that with a fixed search query, DeepSeek V3 demonstrates the best source selection quality, significantly outperforming BM25 without reranking. At the structure planning stage, it was found that adding a preliminary cluster description consistently improves the quality of outlines for all models, including DeepSeek V3, which demonstrated the best understanding of the process. All models showed comparable section generation quality; however, RuadaptQwen3-32B-Instruct-v2 leads in ROUGE-L and BLEU metrics, indicating a text structure more consistent with the reference. The work shows that models possess significant potential, but their reliable application requires further development of methods for analyzing and structuring review materials.

Acknowledgements

The study was supported by grant No. 25-11-00191 from the Russian Science Foundation. The work was carried out using the supercomputer "MSU-270" of the Lomonosov Moscow State University.

References

1. *RussianSuperGlue.* Shavrina, Tatiana, et al. "RussianSuperGLUE: A Russian Language Understanding Evaluation Benchmark." Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020.

2. *Mera.* Fenogenova, Alena, et al. "MERA: A Comprehensive LLM Evaluation in Russian." Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024.

3. *LIBRA.* Churin I. et al. Long Input Benchmark for Russian Analysis //CoRR. – 2024.

4. VikhrModels. RuLLM Arena: Russian LLM Evaluation Benchmark // GitHub repository. – URL: `https://github.com/VikhrModels/ru_llm_arena` (accessed: 01.08.2025).

5. *Ping-Pong.* Gusev I. PingPong: A Benchmark for Role-Playing Language Models with User Emulation and Multi-Model Evaluation.

6. *OpenAI.* Introducing deep research // OpenAI URL: https://openai.com/index/introducing-deep-research/ (accessed: 31.07.2025).

7. *Storm.* Shao Y. et al. Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models //NAACL-HLT. - 2024

8. *ResearchArena.* Kang, Hao, and Chenyan Xiong. "ResearchArena: Benchmarking LLMs' Ability to Collect and Organize Information as Research Agents." arXiv e-prints (2024): arXiv-2406.

9. *Reranking.* Wang X. et al. Searching for Best Practices in Retrieval-Augmented Generation //CoRR. - 2024.

10. Touvron H. et al. LLaMA: Open and Efficient Foundation Language Models.

11. Wu, Jeff, et al. "Recursively summarizing books with human feedback." arXiv preprint arXiv:2109.10862 (2021).

12. *NDCG.* Järvelin, Kalervo, and Jaana Kekäläinen. "Cumulated gain-based evaluation of IR techniques." ACM Transactions on Information Systems (TOIS) 20.4 (2002): 422-446.

13. *R-Precision.* BUCKLEY C. Evaluating Evaluation Measure Stability //ACM SIGIR 2000 Proceedings. - 2000.

14. *BERTScore.* Zhang T. et al. BERTScore: Evaluating Text Generation with BERT //International Conference on Learning Representations.

15. *ROUGE.* Lin, Chin-Yew. "Rouge: A package for automatic evaluation of summaries." Text summarization branches out. 2004.

16. *BLEU.* Papineni K. et al. BLEU: a Method for Automatic Evaluation of Machine Translation.

17. *RuadaptQwen.* Tikhomirov, Mikhail, and Daniil Chernyshev. "Facilitating large language model russian adaptation with learned embedding propagation." Journal of Language and Education 10.4 (40) (2024): 130-145.

18. *DeepSeek V3.* Liu A. et al. DeepSeek-V3 Technical Report //CoRR. - 2024.

19. *Qwen3-235B.* Yang A. et al. Qwen3 technical report //arXiv preprint arXiv:2505.09388. - 2025.

20. T-Bank has opened access to its own Russian-language language model in the weight category of 7-8 billion parameters / T-Bank URL: https://www.tbank.ru/about/news/20072024-t-bank-opened-access-its-own-russian-language-language-model-weight-category-of-7-8-billion-parameters/ (accessed: 10.05.2025).

21. YandexGPT 5 with reasoning mode // Yandex URL: https://ya.ru/ai/gpt (accessed: 30.07.2025).