Literature summarization with large language models

***D. A. Grigoriev*[1 2], *D. V. Khudiakov*[1 3], *D. I. Chernyshev*[1 4]**

This work is dedicated to automatic summarization of Russian literary fiction. An open dataset RuBookSum (600+ "book-synopsis" pairs) was compiled based on texts from LibRuSec and user-generated summaries from "Narodny Briefly". Four approaches to summarization are examined: the baseline hierarchical and the "blueprint" (Text-Blueprint) methods, which builds an outline in the form of question-answer pairs, as well as two new methods proposed in this work: a hierarchical method with node filtering based on cosine similarity of embeddings, and a modification of the blueprint method with question clustering using the KMeans algorithm. The best quality is achieved by Qwen3-235B-A22B in the blueprint method, while the hierarchical method with node filtering provides the best balance between generation time and quality.
Keywords: LLM, summarization, literature, books, brief retelling.

Introduction

Automatic text summarization is one of the key tasks in natural language processing. The goal is to create an informative synopsis of the source text while preserving its main meaning. In recent years, with the advent of large language models (LLMs), interest in automating summarization has increased across many genres, including fiction. Unlike scientific, news, or technical texts, fiction is characterized by high stylistic and semantic complexity. Non-linear storytelling, imagery, metaphor, and stylistic devices make synopsis writing especially challenging. The limited context window of modern models further complicates processing long texts.

At present moment there are not many datasets focusing specifically on summarizing fiction, and the key open datasets concentrate on non-Russian material. BookSum [1] is one of the first and best-known English-language datasets for abstractive summarization of narrative works. It contains books, plays, and short stories paired with summaries of varying granularity (paragraph level, chapter level, book level). Echoes from Alexandria [2] is a multilingual corpus of fiction, including five languages: English, German, French, Italian, and Spanish. FABLES [3] is a hand-curated corpus designed to evaluate factual faithfulness of summaries for book-length fiction. It includes 3,158 claims extracted from LLM-generated summaries for 26 books. Each claim is evaluated across model outputs by experts. According to FABLES, even advanced models (e.g., Claude) commit 20–30% factual errors, including distorted causal relations, incorrect characterization of protagonists, and overemphasis on minor details, judged by three criteria: agreement with original events, logical correctness, and absence of distortions.

In theory, automatic summarization can be performed in two main ways: extractive (selecting key text fragments) and abstractive (generating new text based on the source). For prose, abstractive summarization is typically chosen: key meanings and plot links are distributed throughout the text, so extractive sentence selection yields a fragmented, stylistically uneven result and does not reconstruct the plot, therefore abstractive approach was chosen.

[1]Lomonosov Moscow State University, Moscow Center for Fundamental and Applied Mathematics, Moscow, Russian Federation
[2]E-mail: dagrig14@yandex.ru
[3]E-mail: hydikovv17914@gmail.com
[4]E-mail: chdanorbis@yandex.ru

The topic is motivated by the growing need for tools capable of automatically producing concise, informative, and stylistically appropriate synopses for works of fiction. The goal of this work is to provide such tools, thus, following is proposed:

1. New Russian-language dataset that includes literary works and their synopses;
2. New summarization methods that offer alternatives to existing ones and substantially reduce the time required to produce a synopsis of a book.

Code and data are publicly available[5].

1. Dataset

At the start of the study, there were no open and representative corpora designed specifically for summarizing fiction in Russian. To run experiments and evaluate different methods a new dataset was created, using "Narodny Briefly" platform [4] where users publish synopses of literary works.

The synopses are texts created by users based on the original works. They vary in length-from a few sentences to several paragraphs-and in style: some reproduce key phrases verbatim, while others use freer narration. Some cover the whole work, others split content by chapter. Usually they contain the main facts and conclusions from the source text, but may include the author's commentary.

The book texts were selected from the LibRuSec digital library [5], one of the largest Russian-language fiction resources. Works were selected for which a synopsis existed on chosen source [4]. Each text underwent automatic preprocessing: meta-information (e.g., titles, chapter descriptions, technical inserts) was removed, then the text was formatted into a unified, standardized form suitable for use with models.

To better link books with their synopses, cosine similarity was used: the author name text from Briefly [4] and from LibRuSec [5] was embedded via SentenceTransformers with the model[6] and compared using cosine similarity. The synopses were automatically cleaned of HTML tags, comments, and service markers using LLM Meta-Llama 3-70B-Instruct. Then LibRuSec was searched and a collection of "book text – synopsis" pairs was formed.
The resulting dataset includes:

- 600+ cleaned user synopses from "Narodny Briefly" [4];
- 40+ different genres;
- source works from the LibRuSec digital library [5].

Table 1. Dataset overview

Dataset	Number of documents	Avg. document length (# words)	Avg. synopsis length (# words)	Compression ratio (synopsis length / text length)
RuBookSum	634	35052.64	700.77	8.43%
BookSum	405	112885.15	1167.20	0.79%
Gazeta	60964	632.77	41.94	6.99%

[5]`https://github.com/Nejimaki-Tori/BoookSum`
[6]`https://huggingface.co/deepvk/USER-bge-m3`

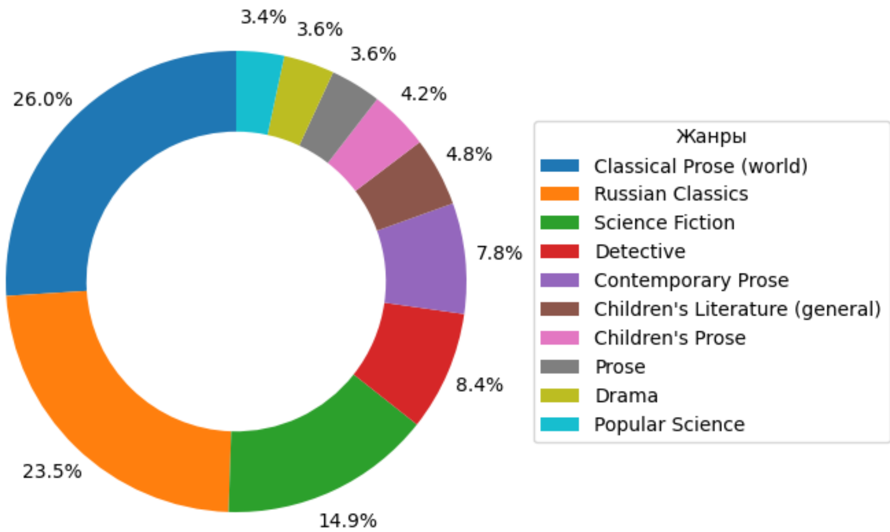

Figure 1. Distribution of texts by genres (top 10 genres)

Fig. 1 shows the genre distribution in the collection and Tab. 1 gives dataset statistics versus analogs.

2. Methodology

2.1. Hierarchical method (Algorithm 1)

This method [6] splits the text into chunks and generates a local synopsis for each chunk. These chunks are then grouped, and summaries are merged into higher-level synopses. The last level yields the final book-level synopsis.

2.2. Hierarchical method with node filtering (Algorithm 2)

The classical hierarchical method constructs the final synopsis through a multi-layered combination of intermediate obtained derived from individual text chunk. However, literary works often contain chunks that have little impact on plot development and include numerous redundant repetitions and minor details. During the generation of the final synopsis, these chunks can reduce its informativeness and, in some cases, even interfere with the model at the stage of summarizing individual chunks.

To address this issue, a node filtering mechanism based on cosine similarity was implemented into the method. To eliminate low-informative or redundant chunks, a global check of cosine similarity between all intermediate summaries is performed at each hierarchy level. Chunks that are close in cosine similarity to previous ones are considered redundant and are not used for compiling the synopsis at the current level. Embeddings are obtained using SentenceTransformers (the USER-bge-m3 model), and processing is performed at high speed on a GPU. This modification aims to accelerate generation by removing potentially superfluous parts of information, thereby increasing the density of useful information in the final summaries.

Algorithm 1 Hierarchical method

Require: W - model context window, D - input text of length $L \gg W$, p_θ - model, C - chunk length
  Split D into chunks $c_1 \dots c_{\lceil \frac{L}{C} \rceil}$
  for $c_i = c_1 \dots c_{\lceil \frac{L}{C} \rceil}$ **do**
    $S_0 \leftarrow SummarizeChunk(p_\theta, c_i)$
  end for
  repeat
    $Groups \leftarrow GroupSummaries(S_l)$
    $\ell \leftarrow \ell + 1$
    for $g \in Groups$ **do**
      $S_l \leftarrow \{MergeGroup(p_\theta, g)\}$
    end for
  until $|S_l| = 1$
  return $S_l[1]$

Algorithm 2 Hierarchical method with node filtering

Require: W - model context window, D - input text of length $L \gg W$, p_θ - model, θ - similarity threshold, C - chunk length
  Split D into chunks $c_1 \dots c_{\lceil \frac{L}{C} \rceil}$
  $S_0 \leftarrow \{c_1 \dots c_{\lceil \frac{L}{C} \rceil}\}$
  repeat
    for $s_i \in S_l$ **do**
      $e_i \leftarrow Encoder(s_i)$
      $M_{ij} \leftarrow \dfrac{\mathbf{e}_i \cdot \mathbf{e}_j}{\|\mathbf{e}_i\| \, \|\mathbf{e}_j\|}$ ▷ Embedding matrix
      Compute the maximum similarity with previous summaries.
      $m_j = \max_{i<j} M_{ji}$
      $S_l \leftarrow \{s_i \mid m_i < \theta \;\; \textbf{or} \;\; i = 0\}$
    end for
    $Groups \leftarrow GroupSummaries(S_l)$
    $\ell \leftarrow \ell + 1$
    for $g \in Groups$ **do**
      $S_l \leftarrow \{MergeGroup(p_\theta, g)\}$
    end for
  until $|S_l| = 1$
  return $S_l[1]$

2.3. Text-Blueprint (Algorithm 3)

This method [7] is essentially a modification of the hierarchical method and focuses on building an intermediate outline before text generation. The outline is formed as a set of question-answer pairs, which enhances the controllability of the generation process and ensures the structured nature of the result. First the model creates a list of questions reflecting key events, themes, and characters. Then short answers are automatically generated for each question. This structure serves as a blueprint used to produce the final synopsis.

2.4. Blueprint method with question clustering (Algorithm 4)

The baseline blueprint implementation generates a question-answer outline for each chunk and at each merge level. With fiction, however, questions produced for different chunks may overlap and yield conflicting answers, which in turn confuses merging, making the synopsis less structured and complete. Moreover, generating an outline at every step slows the method and consumes extra LLM time. To reduce model calls and improve structure, we added clustering of questions using SentenceTransformers and KMeans.

Algorithm 3 Blueprint method

Require: W - model context window, D - input text of length $L \gg W$, p_θ - model, C - chunk length, R - length limit

Split D into chunks $c_1 \dots c_{\lceil \frac{L}{C} \rceil}$

for $c_i = c_1 \dots c_{\lceil \frac{L}{C} \rceil}$ **do**

  $b_i \leftarrow \mathit{GenerateBlueprint}(p_\theta, c_i)$

  $S_0 \leftarrow \{\mathit{SumWithBp}(p_\theta, b_i, c_i)\}$

end for

repeat ▷ Merging summaries

  $\mathit{Groups} \leftarrow \mathit{GroupSummaries}(S_l)$

  $\ell \leftarrow \ell + 1$

  for $g \in \mathit{Groups}$ **do**

    if $\mathit{Length}(g) > R$ **then**

      $b_i \leftarrow \mathit{GenerateBlueprint}(p_\theta, g)$

      $S_l \leftarrow \{\mathit{SumWithBp}(p_\theta, b_i, g)\}$

    else

      $S_l \leftarrow \{g\}$

    end if

  end for

until $|S_l| = 1$

return $S_l[1]$

Algorithm 4 Blueprint method with clustering

Require: W - model context window, D - input text of length $L \gg W$, p_θ - model, C - chunk length, R - length limit

Split D into chunks $c_1 \dots c_{\lceil \frac{L}{C} \rceil}$

for $c_i = c_1 \dots c_{\lceil \frac{L}{C} \rceil}$ **do**

  $b_i \leftarrow \mathit{GenerateBlueprint}(p_\theta, c_i)$

  $Q \leftarrow \{\mathit{ExtractQuestions}(p_\theta, b_i)\}$

end for

for $q_i \in Q$ **do**

  $E \leftarrow \{\mathit{Encoder}(q_i)\}$

end for

$K \leftarrow \mathit{KMeans}(E)$

for $k_i \in K$ **do**

  $q_i \leftarrow \mathit{Generalize}(p_\theta, k_i)$

  $Q \leftarrow \{q_i\}$ ▷ Build a global outline

end for

for $c_i = c_1 \dots c_{\lceil \frac{L}{C} \rceil}$ **do**

  $S_0 \leftarrow \{\mathit{SumWithBp}(p_\theta, b_i, c_i)\}$

end for

Summaries merge as in the Blueprint method, except the blueprint here is the single global outline Q.

3. Metrics

For an objective comparison of the described methods and models in the task of summarizing literary texts, four groups of metrics were used.

ROUGE-L [8] - based on the length of the longest common subsequence (LCS) between the generated synopsis S and the reference R.

$$\text{Precision} = \frac{\text{LCS}(S, R)}{|S|}, \tag{1}$$

$$\text{Recall} = \frac{\text{LCS}(S, R)}{|R|}, \tag{2}$$

$$\text{ROUGE-L} = \frac{2\ \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{3}$$

BERTScore [9]. For every token pair from prediction and reference we compute cosine similarity of their embeddings in USER-bge-m3. Then:

$$P = \frac{1}{|S|} \sum_{t \in S} \max_{u \in R} \text{sim}(e_t, e_u), \tag{4}$$

$$R = \frac{1}{|R|} \sum_{u \in R} \max_{t \in S} \text{sim}(e_u, e_t), \tag{5}$$

$$\text{BERTScore} = \frac{2\,P\,R}{P + R}, \tag{6}$$

where S is the reference text and R is the generated one; each sentence is encoded by USER-bge-m3, followed by cosine similarity.

Coverage of key questions (Coverage) - the proportion of pre-generated questions (from the reference) using Qwen3-235B-A22B [10] to which the model "answers" in its synopsis:

$$\text{Coverage} = \frac{\#\{q_i \colon P(\text{yes} \mid q_i, S) > 0.75\}}{N}, \tag{7}$$

where N is the total number of questions and $P(\text{yes} \mid q_i, S)$ is the probability that the answer to q_i is present in S, obtained with an LLM (Qwen3-235B-A22B [10]).

Answer similarity - the average cosine similarity between generated answers a_i^{pred} and reference answers a_i^{ref} to the same key questions:

$$\text{AnswerSimilarity} = \frac{1}{N} \sum_{i=1}^{N} \text{sim}\big(a_i^{\text{pred}},\, a_i^{\text{ref}}\big), \tag{8}$$

where `sim` is cosine similarity of embeddings computed with USER-bge-m3.

4. Experimental setup

All measurements were performed on the test portion of the dataset, selecting sources that did not exceed 800,000 characters. For all methods the summaries were limited to 500 words maximum. The input text was split into fixed-size chunks of 2,000 tokens. Tokenization used `AutoTokenizer` from `DeepPavlov/rubert-base-cased` [7] with default settings. To ensure reproducibility the random seed is fixed ($random\_seed = 42$).

In the **hierarchical method with node filtering**, a matrix of cosine similarities between the embeddings of the intermediate synopses is computed at each level to assess their redundancy. A similarity threshold is set at $\theta = 0.85$: if for a synopsis S_j there existed a previous synopsis S_i with a cosine similarity above this threshold, then S_j is discarded as redundant. This choice of threshold provides a compromise between preserving meaningful information and eliminating duplication, which empirically led to a noticeable reduction in the volume of intermediate representations without significant degradation in quality.

In the **blueprint method** with question clustering, the number of clusters for KMeans is chosen using a heuristically derived rule:

$$n_{\text{clusters}} = \max\Big(2,\ \Big\lceil \sqrt{N_{\text{questions}}} \Big\rceil\Big) \tag{9}$$

where $N_{\text{questions}}$ is the total number of questions generated across all chunks before clustering.

Runtime was measured as the average value (in seconds) of the generation time per book for each method across 100 books.

[7] `https://huggingface.co/DeepPavlov/rubert-base-cased`

5. Results

5.1. Models used

The experiments used the following large language models: RuadaptQwen2.5-7B-Lite-Beta [11], RuadaptQwen3-32BInstruct-v2 [11], DeepSeek V3 [12], Qwen3-235B-A22B [10], tpro [13] and yagpt5lite [14]. In all tables, models are ordered by size, and the best results within each parameter group are highlighted.

Table 2. Results by methods and models

Model	Metrics	Hierarchical	Blueprint	Hierarchical with node filtering	Blueprint with clustering
DeepSeek V3	bertscore	60.0 ± 3.1	58.0 ± 4.0	60.0 ± 2.9	58.4 ± 3.6
	rouge-l	13.7 ± 3.9	12.6 ± 4.6	13.5 ± 3.7	11.2 ± 3.9
	coverage	**53.57 ± 21.66**	40.19 ± 23.68	**45.00 ± 23.03**	**34.68 ± 23.77**
	similarity	42.38 ± 17.73	32.31 ± 19.33	35.64 ± 18.88	27.76 ± 19.75
	time	196.77 ± 187.85	315.67 ± 321.89	147.21 ± 146.4	132.60 ± 197.25
Qwen3-235B-A22B	bertscore	61.2 ± 3.0	61.6 ± 3.3	60.9 ± 2.7	59.3 ± 3.4
	rouge-l	14.9 ± 4.0	15.8 ± 4.5	14.8 ± 3.7	12.2 ± 3.6
	coverage	52.48 ± 20.79	**54.78 ± 21.16**	44.54 ± 23.03	30.19 ± 21.96
	similarity	41.68 ± 17.18	43.99 ± 17.54	35.67 ± 18.87	24.10 ± 17.62
	time	103.49 ± 97.30	230.35 ± 271.03	83.06 ± 102.05	158.30 ± 196.35
RuadaptQwen3-32B Instruct-v2	bertscore	57.3 ± 2.9	58.9 ± 3.6	57.7 ± 3.3	55.3 ± 3.3
	rouge-l	11.0 ± 2.4	10.6 ± 3.2	10.7 ± 2.4	7.8 ± 2.1
	coverage	33.12 ± 21.50	33.18 ± 22.83	32.19 ± 22.52	17.72 ± 15.23
	similarity	25.25 ± 16.94	26.21 ± 18.22	24.82 ± 17.74	13.97 ± 12.39
	time	218.30 ± 195.16	379.24 ± 500.40	166.79 ± 164.61	286.35 ± 395.97
tpro	bertscore	59.4 ± 3.0	59.0 ± 4.9	59.5 ± 3.3	58.2 ± 3.7
	rouge-l	13.8 ± 3.1	14.7 ± 4.9	13.5 ± 3.0	11.8 ± 3.9
	coverage	40.27 ± 20.23	40.83 ± 22.42	37.13 ± 20.72	26.03 ± 18.44
	similarity	31.77 ± 16.63	32.60 ± 18.57	29.44 ± 16.83	20.83 ± 15.26
	time	367.32 ± 324.49	592.39 ± 772.19	267.73 ± 253.34	247.59 ± 361.20
RuadaptQwen2.5-7B Lite-Beta	bertscore	55.4 ± 2.9	56.1 ± 4.9	55.8 ± 2.9	54.0 ± 4.0
	rouge-l	8.6 ± 2.5	10.1 ± 3.9	8.7 ± 2.5	7.7 ± 2.8
	coverage	19.66 ± 17.77	24.94 ± 21.08	20.31 ± 17.95	15.51 ± 14.83
	similarity	15.16 ± 14.11	20.03 ± 17.50	15.94 ± 14.39	12.23 ± 12.30
	time	68.86 ± 64.85	126.84 ± 145.74	53.59 ± 47.28	76.66 ± 91.78
yagpt5lite	bertscore	62.5 ± 3.5	61.1 ± 3.8	62.1 ± 3.2	61.5 ± 3.3
	rouge-l	16.9 ± 5.1	15.8 ± 5.1	16.4 ± 4.7	14.3 ± 4.4
	coverage	36.85 ± 19.40	33.17 ± 21.58	31.75 ± 20.06	24.28 ± 16.95
	similarity	29.69 ± 16.43	26.58 ± 18.13	25.60 ± 16.85	19.70 ± 14.29
	time	31.02 ± 28.51	113.34 ± 123.78	27.39 ± 28.05	42.15 ± 56.50

5.2. Findings

Tab. 2 shows metrics of automatic book summarization across models and methods. The best overall performance was achieved by Qwen3-235B-A22B: it delivered the highest coverage and answer similarity. At the same time, the hierarchical method with node filtering offered the best quality–time trade-off. It significantly sped up processing (e.g., almost 2× faster for DeepSeek V3), and compared to the blueprint method-which on average achieved the best metrics-it lagged only slightly. The exception was Qwen3-235B-A22B, which achieved its top results with the baseline blueprint. Experiments show that the hierarchical method with node filtering provides the best compromise between speed and quality.

5.3. Analysis and comparison

Table 3. Comparison of the best and worst generated summaries

Title	Text
A Sound of Thunder	… **The main character, Eckels**, a thrill-seeking and overconfident hunter, **pays** a huge sum of money **for the chance to travel 60 million years back in time to kill a Tyrannosaurus rex**. Before the journey, the guide Travis strictly warns him about the rules: **under no circumstances should anyone step off the anti-gravity Path** or interfere with the natural course of events, as even the slightest violation could catastrophically change the future.… Travis explains **the fragility of the temporal balance: even the death of a single mouse could wipe out entire species**, and thus alter human history. **The group tracks down a Tyrannosaurus**, marked with red paint — a sign that **its death will not affect the future**. However, at the sight of the giant predator **Eckels panics, steps off the Path, and accidentally crushes a butterfly**… Upon returning to 2055 … **the world has changed beyond recognition: the language is coarse, the atmosphere oppressive, and instead of the moderate President Keith, a cruel dictator, Deutscher, is in power.** Eckels realizes that **his carelessness triggered the “butterfly effect”** — the crushed insect set off a chain of events that distorted history. In despair, he begs to undo the mistake, but **Travis, understanding the irreversibility of the consequences, raises his rifle.** …
Kastrjuk	… The story takes place in a Russian village in early spring, where nature awakens, but people’s lives remain harsh and monotonous. **The main character — an old man named Semyon, nicknamed Kastrjuk**, is spending his final days in loneliness, **tormented by memories of his former strength** and regrets over his present frailty. **Once he was known as the best worker in the district**, but now, frail and forgotten, he is forced to stand aside while his fellow villagers work in the fields. … **Only in the evening, persuading his son to let him go to the night watch (to herd horses), does Kastrjuk find brief happiness**. Out in the open, among the children and under the starry sky, he feels almost young again. By the pond, a mare drinks water reflecting the sunset, while the **old man, looking at the Milky Way, whispers a prayer — as if he is reconnecting with the world and regaining lost harmony**. But it is only a fleeting solace: tomorrow he will again face hopeless loneliness and the realization of his own uselessness. …

The spread of QA metrics can be illustrated using the same model (DeepSeek V3) within the hierarchical method. Two summaries were chosen for the analysis: "A Sound of Thunder" and "Kastrjuk". In the first case the model scored high, answering all but one question, but in the second case the synopsis contained answers to only two out of eleven, leading to a low score. Tab. 3[8] shows the two synopses. For brevity only the main points that affected the final metric were highlighted. The "Kastrjuk" synopsis contains many lyrical digressions and stylistic details, making it hard to capture the essence, so the model gets distracted from key facts, whereas in "A Sound of Thunder" events are presented sequentially and clearly, with core plot elements explicitly listed, simplifying retrieval of important information. In the texts, bold marks plot-relevant fragments, while underlines indicate content that could be omitted.

Table 4. Comparison of models in synopsis generation using the "Blueprint" method

Model	Text
RuadaptQwen3	"The company *Time Safari* organizes paid excursions into the past for dinosaur hunting, using time machines capable of moving between eras. Clients are required to follow strict rules: to stay on the metal Path ...
tpro	"In the text, the main character, Eckels, goes on a time safari in order to kill a Tyrannosaurus rex. The company that organizes the safari guarantees only dinosaurs and strictly forbids hunters from stepping off the Path ... Mr. Travis, the safari guide, explains that even the destruction of a single mouse could lead to the extinction of all its descendants ...
DeepSeek V3	"**Summary by outline:** 1. **Eckels** — the hunter ... 2. **The company 'Time Safari'** organizes hunting in the past ... 3. **Travis** — the guide supervising the expedition. ...

Comparing models, DeepSeek V3 generally outperforms smaller models; however, within the blueprint method, in 30% of cases RuadaptQwen3-32B-Instruct-v2 performs best, and tpro in 43%. For reference, consider the synopsis for "A Sound of Thunder" generated with the blueprint method, with small excerpts shown in Tab. 4[8]. While the DeepSeek V3 synopsis resembles a numbered list of main events, the outputs from RuadaptQwen3-32B-Instruct-v2 and tpro are cohesive narratives that cover the key plot points.

Note that the best result overall was achieved by the blueprint method with the large model Qwen3-235B-A22B, as shown in Tab. 2. For comparison, on the story "Barbos and Zhulka", the hierarchical method with Qwen3-235B-A22B misclassified "Zhulka" as a horse rather than a dog. Also, DeepSeek V3 tends to strictly follow the blueprint template and produces a numbered list of key events and main characters, rather than a flowing synopsis, whereas Qwen3-235B-A22B writes plain text. Thus, the unmodified blueprint method delivered the best results when using the strongest available model - Qwen3-235B-A22B.

5.4. Time measurements

The results in seconds (average of three runs) are in Tab. 5. They confirm that modifications speed up generation.

Interestingly, ultra-large models such as Qwen3-235B-A22B and DeepSeek V3 showed higher speed than some 32B models. A key reason is the Mixture-of-Experts (MoE) architecture: during generation only a subset of parameters is active (e.g., 30B out of 600B), and such models are typically optimized further for throughput.

[8]Translated

Table 5. Runtime (seconds) for a text of 81,049 characters (11 chunks).

Model	Hierarchical	Hierarchical with node filtering	Blueprint	Blueprint with clustering
DeepSeek V3	237.83	72.42	292.80	268.75
Qwen3-235B-A22B	113.24	39.45	215.63	145.20
RuadaptQwen3-32BInstruct-v2	218.23	72.54	420.95	470.4
tpro	472.23	127.38	421.65	185.94
RuadaptQwen2.5-7B-Lite-Beta	84.64	25.70	103.66	78.99
yagpt5lite	34.17	14.08	99.70	27.26

Conclusion

We presented the first open dataset that pairs book texts with their synopses from the "Narodny Briefly" resource [4]. We proposed two improved LLM-based approaches to summarizing fiction: hierarchical with node filtering and blueprint method with clustering. The hierarchical method with node filtering speeds up generation with minimal quality loss, making it suitable for long works under limited model context.

Our comparative analysis shows that large models such as DeepSeek V3 and Qwen3-235B-A22B generally deliver higher QA coverage and more complete synopses than compact models, especially with hierarchical and blueprint methods. However, for certain text types and methods (e.g., baseline blueprint), more compact models such as RuadaptQwen3-32B-Instruct-v2 can be competitive at lower compute cost. Thus, model choice should balance available resources, quality requirements, and the nature of the processed texts.

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. BOOKSUM: A Collection of Datasets for Long-form Narrative Summarization / Wojciech Kryscinski, Nazneen Rajani, Divyansh Agarwal et al. // Findings of the Association for Computational Linguistics: EMNLP 2022 / Ed. by Yoav Goldberg, Zornitsa Kozareva, Yue Zhang. - Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022. - . - Pp. 6536-6558. https://aclanthology.org/2022.findings-emnlp.488/.

2. Echoes from Alexandria: A Large Resource for Multilingual Book Summarization / Alessandro Scir'e, Simone Conia, Simone Ciciliano, Roberto Navigli // Findings of the Association for Computational Linguistics: ACL 2023 / Ed. by Anna Rogers, Jordan Boyd-Graber, Naoaki Okazaki. - Toronto, Canada: Association for Computational Linguistics, 2023. - . - Pp. 853-867. https://aclanthology.org/2023.findings-acl.54/.

3. FABLES: Evaluating faithfulness and content selection in book-length summarization / Yekyung Kim, Yapei Chang, Marzena Karpinska et al. // First Conference on Language Modeling. - 2024. https://openreview.net/forum?id=YfHxQSoaWU.

4. *"Narodny Briefly".* Digital library of short summaries of literary works. `https://wiki.briefly.ru/` (accessed: 30.07.2025).

5. Library of works of art. `https://librusec.org//` (accessed: 30.07.2025).

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

7. Text-Blueprint: An Interactive Platform for Plan-based Conditional Generation / Fantine Huot, Joshua Maynez, Shashi Narayan et al. // Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations / Ed. by Danilo Croce, Luca Soldaini. - Dubrovnik, Croatia: Association for Computational Linguistics, 2023. - . - Pp. 105-116. https: //aclanthology.org/2023.eacl-demo.13/.

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

9. *BERTScore.* BUCKLEY C. Evaluating Evaluation Measure Stability //ACM SIGIR 2000 Proceedings. - 2000.

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

11. *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.

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

13. 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).

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