GigaChat Family: Efficient Russian Language Modeling Through Mixture of Experts Architecture

GigaChat team
SaluteDevices / Moscow
Correspondence: minkin.fyodor@gmail.com

Abstract

Generative large language models (LLMs) have become crucial for modern NLP research and applications across various languages. However, the development of foundational models specifically tailored to the Russian language has been limited, primarily due to the significant computational resources required. This paper introduces the GigaChat family of Russian LLMs, available in various sizes, including base models and instruction-tuned versions. We provide a detailed report on the model architecture, pre-training process, and experiments to guide design choices. In addition, we evaluate their performance on Russian and English benchmarks and compare GigaChat with multilingual analogs. The paper presents a system demonstration of the top-performing models accessible via an API, a Telegram bot, and a Web interface. Furthermore, we have released three open GigaChat models in open-source [1], aiming to expand NLP research opportunities and support the development of industrial solutions for the Russian language.

1 Introduction

The rapid advancement of generative large language models (LLMs) has significantly transformed the landscape of natural language processing (NLP), enabling innovative research and applications across multiple languages. However, developing foundation and post-trained models for the Russian language is still a significant challenge. This resource-intensive task hinders progress in the field and fails to address the cultural specifics of the Russian language and culture.

In response to this gap, we introduce the GigaChat family of Russian LLMs, created from scratch, which encompasses a variety of sizes, including both pre-trained and instruction-tuned versions. This paper describes our experience creating a model family based on the mixture of experts (MoE) architecture, the experiments in training such an architecture, and the description of the new tokenizer designed for the Russian language. Furthermore, we thoroughly evaluate the model's performance on Russian and English benchmarks and tests. This paper not only highlights the strengths of GigaChat in comparison to existing multilingual models but also offers a practical demonstration of our top-performing proprietary models through accessible interfaces such as an API, a Telegram bot, and a web application. By releasing three open versions of the GigaChat models as open-source resources, we aim to encourage further research in natural language processing (NLP) and support the ongoing development of industrial applications tailored to the Russian language.

Our contributions are as follows:

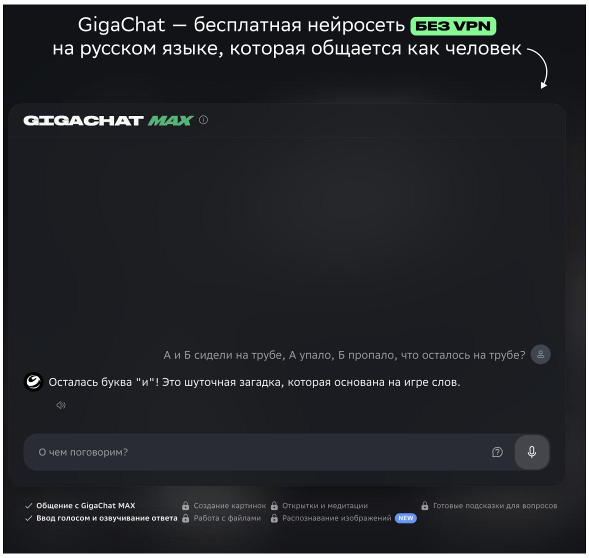
- We introduce the first family of foundation and post-trained models specifically designed for the Russian language, based on the Mixture of Experts (MoE) architecture. Three of these models are available in open-source (including their variations in int8 and bf16 formats) [2].
- We present experimental results and metrics on various benchmarks, demonstrating that our models are comparable to the state-of-the-art (SOTA) models of similar sizes among existing open-source models.
- We also share our experiments with the MoE concentration mechanism and provide code for MoE expert control.
- We release the Telegram bot and the System demo Web interface [3] for our most advanced model.

[1] https://huggingface.co/ai-sage

[2] Under the MIT license, commercial/non-commercial use, re-hosting, and fine-tuning are permitted without restrictions.

[3] The video demonstration is available on YouTube.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 93–106
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics


GigaChat — бесплатная нейросеть БЕЗ VPN
на русском языке, которая общается как человек
GIGACHAT MAX
О чем поговорим?
✓ Общение с GigaChat MAX
✓ Ввод голосом и озвучивание ответа
Создание картинок
Работа с файлами
Открытки и медитации
Распознавание изображений NEW
Готовые подсказки для вопросов

А и Б сидели на трубе, А упало, Б пропало, что осталось на трубе?

Осталась буква "и"! Это шуточная загадка, которая основана на игре слов.

Figure 1: A screenshot of the system demo for the open Web demo of the GigaChat Max. To access more features of GigaChat, registration is required.

2 Related Work

MoE architecture Sparse MoE models have gained significant attention in recent years (Cai et al., 2024) due to their capacity for efficient scaling while maintaining computational effectiveness. The foundational work Shazeer et al. (2017) introduced the sparse MoE layer, demonstrating its effectiveness in training large-scale language models in application to LSTM-based architectures. More recently, Mixtral (Jiang et al., 2024) set a new SOTA for MoE-based LLMs with 47 billion total parameters but only 13 billion active parameters, outperforming dense models such as LLaMA 2 70B. Another notable contribution, DeepSeek MoE (Dai et al., 2024), explored modifications to MoE architecture by increasing the number of experts while reducing their sizes and adding shared experts that are always activated, improving expert specialization and overall model performance.

Russian generative LLMs. Pre-trained open models for the Russian language remain scarce. The work of Zmitrovich et al. (2024) introduces a collection of 13 Russian Transformer-based language models, which include encoder architectures (ruBERT, ruRoBERTa, ruELECTRA), decoder architectures (ruGPT-3), and encoder-decoder architectures (ruT5, FRED-T5). However, even the latest generative models, such as ruGPT-3.5 [4], demonstrate subpar performance on benchmarks like the MERA SOTA instruction models (Fenogenova et al., 2024). Most SOTA models, mainly those available as open-source, are either English-based or multilingual (e.g., Qwen, Mistral, and their Russian-adapted variants [5]), which have been post-trained on Russian texts. Among the Russian proprietary models, only a few exist, such as Cotype by MTS AI and the YandexGPT family [6], both of which lack transparency regarding their training methodologies and architectural details and are not fully pre-trained on Russian texts. To bridge this gap and address the need for high-performing, Russian-focused generative models that rival their multilingual counterparts, we introduce the **GigaChat** family.

3 GigaChat Family

3.1 Overview

The GigaChat family is the first collection of foundation and post-trained models specifically designed and pre-trained from scratch for the Russian language. The initial version [7] of the GigaChat family employs the MoE architecture that we now release in open-source: base model, instructed version, and aligned with Direct Preference Optimization (DPO) (Rafailov et al., 2023). Advanced proprietary models — Lite, Pro, and MAX — are continually updated and accessible through a user API and a dedicated Telegram bot, ensuring ongoing improvements and enhanced usability.

3.2 System demo

The GigaChat models support a versatile user interaction system, offering free access through a Telegram bot and a Web demo interface [8]. The Web version contains the advanced proprietary model, GigaChat Max [9] Max allows users to engage in conversations by submitting text prompts in both Russian and English, all within a predefined character limit. The screenshot in Figure 1 illustrates the interface of the free version, which offers two primary features: 1) chatting capability and 2) audio ASR input via GigaAM [10]. The full version

[4] https://mera.a-ai.ru/ru/submits/11273

[5] T-pro-it-1.0, RuadaptQwen2.5-32B-Instruct, Zero-Mistral-Small-24B

[6] https://ya.ru/ai/gpt-4

[7] It is noteworthy that the three open models were previously also available through an API, and they continue to receive regular enhancements and improvements.

[8] https://giga.chat/

[9] The API for the system demo is updating to the latest versions; we are reporting the version of GigaChat 2 as of March 2025.

[10] https://github.com/salute-developers/GigaAM

of the interface is available only after registration and includes additional functionalities such as file processing and predefined prompts for various use cases.

The key features of the Telegram bot (@gigachat_bot) include an interactive chatbot that engages users in conversation and the capability to invoke the Kandinsky model (Arkhipkin et al., 2024) for image generation based on user prompts. Additionally, the bot offers a variety of predefined user prompts and can process files.

3.3 Open models

In this section, we explain the choice of the architecture and all the parts of the models creation, starting with the pre-trained base model.

3.3.1 Models architecture

The GigaChat-A3B-base model leverages a MoE architecture with 20 billion total parameters, of which approximately 3.3 billion are activated per forward pass (see Table 1). In our experiments using the same data, the MoE design demonstrates significant efficiency gains, including double the training speed and a 40% reduction in inference latency compared to similarly sized dense models, such as 8B LLaMA 3.

The efficiency stems from block-sparse computation using optimized STK Triton kernels rather than Megablocks and selective activation checkpointing, reducing computational requirements by 40% versus a 7B dense model while processing 1 trillion tokens. These optimizations eliminate the need for expert parallelism while maintaining model performance. The architecture replaces standard MLP blocks with MoE layers (except the first layer, which uses a gated MLP due to token distribution challenges). Each MoE block employs multiple experts and an unnormalized router to promote specialization, following insights from DeepSeek MoE. The intermediate dimension is expanded to 14,336 (as in Mistral 7B (Jiang et al., 2023)) to enhance capacity, and experts are shared across layers to improve parameter efficiency. This combination of sparse computation, expert sharing, and optimized routing enables high throughput with reduced resource consumption, making the model scalable for large-scale training and inference.

Section A.1 of the Appendix describes the training process details.

3.3.2 Pre-train

The base model was trained using a constant multi-step learning rate scheduler with warmup. The scheduler included a warmup period of 2000 batches, after which four learning rate decay steps took place at 30%, 60%, 90%, and 98% of the total training duration. At these milestones, the learning rate was reduced by multiplying by factors of 0.25, 0.0625, 0.015625, and 0.00390625 (i.e., $(0.25)^1$, $(0.25)^2$, $(0.25)^3$, and $(0.25)^4$, respectively). The initial learning rate was set to 1e-4. The training process used a global batch size of approximately 16 million tokens (2048 sequences with 8192 tokens per sequence) and accumulated 9.5 trillion tokens across 8k pre-training steps.

After the initial training step, we conducted a context extension in two stages: first to 32K and then to 128K. To improve performance with the extended context, we adjusted the base for RoPE embeddings (Su et al., 2024) using the ABF approach (Xiong et al., 2023). For each training stage, we utilized the following values: 10K for the initial 8K context, 300K for 32K, and 1.4M for 128K. The model employed a constant learning rate scheduler with predefined drops during training. Continuous training in the long context used the final learning rate from the 8K context, maintaining this rate throughout both training stages.

To evaluate the adaptation of the model, we used English PassKey[11] and LongBench (v1) (Bai et al., 2023). The LongBench evaluation set the maximum sample length according to the target context length, while the PassKey evaluation ranged from 8,000 to 128,000 tokens. Overall, the extension involved about 1.8 trillion tokens and tens of thousands of steps, but evaluations showed that it could be accomplished in just a few thousand steps.

3.3.3 Post-train

Each model trained on various versions of the post-train data (see the Section 4.2) has its own hyperparameters, so in addition to several checkpoints within a single training (the model state is saved twice per epoch), we run several training iterations to select the best model from all of them. The final hyperparameters for the best open models are presented in Table 2.

It is important to note that the final checkpoint does not always yield the highest performance metrics. In some versions of the dataset, the optimal

[11] passkey.py

Model	Architecture	Parameters	Hidden Layers	Shared experts	Routed experts	KV Heads	Heads	Context Length
GigaChat-A3B-base	MoE	20B	28	2	64	8	16	131k

Table 1: Summary of the GigaChat-A3B-base model architecture configurations.

model is achieved during the middle of the training process, while in others, it may be reached closer to the end. Therefore, selecting the best model involves a variety of heuristics based on specific needs. We choose from the metrics described in Section 5.1.

3.3.4 DPO

In developing the GigaChat-A3B-instruct 1.5, we identified key issues with DPO, such as its focus on widening the gap between good and bad responses rather than improving accuracy, leading to hallucinations and instability. It also overlooks the importance of common token prefixes. To tackle these issues, we proposed modifications to the DPO loss function (Equation 1), including unique weighting factors that prioritize enhancing good responses over suppressing bad ones, particularly concerning shared prefixes. We also added a normalized negative log-likelihood term relative to a reference model to stabilize loss ratios.

$$\text{loss} = \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Bigg[-\log\sigma\Bigg(\beta_w \log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta_l \log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\Bigg) + \log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)}\Bigg] \tag{1}$$

3.3.5 Optimal Tokenization

A new tokenizer has been developed to enhance the text encoding for Cyrillic words, programming languages, and LaTeX. We improve accuracy in handling code data by including common keywords and supporting spaces, tabs, and line breaks. High-frequency terms from LaTeX and programming are incorporated to minimize fragmentation, ensuring efficient tokenization of essential syntax elements. The selection of tokenizers was optimized to maximize the average length of tokens within domain-specific datasets.

Training Process We employed an iterative refinement process on a training dataset to maximize tokenization efficiency. Our focus was to ensure balanced performance across multiple domains, including programming languages such as C, Java, C#, LaTeX markup, and general language corpora. The primary language of concern was Russian, with additional support for English and European languages, Arabic, Uzbek, and Kazakh. This effort primarily aims at the Russian community and the support of rarer languages, for which high-quality language models are scarce.

For training, we leveraged the Hugging Face Byte-Pair Encoding (BBPE) algorithm, conducting multiple experiments to generate candidate tokenizers. During these experiments, we gradually adjusted the proportion of texts from different domains (Russian, English, other languages, and code). This process resulted in a large number of candidate tokenizers (more than a hundred). From these, we selected the tokenizer that demonstrated the best performance compared to other tokenizers. The tokenizer training data and tokenizer comparison details are presented in Appendix A.3.

4 Data

4.1 Pre-train data

We aggregate diverse textual sources to construct a robust pre-training dataset, ensuring a balance between linguistic richness, domain-specific knowledge, and data quality. The dataset comprises 1) web-scraped texts, 2) high-quality publications, 3) programming code, and 4) synthetic data. The data statistic is presented in Table 3. We implement precise deduplication across all languages and sources to ensure corpus integrity and reduce redundancy. Additionally, we enhance the dataset for English-language data through MinHash deduplication (Broder, 1997), which effectively minimizes semantic duplicates.

Web data To construct a high-quality pre-training corpus, we leverage Common Crawl web dumps from 2017-2023 (Penedo et al., 2023b), (Li et al., 2024) and used a lightweight classifier (Joulin et al., 2016) to extract multilingual texts in Russian, English, Kazakh, Uzbek, Portuguese, and Arabic. These texts were further classified using LLMs and specialized models to identify educational [12] and high-value informational con-

[12] https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu

model	optimizer	scheduler	params of scheduler	hyperparameters
GigaChat-A3B-instruct	AdamW	Constant	custom drop	-
GigaChat-A3B-instruct 1.5	AdamW	Cosine	warmup: 200 steps, max steps: 7900	betas (0.9, 0.95), eps: 1.0e-8

Table 2: Hyperparameters of the post-training models during the training.

tent (Li et al., 2024), resulting in 4.4T tokens of curated data. The dataset is predominantly English (63.76%) and Russian (26.49%), with Portuguese (7.80%) and Arabic (1.90%), and less than 0.06% combined for Kazakh and Uzbek.

High-Quality Textual Sources We incorporate high-quality textual content from open-access books and academic articles, processed using advanced optical character recognition for accurate extraction. This adds 630B tokens of linguistic data. Additionally, we enrich the dataset with scientific and encyclopedic sources like arXiv, Wikipedia, and PubMed [13], improving reasoning and factual consistency in the pre-training model.

Programming Code Corpus We use the StarCoder2 (Lozhkov et al., 2024) dataset alongside a curated set of open-source software code to create a diverse programming dataset that complies with licensing requirements. Machine learning models filter out low-quality code, yielding a 230B token subset ideal for code generation and understanding tasks.

Synthetic data Real-world data is limited by bias, privacy, and scarcity, while synthetic data is scalable and controlled. Phi-4 (Abdin et al., 2024) demonstrates that synthetic data pre-training improves performance on reasoning and STEM benchmarks. For math and programming, we built a Numina-inspired pipeline (Jia et al., 2024) that expands seed mathematical problems by solving them multiple times and filtering via majority vote and threshold. We also created high-quality synthetic code tasks (complex Python problems with documentation, explanations, and assertions) with structured prompts and diversified them using personas (Ge et al., 2024) and lipograms [14].

Data source	Unique Tokens	Seen Tokens
Web	4.4T	5.6T
HQ Sources	630B	1.3T
Code	230B	1.3T
Synthetic data	9B	81B

Table 3: Pre-train data distribution.

4.2 Post-train data

Clean training data is essential during the post-training phase. All supervised fine-tuning dialogues are annotated by professional AI trainers who evaluate responses based on criteria like adherence to instructions, context awareness, factual accuracy, and safety. We created the Dialog Creation annotation project on the crowdsourcing platform Tagme [15] to generate diverse dialogs across various domains while maintaining high data quality standards. AI trainers select the best responses from different model variants, using metadata for dataset balancing and error analysis to enhance model performance. To overcome the challenge of models retaining information from rare documents, we improved our model's memory and retrieval abilities through Retrieval-Augmented Generation following the experiments of the Grattafiori et al. (2024). This approach generates domain-specific training data from the pre-training corpus, enhancing contextual understanding.

Thus, the post-training of the open GigaChat-A3B-instruct model comprises about 250k items in the following proportion of data sources described in Table 4.

Domain	Proportion
chats	10%
long context (books)	4%
code	4%
science	16%
general world knowledge (web)	34%
translations	1%
text editing	12%
business specifics	3%
functions / api	16%

Table 4: Post-training proportion of the task domains and instructions in the GigaChat-A3B-instruct.

[13] https://pubmed.ncbi.nlm.nih.gov/download/
[14] https://en.wikipedia.org/wiki/Lipogram

Benchmark	Shots	GigaChat-A3B-instruct	GigaChat-A3B -instruct 1.5	Qwen 2.5	T Lite	Llama 3.1	GigaChat2 Pro	GigaChat2 MAX
GSM8K	5	0.764	0.774	0.895	0.882	0.789	0.95	**0.956**
MATH	4	0.462	0.393	0.704	0.592	0.329	0.752	**0.773**
HumanEval	0	0.329	0.378	0.854	0.799	0.683	**0.915**	0.871
MBPP	0	0.385	0.441	0.820	0.759	0.725	0.862	**0.894**
MMLU EN	5	0.648	0.650	0.710	0.718	0.682	0.821	**0.86**
MMLU RU	5	0.598	0.600	0.632	0.626	0.569	0.775	**0.805**
MMLU PRO EN	5	0.348	0.357	0.565	0.509	0.443	0.644	**0.667**
RUBQ	0	0.675	0.688	0.373	0.583	0.484	0.658	**0.723**
WINOGRANDE	4	0.750	0.762	0.636	0.670	0.624	0.796	**0.832**
CyberMetric	0	0.798	0.791	0.787	0.883	0.796	**0.84**	0.832
IFEval	0	0.411	0.433	0.819	0.730	0.812	0.837	**0.899**

Table 5: Comprehensive comparison of models across Russian/English benchmarks. The best result in each column is highlighted in bold, the best result in the same model size is underscored.

Model	Total	RWSD	ruModAr	USE	MaMuRAMu	ruHHH		
						Honest	Helpful	Harmless
Human Benchmark	0.852	0.835	0.942	0.701	0.796	0.705	0.797	0.948
Claude 3.7 Sonnet	**0.682**	**0.788**	0.919	0.536	**0.89**	0.82	**0.864**	0.931
GigaChat 2 MAX	0.67	0.642	**0.963**	**0.581**	0.864	0.803	0.831	**0.948**
Gemini 1.5 Pro	0.675	0.627	0.707	0.433	0.868	0.836	0.797	0.931
GPT-4o	0.642	0.496	0.729	0.457	0.874	**0.852**	0.729	0.862
DeepSeek V3	0.677	0.612	0.718	0.499	0.882	0.803	0.763	0.793
Phi-3.5-MoE-Inst	0.487	0.465	0.464	0.199	0.726	0.656	0.644	0.81
GigaChat 2 Pro	**0.649**	0.665	**0.943**	**0.534**	0.831	0.803	0.814	0.897
Mixtral-8x22B-Inst	0.486	0.473	0.523	0.269	0.747	0.836	**0.881**	**0.966**
Qwen2.5-72B-Inst	0.601	**0.715**	0.665	0.32	0.849	**0.869**	0.831	0.897
Llama-3.1-405B-Inst	0.59	0.677	0.573	0.357	**0.868**	0.803	0.864	0.759
RuadaptQwen2.5-7B	0.536	0.465	0.492	0.162	0.751	0.738	0.78	0.776
GigaChat 2	0.541	0.369	**0.854**	**0.361**	0.766	0.754	0.814	**0.931**
T-lite-it-1.0	0.552	0.535	0.493	0.147	0.775	0.689	0.797	0.862
GigaChat-A3B-instruct	0.512	0.535	0.853	0.325	0.728	0.689	0.78	0.759
GigaChat-A3B-instruct 1.5	0.511	0.512	0.84	0.32	0.728	0.689	0.831	0.793
gemma-3-27b	**0.567**	**0.588**	0.626	0.328	**0.797**	**0.82**	**0.864**	0.914
gemma-2-9b	**0.453**	0.558	0.592	**0.154**	**0.689**	0.574	0.627	0.552
GigaChat-A3B-base	0.422	0.508	**0.608**	0.127	0.675	0.574	0.593	0.552
Llama-3.2-3B	0.362	0.477	0.592	0.075	0.528	0.41	0.542	0.483
Yi-1.5-9B-32K	0.428	0.569	0.516	0.12	0.516	**0.59**	**0.661**	**0.621**
Qwen1.5-7B	0.374	0.558	0.485	0.056	0.52	0.541	0.627	0.603
Mistral-7B-v0.1	0.404	**0.581**	0.517	0.107	0.585	0.574	0.559	0.552
ruGPT-3.5	0.213	0.462	0.001	0.082	0.226	0.459	0.475	0.483

Table 6: MERA benchmark results. The model's descriptions are available in the MERA leaderboard

5 Evaluation

5.1 Benchmarks

For the evaluation of the models, we use various common benchmarks in English and Russian that assess skills such as Mathematics Performance (GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021)), Coding Ability (HumanEval (Chen et al., 2021), MBPP [16]), General Knowledge (MMLU EN (Hendrycks et al., 2020), MMLU RU [17], MMLU PRO (Wang et al., 2024), RUBQ (Korablinov and Braslavski, 2020), WINOGRANDE (Sakaguchi et al., 2021)), Cybersecurity Knowledge (CyberMetric (Tihanyi et al., 2024)), and Instruction Following (IFEval (Zhou et al., 2023)). Table 5 presents a comprehensive performance comparison between open versions of GigaChat models and other open post-trained LLMs of compatible sizes (Llama 3.1 8b [18], Qwen 2.5 7 [19], and T-Lite [20]) across benchmarks. As the benchmark was created specifically for the Russian language, we present the assessment of pre-training and instructing models on the benchmark MERA (Fenogenova et al., 2024). For all tests, the LM Evaluation Harness framework [21] was used.

[15] https://tagme.sberdevices.ru/

[16] https://github.com/google-research/google-research/tree/master/mbpp

[17] https://mera.a-ai.ru/ru/tasks/9

[18] https://huggingface.co/meta-llama/Llama-3.1-8B

[19] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

[20] https://huggingface.co/AnatoliiPotapov/T-lite-instruct-0.1

[21] https://github.com/EleutherAI/lm-evaluation-harness

5.2 Results

English Benchmarks The GigaChat-A3B-instruct and GigaChat-A3B-instruct 1.5 models (3.3B active parameters) show a balanced trade-off between scale and performance against larger 7–8B counterparts (Qwen2.5 7B, Llama 3.1 8B, T-Lite). While mathematical (-14% GSM8K, -34% MATH) and programming (-46% MBPP, -55% HumanEval) gaps reflect parameter limitations, they excel in reasoning (+15% RUBQ, +12% WINOGRANDE) and retain competitiveness in MMLU (-5% to -8%). Challenges in high-difficulty MMLU PRO (-36%) and instruction following (-47% IFeval) persist, though DPO optimization yields targeted improvements. For the CyberMetric benchmark, new models also show competitive results, being 11% lower than the leader. Concerning GigaChat 2 MAX, GigaChat 2 Pro, the models show the best scores for all benchmarks, slightly falling short only on CyberMetric (-5%).

Russian Benchmarks Designed for Russian-language proficiency, the models (GigaChat 2 MAX, GigaChat 2 Pro, GigaChat 2) achieve near-state-of-the-art results on MERA benchmark (±2–7%) and dominate specialized tasks: ruModAr (+4% to +29%) and USE (+7% to +33%) highlight strengths in logic and complex comprehension. Coreference resolution (RWSD: -7% to -18%) and advanced reasoning (MaMuRAMu: -3% to -4%) show room for growth, yet performance remains competitive against both frontier models (e.g., GPT-4) and mid-tier alternatives. GigaChat-A3B-instruct and GigaChat-A3B-instruct 1.5 show a performance close to GigaChat 2. GigaChat-A3B-base reaches the level of best 9 billion pre-train models trailing by 12-20% on RWSD and USE, by 2% on MaMuRAMu, leading by 2% on ruModAr. Concerning the ruHHH dataset aimed at scoring the model's ability to determine the Honest, Helpful and Harmless behavior all GigaChat models show nearly the highest results among the same tier models: GigaChat 2 MAX, GigaChat 2 Pro, GigaChat 2 show the best or nearly the best scores for Harmless while being slightly behind the leaders for Honest and Helpful (-9% to -4%); GigaChat-A3B-base remains competitive against the other 3B–13B models (-12% to -3%); GigaChat-A3B-instruct, GigaChat-A3B-instruct 1.5 show close scores while demonstrating that DPO may help determine Helpful behavior better (+7% compared to without DPO).

6 Conclusion

We present the GigaChat family of LLMs, which is the only model developed from scratch during the pre-training stage specifically for the Russian language. By employing the MoE architecture and a specialized tokenizer, we have developed models that effectively address Russian linguistic and cultural nuances while achieving competitive performance against leading benchmarks. Our open-source release of three GigaChat models and user-friendly interfaces like a Telegram bot and a Web application for the frontier models aims to encourage further research and industrial applications in Russian NLP. The contributions outlined, including the introduction of Russian-focused models and experimental results, reflect our commitment to enhancing the field. By providing these resources to the community, we hope to foster innovation and collaboration in developing inclusive and effective language technologies for Russian-speaking users.

Ethical Statement

Possible Misuse Our research should not contribute to creating content that negatively impacts individual or community well-being. This includes the following restrictions: (i) involvement in legislative applications or censorship, (ii) dissemination of disinformation or infringement on the right to access information, (iii) dehumanizing or misrepresenting individuals or their religions, cultures, or beliefs, and (iv) promoting harmful or discriminatory content. To address this issue, the models' API format includes a censorship filter to mitigate inappropriate content that could pose potential risks.

Biases and data quality The pre-training data for all the models includes a wide range of content from Russian and English internet sources, which may introduce various stereotypes and biases. Thorough evaluations of these models are crucial to identifying potential vulnerabilities when applied to data outside their training domain.

Energy Efficiency and Usage We compute the CO_2 emissions from training our LLMs as Equation 2 (Strubell et al., 2019):

$$CO_2 = \frac{PUE * kWh * I^{CO2}}{1000} \quad (2)$$

The resulting number of the CO_2 for the open models is presented in Table 7. 251k kg of CO_2 is approximately equivalent to a round-trip flight

Model	CO_2 (kg)
GigaChat-A3B-base	251k
GigaChat-A3B-instruct	253k
GigaChat-A3B-instruct 1.5	255k

Table 7: CO_2 emissions of the models training.

from New York to London emits 1,600 kg of CO_2 per passenger.

Limitations

Lack of Reasoning Capabilities The models do not exhibit advanced reasoning abilities (like the models like DeepSeeek R1), which may restrict its effectiveness in tasks requiring complex problem-solving or logical inference.

Alignment Preferences The models have been specifically aligned to generate long and aesthetically pleasing chat responses. While this may appeal to some users, others might find such responses verbose or less practical for their needs.

Tokenizator The effectiveness of the trained tokenizer and the trained LMs is highly dependent on the quality and size of the corpus used. A limited or biased corpus can lead to suboptimal tokenization and model performance, potentially missing critical linguistic nuances and specific domain cases, such as characters from formal or other languages.

Reproducibility Issues Due to the use of closed pre-training, fine-tuning, and DPO datasets for proprietary models, the results cannot be independently replicated or verified. This lack of transparency may inhibit further research and validation efforts. However, we are open-sourcing three versions of the MoE-based GigaChat, and we hope this will encourage further research in Russian.

Acknowledgments

We extend our deepest gratitude to all members of the GigaChat Team for their tireless efforts and dedication to this ambitious project. Their collective expertise and collaboration were instrumental in bringing this research to life. We are especially thankful to Sergey Markov and the R&D team for their invaluable feedback and contributions during the early stages of this work. Our sincere appreciation also goes to Andrey Belevtsev for his visionary insights and strategic guidance, which helped shape the direction of this project. We sincerely acknowledge Anton Chigin and Sergei Galustian for their contributions to runtime and architecture development. Additionally, we acknowledge the many technical teams whose infrastructure and engineering excellence enabled the development, hosting, and deployment of the system for users.

Author Contributions

- *Administration and Supervision*: Fedor Minkin
- *Pre-training Team. Data*: Ivan Baskov, Valeriy Berezovskiy, Dmitry Kozlov, Ainur Israfilova, Lukyanenko Ivan
- *Pre-training Team. Training*: Gregory Leleytner, Evgenii Kosarev, Mamedov Valentin
- *Supervised Fine-Tuning (SFT) Team. Training*: Gregory Leleytner, Emil Shakirov, Smirnov Daniil, Mikhail Kolesov, Kolodin Egor, Aleksandr Proshunin
- *Supervised Fine-Tuning (SFT) Team. Data*: Nikita Savushkin, Eldar Damirov, Daria Khomich, Daria Latortseva, Sergei Porkhun, Yury Fedorov, Oleg Kutuzov, Polina Kudriavtseva, Sofiia Soldatova, Stanislav Pyatkin, Dzmitry Menshykh, Grafov Sergei, Karlov Vladimir, Ruslan Gaitukiev, Arkadiy Shatenov
- *Evaluation and Metrics*: Emil Shakirov, Smirnov Daniil, Artem Orlov, Alena Fenogenova
- *Tokenization*: Sergei Averkiev
- *Expert Interpretations*: Ilya Shchuckin
- *Research Supervision and Coordination*: Alena Fenogenova

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. 2024. Phi-4 technical report. *Preprint*, arXiv:2412.08905.

Vladimir Arkhipkin, Viacheslav Vasilev, Andrei Filatov, Igor Pavlov, Julia Agafonova, Nikolai Gerasimenko, Anna Averchenkova, Evelina Mironova, Anton Bukashkin, Konstantin Kulikov, Andrey Kuznetsov, and Denis Dimitrov. 2024. Kandinsky 3: Text-to-image synthesis for multifunctional generative framework. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 475–485, Miami, Florida, USA. Association for Computational Linguistics.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Andrei Z. Broder. 1997. On the resemblance and containment of documents. *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pages 21–29.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *Preprint*, arXiv:2407.06204.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Damai Dai, Chengqi Deng, Chenggang Zhao, Runxin Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Annual Meeting of the Association for Computational Linguistics*.

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.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. 2024. Scaling synthetic data creation with 1,000,000,000 personas. *Preprint*, arXiv:2406.20094.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal

Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

L. I. Jia, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [https://github.com/project-numina/

aimo-progress-prize](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L'elio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *ArXiv*, abs/2401.04088.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *ArXiv*, abs/1607.01759.

Vladislav Korablinov and Pavel Braslavski. 2020. Rubq: A russian dataset for question answering over wikidata. In *The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II 19*, pages 97–110. Springer.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Kumar Guha, Sedrick Scott Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean-Pierre Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham M. Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke S. Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldani, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. 2024. Datacomp-lm: In search of the next generation of training sets for language models. *ArXiv*, abs/2406.11794.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*.

Ziyue Li and Tianyi Zhou. 2024. Your mixture-of-experts llm is secretly an embedding model for free. *Preprint*, arXiv:2410.10814.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu, Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan L. Risdal, Jia Li, Jian Zhu, Terry Yue Zhuo, Evgenii Zheltonozhskii, Nii Osae Osae Dade, W. Yu, Lucas Krauss, Naman Jain, Yixuan Su, Xuanli He, Manan Dey, Edoardo Abati, Yekun Chai, Niklas Muennighoff, Xiangru Tang, Muhtasham Oblokulov, Christopher Akiki, Marc Marone, Chenghao Mou, Mayank Mishra, Alexander Gu, Binyuan Hui, Tri Dao, Armel Randy Zebaze, Olivier Dehaene, Nicolas Patry, Canwen Xu, Julian J. McAuley, Han Hu, Torsten Scholak, Sébastien Paquet, Jennifer Robinson, Carolyn Jane Anderson, Nicolas Chapados, Mostofa Patwary, Nima Tajbakhsh, Yacine Jernite, Carlos Muñoz Ferrandis, Lingming Zhang, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2024. Starcoder 2 and the stack v2: The next generation. *ArXiv*, abs/2402.19173.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023a. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra-Aimée Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023b. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only. *ArXiv*, abs/2306.01116.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Norbert Tihanyi, Mohamed Amine Ferrag, Ridhi Jain, Tamas Bisztray, and Merouane Debbah. 2024. Cybermetric: a benchmark dataset based on retrieval-augmented generation for evaluating llms in cybersecurity knowledge. In *2024 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 296–302. IEEE.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset [online]. 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. 2023. Effective long-context scaling of foundation models. *Preprint*, arXiv:2309.16039.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Dmitry Zmitrovich, Aleksandr Abramov, Andrey Kalmykov, Vitaly Kadulin, Maria Tikhonova, Ekaterina Taktasheva, Danil Astafurov, Mark Baushenko, Artem Snegirev, Tatiana Shavrina, et al. 2024. A family of pretrained 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.

A Appendix

A.1 Training details

We used a mixed precision training methodology (bfloat16 for most operations and fp32 for critical components, such as the router). The complete training process accumulated approximately 10 trillion tokens, with the final annealing phase comprising 40 billion tokens of pre-trained data described in Section 4.1.

We tackled communication bottlenecks in large-scale distributed training environments with over 256 GPUs by increasing batch size instead of adding more devices with the same workload. This strategy allowed for overlapping communication and computation, minimizing idle time and enhancing training throughput. The sparse computation patterns of the MoE architecture, along with a moderate hidden size, enabled us to significantly increase the batch size per device while staying within memory limits.

Throughout the training process, we systematically monitored expert utilization and router confidence using entropy-based metrics: $H\_utilization$ (quantifying token distribution between experts) and $H\_sparsity$ (measuring router confidence). We analyzed token distribution among experts and monitored $top-k$ router scores, identifying several critical issues: *expert collapse phenomena* (experts receiving minimal token assignments), *disproportionate token* processing by specific experts, and *router uncertainty* indicated by consistently low confidence scores. These metrics guided our hyperparameter optimization, especially for the auxiliary load balancing loss for uniform expert utilization. Visualizing expert utilization patterns offered insights that shaped our decision to implement a standard Gated MLP in the first layer.

A.2 Ablation study: Expert interpretations

During the experiments on the model architecture, we analyze router behavior to investigate if experts in GigaChat-A3B-base, specialize in specific domains such as math, medicine, and code. To do this, we constructed embeddings for a subset of the Pile (Gao et al., 2020) dataset [22] using router activations. Each embedding emb is a matrix of size $l \times e$, where l is the number of MoE layers and e is the number of experts in one layer (not including shared experts). Each sample emb_{ij} is calculated as the number of activations of expert j in layer i normalized by the length of the sample in tokens.

We clustered the embeddings with UMAP and HDBSCAN, revealing that samples grouped by domain (Fig. 2), indicating that router decisions encode domain information. This aligns with the findings in (Li and Zhou, 2024), where MoE models provided effective embeddings without the need

[22] We use the version `https://huggingface.co/datasets/monology/pile-uncopyrighted` of the set where all copyrighted content was removed

for fine-tuning. Clusters were identified in sports, cooking, biology, and programming domains.

We created domain-specific embeddings by averaging values within clusters. These embeddings help differentiate experts in those fields. To identify significant experts, we set values below $\frac{3}{e}$ to zero, keeping only those at least three times greater than expected. We then use filtered embeddings to guide our model toward specific domains by adjusting router activations to prioritize selected experts.

We found that this method allows us to control generation flow [23]; for example, using sports-related embeddings led to texts focused on sports. Similar patterns emerged in other domains. While this method has potential benefits, it also has limitations that may hinder the model's language modeling capabilities. Despite these challenges, we view this approach as promising and intend to provide a more detailed analysis in future research.

A.3 Tokenizer details

For tokenizer training, we utilized both open-source datasets, namely FRW (Penedo et al., 2023a), RedPajama (Together Computer, 2023), StarCoder (Li et al., 2023), as well as collected from the Web like Common Crawl [24], Wikipedia [25] and Stack Exchange [26]. For details on post-processing and cleaning the open-source datasets, refer to their respective articles. We filtered the datasets using established heuristics, such as language-based filtering and removing personal information, promotional content, and duplicates. Several sets of data were prepared for training tokenizers, varying in size from 30 billion to 300 billion characters to reflect different text lengths.

To ensure the effectiveness of our approach, we tested tokenizers against established models, including GPT-4, GPT-4o, Mistral, Qwen2, and DeepSeek. The comparison was based on the average character-per-token ratio across different domains, as summarized in Table 8 with selected domains. Tokenizers with the prefix giga_tokenizer represent multiple variants from our experiments, differing in data balancing strategies and the number of additional tokens introduced.

[23] Examples of the code for generation control are presented in the example notebook.

[24] https://commoncrawl.org/get-started

[25] https://dumps.wikimedia.org/ruwiki/latest/

[26] https://archive.org/details/stackexchange

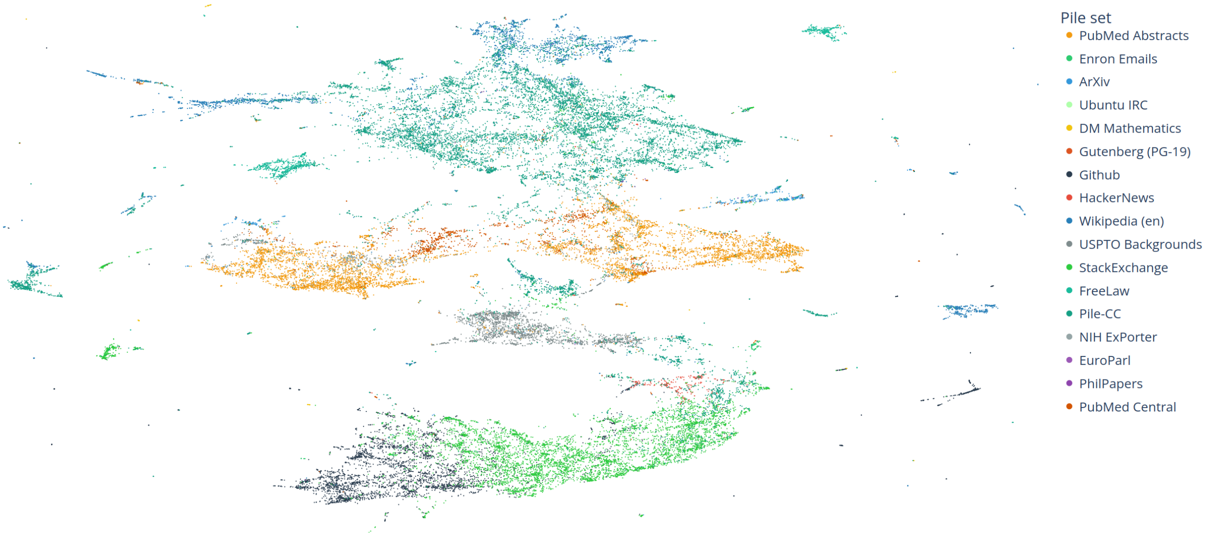


Figure 2: 2d-projection of embeddings with UMAP

Tokenizer	Languages			ArXiv	Wiki			Mean Score
	C	Java	C#		Ru	Ar	En	
giga_tokenizer_1	3.57	4.15	4.62	**3.61**	4.18	3.34	4.47	**3.99**
giga_tokenizer_2	3.56	4.14	4.60	**3.61**	4.14	3.30	4.44	3.97
gpt-4o	3.74	4.43	4.88	3.39	3.40	3.07	**4.68**	3.94
giga_tokenizer_5	3.39	3.97	4.44	3.54	**4.20**	**3.50**	4.43	3.92
giga_tokenizer_3	3.51	4.11	4.59	3.54	4.04	3.25	4.35	3.91
giga_tokenizer_4	3.50	4.11	4.58	3.53	4.00	3.21	4.33	3.90
llama-3	**3.75**	4.54	**4.99**	3.38	3.02	2.60	4.62	3.85
mistral-nemo	3.38	4.06	4.50	3.49	3.18	3.24	4.51	3.76
qwen2	3.69	4.52	4.95	3.31	2.70	2.56	4.50	3.75
gpt-4	3.74	**4.55**	4.98	3.38	2.04	1.44	4.62	3.54
nemotron-4-256k	2.82	3.34	3.76	3.25	3.20	2.93	4.57	3.41
deepseek-coder-v2	2.95	3.51	3.92	3.35	2.39	1.11	4.42	3.10
deepseek-v2	2.95	3.51	3.92	3.35	2.39	1.11	4.42	3.10
mistral-large	2.75	3.26	3.64	3.14	2.46	1.13	4.04	2.92

Table 8: Comparison of Tokenizers by Character-per-Token Ratio.