DOI: 10.15514/ISPRAS-2026-38(3)-51

RFCB: Russian Function Calling Benchmark

T.R. Ionov, ORCID: 0009-0002-2708-6353 <t.ionov@mts.ai>
V.A. Malykh, ORCID: 0000-0002-4508-2527 <v.malykh@mts.ai>

MWS AI, ITMO University,
49, Kronverkskiy Ave., Saint Petersburg, 197101, Russia.

Abstract. We present RFCB (Russian Function-Calling Benchmark) – a schema-preserving Russian localization of selected Single-turn and Multi-turn subsets from the Berkeley Function-Calling Leaderboard (BFCL). RFCB retains the structure, evaluation semantics, and JSON schemas of the original BFCL while providing fully translated user prompts, documentation, and string-valued targets in Russian. This benchmark enables apples-to-apples cross-lingual comparison of tool-using LLMs and serves as a foundation for evaluating function-calling capabilities in Russian. We evaluate proprietary and open-source models of various sizes. On top of the localized benchmark and its evaluation, we use a training pipeline that collects executable trajectories and supports three optimization regimes: supervised fine-tuning (SFT), direct preference optimization (DPO), and group relative policy optimization (GRPO). The pipeline is implemented with a modified Feedback-Driven Tool-Use Improvements (FTRL) based framework that performs multi-path exploration. We report cross-lingual comparisons on BFCL single-turn metrics, multi-turn state-based success, and robustness to long context and missing information, together with efficiency indicators. Our results show that single-turn accuracy remains close to baseline levels, with Russian consistently lagging behind English, whereas multi-turn evaluation exposes clear benefits of scaling and reinforcement-based optimization. RL-based methods (DPO, GRPO) markedly improve multi-turn behaviors across both languages. In particular, GRPO training yields the highest overall scores, moreover, with Russian results exceeding English by +6.5 percentage points, effectively reversing the usual cross-lingual gap.

Keywords: function calling; large language models; tool use; benchmark; evaluation; translation.

For citation: Ionov T.R., Malykh V.A. RFCB: Russian Function Calling Benchmark. Trudy ISP RAN/Proc. ISP RAS, 2026, vol. 38, issue 3, part 4, pp. 131-144. DOI: 10.15514/ISPRAS-2026-38(3)-51.

RFCB: Оценка вызова функций для русского языка

Т.Р. Ионов, ORCID: 0009-0002-2708-6353 <t.ionov@mts.ai>
В.А. Малых, ORCID: 0000-0002-4508-2527 <v.malykh@mts.ai>

Центр искусственного интеллекта МВС университета ИТМО,
Россия, 197101, Санкт-Петербург, Кронверкский проспект, д.49, литер А.

Аннотация. В статье представлен RFCB (Russian Function-Calling Benchmark) — русскоязычная адаптация одно- и многоходовых подмножеств тестового набора BFCL (Berkeley Function-Calling Leaderboard), предназначенного для оценки способности больших языковых моделей вызывать внешние функции. Пользовательские запросы, описания функций и строковые значения переведены на русский язык; идентификаторы функций, JSON-схемы и типизированные поля сохранены без изменений. Проведена оценка закрытых и открытых моделей разного размера. Предложен обучающий конвейер, собирающий траектории выполнения задач и поддерживающий три режима оптимизации: дообучение с учителем (SFT), прямую оптимизацию предпочтений (DPO) и групповую относительную оптимизацию политики (GRPO). Конвейер построен на модифицированной платформе FTRL с параллельным исследованием нескольких траекторий решения задачи. Представлены результаты межъязыкового сопоставления по одноходовым показателям, по успешности многоходовых задач (оцениваемой по итоговому состоянию окружения), по устойчивости к длинному контексту и отсутствующим данным, а также по вычислительной эффективности. Одноходовая точность остаётся близкой к исходной при устойчивом отставании русскоязычной версии от английской, тогда как многоходовая оценка выявляет заметные преимущества увеличения размера модели и применения методов обучения с подкреплением (DPO, GRPO). Наилучший результат даёт обучение методом GRPO, причём на русскоязычной версии показатели превосходят английские на 6,5 процентных пункта, обращая вспять типичное межъязыковое отставание.

Ключевые слова: вызов функций; большие языковые модели; использование инструментов; оценка качества; перевод.

Для цитирования: Ионов Т.Р., Малых В.А. RFCB: Оценка вызова функций для русского языка. Труды ИСП РАН, 2026, том 38, вып. 3, часть 4, стр. 131–144 (на английском языке). DOI: 10.15514/ISPRAS-2026-38(3)-51.

1. Introduction

Function calling – the ability of a large language model (LLM) to select appropriate tools, compose well-typed arguments, and recover across turns – has become a central mechanism for deploying LLMs in real applications, from enterprise automation to agentic assistants. The Berkeley Function-Calling Leaderboard (BFCL) systematizes this capability with executable tests and realistic API backends. In addition to AST-level checks for argument structure, the benchmark emphasizes runtime validity of calls and, starting with its multi-turn iteration (BFCL-V3), introduces state-based and subset-matched metrics that credit legitimate alternative trajectories and measure progress across steps within and across turns. These design choices move evaluation beyond single-shot templating toward realistic planning, exploration, and error recovery.

While single-turn leaderboards remain informative, they cannot by themselves reveal failure modes that arise when agents must recover from missing information, operate under long contexts, or decide not to call a tool when none is applicable. BFCL-V3 addresses this by explicitly modeling force terminations and per-turn state alignment, thereby penalizing loops and crediting valid (possibly non-canonical) plans. Our work extends this line to Russian by creating RFCB, a schema-preserving localization that translates only natural-language surfaces (user prompts, documentation, and string-valued targets) while keeping function identifiers, types, and schemas intact. This enables apples-to-apples cross-lingual comparison under identical scoring.

In summary, this paper contributes: (i) open-source RFCB, a license-compliant Russian localization of BFCL single- and multi-turn subsets that preserves schemas and evaluation semantics; (ii) a

reproducible evaluation harness compatible with OpenAI-style tool-calling interfaces; (iii) an empirical study covering single-turn and multi-turn metrics, as well as robustness to long-context overload, missing parameters/functions; (iv) analysis of different training regimes and (v) analysis of token efficiency and common error modes specific to Russian

2. Related Work

BFCL [1] established executable evaluation for function calling with (Abstract Syntax Tree) AST and runtime checks in single-turn, and later introduced multi-turn/multi-step with state-based metrics (BFCL-V3). Strengths include runnable backends, irrelevance detection, and explicit handling of force terminations; limitations are that most public artifacts are English-centric and backends are simulated rather than live, by design.

The Nexus Function Calling Leaderboard (NFCL [2]) is a widely used single-turn eval covering single, parallel, and nested calls. It does not include irrelevance detection and does not model multi-turn recovery, but offers diverse API categories and simple enterprise integration. This makes NFCL useful for tool-selection diagnostics, while BFCL-V3 remains better suited for multi-turn reliability analysis.

API-Bank [3] provides a runnable suite of 73 APIs with 314 dialogues and 753 API calls for assessing planning, retrieval, and calling; it further supplies a large training set (1,888 tool-use dialogues) spanning 2,138 APIs. Strengths are breadth and runnable tools; limitations include curation costs and the gap between live API drift and stable evaluation.

ToolLLM [4] scales tool-use data and training to more than 16000 real-world APIs, with an evaluation pipeline. However, reliance on live endpoints and LLM-based judges introduces instability; StableToolLLM [5] addresses this with a virtual API server, caching, simulators, and more stable pass/win rates. These resources are excellent for training and breadth, whereas BFCL-V3 emphasizes multi-turn evaluation semantics and executability with controlled backends.

WorkBench [6] targets workplace tasks (email, CRM, calendar and etc.) in a sandbox with 26 tools and 690 tasks, using outcome-centric evaluation (database state changes). It probes planning and action sequences across realistic business operations. While highly relevant for agents, it provides only multi-step, yet single-turn interaction, thus it is complementary benchmark.

The τ-bench [7] family evaluates agents in user-in-the-loop settings with domain-specific tools and policies. Strength: it exposes failures in dialogue management and policy adherence. Limitation: user simulators are themselves LLMs, so measurements can entangle agent errors with simulator artifacts.

In summary, BFCL in single- and multi-turn setup cover most of the tool calling cases without reliance on API or LLM simulated users. NFCL complements it for single-turn diagnostics; API-Bank/ToolBench ecosystems supply breadth and training corpora; tau-bench surface user-interaction issues; and WorkBench probes end-to-end office tasks. Our contribution is to port BFCL's evaluative semantics to Russian with a translation and validation pipeline aligned with best practices in multilingual dataset adaptation.

3. BFCL: Task Settings, Taxonomy, and Validation

3.1 Task settings and taxonomy

In our work we focus on self-contained (without external APIs) subsets of BFCL which support Python language.

Single-turn categories:

- Simple Function (400 instances): The instruction defines a single function, and the answer is exactly one valid function call with correct arguments is expected;

- Multiple Functions (200 instances): The instruction provides 2-4 function definitions, and the model must select and call the single relevant one from the given set;
- Parallel Functions (200 instances): The instruction defines a single function to be invoked multiple times in parallel with different argument values;
- Parallel-Multiple Functions (200 instances): A combination of the two scenarios above: several functions are provided, and each may be invoked zero or multiple times depending on relevance;
- Irrelevance Detection (200 instances): The instruction lists one or more functions, none of which are relevant to the user request. The correct behavior is to produce no function calls.

Multi-turn categories:

- Base (200 instances): Dialogue-style tasks where the model must plan and execute a sequence of function calls to achieve a target state, using prior context across turns.
- Missing Parameters (200 instances): Tasks where one or more required parameters are omitted in the initial user request. The model must identify the missing slots, ask clarifying questions, and correctly complete the call after receiving the needed information.
- Missing Functions (200 instances): Scenarios in which the required tool is not initially available; the model should signal that the function is missing and resume execution once it becomes accessible.
- Long-Context (200 instances): Large or noisy results from tools that require filtering, retrieval, and consistent memory across turns to preserve task coherence.

Overall, Single-turn subset contains 1200 samples and multi-turn – 800.

3.2 Validation process

Selected BFCL subset employs Abstract Syntax Tree (AST) evaluation, which structurally compares the predicted call to the ground-truth schema. Each model output is parsed into an AST representation. The validator extracts the function name, checks its correspondence with the one in the reference (allowing underscore substitutions for dots in identifiers), and verifies that:

- all required parameters listed in the function documentation are present;
- no extraneous parameters appear (hallucinated keys);
- parameter types and values match the specification.

Type checking follows strict but language-aware rules:

Casting integer to float is allowed; list and tuple elements are order-sensitive; strings are compared case-insensitively after whitespace and punctuation normalization; dictionaries are validated by key and value accuracy regardless of key order. Optional parameters are treated as correct if the model either uses the documented default or omits the value explicitly marked as optional.

For Multiple, Parallel, and Parallel-Multiple tasks, the system evaluates a set of predicted calls against the corresponding set of valid answers. Matching is order-independent and all-or-nothing: each expected function must be correctly instantiated at least once, and extraneous calls invalidate the result.

4. Translation Methodology for RFCB

4.1. Goals and constraints

We pursue semantic and structural equivalence to the English BFCL while producing idiomatic Russian: (i) preserve evaluation semantics (same intended tool choice and argument values), (ii)

preserve structure (function identifiers, JSON schemas, types, enums), and (iii) achieve natural yet technically precise Russian phrasing so models trained on Russian interpret slots reliably.

4.2. What is localized vs. preserved

Localized (translated): user-facing text in prompts and dialogue turns; free-text descriptions in function documentation; parameter and response-field descriptions; string-valued targets in ground truth; natural-language payloads in initial configurations. Preserved (not translated): function/parameter names; data types and formats; enum tokens; numeric/date/time values; IDs; URLs; and the JSON structure itself.

4.3. Two-stage pipeline: Claude 3.7 Sonnet MT and professional post-editing

Stage 1 – Automatic pass. We generate Russian drafts for all natural-language fields using Claude 3.7 Sonnet with a schema-aware prompt: translate only the specified fields; keep identifiers/types/enums and numeric/date formats intact; avoid code and structural edits. The MT output is thus constrained to natural-language surfaces relevant to tool selection and argument construction. Prompt provided in Listing 1.

```
ПОЛЯ ДЛЯ ПЕРЕВОДА:
question[*][*]["content"], где role="user" – пользовательские запросы
→ Переводи так, чтобы смысл точно соответствовал ожидаемым вызовам функций в ground_truth.
→ Сохраняй ключевые детали: числа, имена, даты, условия.
function[*]["description"] – описание функций
→ Передавай назначение и поведение точно, с технической терминологией.
function[*]["parameters"]["properties"][*]["description"] – описание параметров
→ Указывай тип данных, формат и ограничения.
ground_truth[*][function_name][parameter_name][*] – эталонные значения параметров
→ Переводи только строковые значения (имена, описания); не изменяй числа, даты, коды.
ОБЕСПЕЧЕНИЕ СОГЛАСОВАННОСТИ:
→ Если в вопросе и ответе встречается одно имя (напр. «Taylor Swift»), переводи одинаково («Тейлор Свифт»).
→ Сохраняй все числовые значения и даты в исходном виде.
→ Технические термины (API keys, URLs, endpoints) оставляй на английском.
НЕ ПЕРЕВОДИ:
→ Имена функций и параметров (calculate_area, artist_name);
→ Типы данных (string, integer, boolean);
→ Технические конструкции JSON и числовые идентификаторы.
```

Listing 1. Prompt for automatic translation with Claude 3.7 Sonnet.

Stage 2 – Human post-editing. A team of five professional linguists performs schema-preserving post-editing focused on semantic fidelity, slot integrity, and technical style. Editors cross-check each item against the English original and the function schemas; ambiguous items are adjudicated collectively. All edits are logged, and validators are re-run after each batch to ensure structural correctness.

To illustrate the preservation of structure and selective localization of natural-language fields, Listing 2 shows an example of a fully localized sample. Localized Multi-turn Base task provided in Listing 3.

4.4. Consistency, normalization, and bilingual acceptables

Named entities and transliteration. Proper names and product titles may legitimately appear in Russian or English. When both are acceptable to the backend, ground truth contains bilingual acceptables (e.g., ["Тейлор Свифт", "Taylor Swift"]) so that correct outputs are not penalized. For slot values, we prefer invariant dictionary forms (no case inflection) unless morphology is strictly required by the user text.

Strict enum compliance. If a parameter is constrained by an enum, the ground truth must use a form from that enum exactly; Technical keys and field names are likewise preserved. This ensures that JSON validation and executable checks behave identically across languages.

```
{
 "question": [[
  {
    "role": "user",
    "content": "Вычисли площадь треугольника, если его основание равно 10 единицам, а высота равна 5 единицам.",
  }
 ] ],
 "function": [ {
   "name": "calculate_triangle_area",
   "description": "Вычислить площадь треугольника по его основанию и высоте.",
   "parameters": {
     "type": "dict",
     "properties": {
       "base": {
         "type": "integer",
         "description": "Основание треугольника.",
       },
       "height": {
         "type": "integer",
         "description": "Высота треугольника.",
       },
       "unit": {
         "type": "string",
         "description": "Единица измерения (по умолчанию используется значение 'единицы', если не указано конкретное).",
       }
     },
     "required": ["base", "height"]
   }
 } ]}
```

Listing 2. Example from Single-turn Simple subset of RFCB.

Numbers, dates, and formats. Numeric strings, ISO-like dates, time stamps, and identifiers remain unchanged. Decimal/thousands separators and time zones follow the English source to avoid silent parse errors in tools that expect specific formats.

Style and register. Function documentation uses a concise technical register (e.g., "удалить элемент", "сбросить параметр", "инициировать запрос"), with disambiguation where English has polysemy ("remove/clear/reset").

4.5. Irrelevance discipline

In irrelevance items no ground truth call is provided intentionally; the correct behavior is not to call any function. Russian prompts are phrased neutrally not to provoke over-helpful behavior and to test abstention properly.

4.6. Quality-control workflow and metrics

Automatic validators: JSON schema checks; AST equality on single-turn; executable checks where available; enum membership validation; per-turn state comparison in multi-turn.

4.7. Typical issues and our fixes

- Slot boundary erosion from rich morphology: use uninflected forms in value slots; restructure sentences to keep values adjacent to keywords.
- Enum token over-translation: keep tokens verbatim; translate only explanatory text.
- Reference drift across turns: minimal re-mentioning of key entities; avoid pronoun chains that obscure targets.

5. Experimental Setup

Models are evaluated with original open-source BFCL repository with extended test files in Russian. Zero temperature set for all models to ensure stable results.

```
{ "tools": [
   {
     "name": "mkdir",
     "description": "Создание новой директории в текущей директории.",
     "parameters": …
   },
   {
    "name": "mv",
    "description": "Перемещение файла или директории из одного места в другое.",
    "parameters": …
   },
   <Other tools>
 ],
 "question": [[{
     "role": "user",
     "content": "Перемести файл 'final_report.pdf' из директории 'document' в директорию 'temp' внутри папки 'document'.
Убедись, что эта директория была создана.",          }
     ],
     [{
      "role": "user",
      "content": "Выполни детальный поиск с помощью grep, чтобы найти разделы в файле, относящиеся к анализу бюджета.",
     }],
     [{
      "role": "user",
      "content": "После обнаружения нужных данных, касающихся анализа бюджета, отсортируй файл 'final_report.pdf'
построчно для лучшего восприятия.",
     }],
     [{
      "role": "user",
      "content": "Перемести файл 'previous_report.pdf' из директории 'document' в 'temp'. Помести туда же финальный отчёт.
Сравни его с 'previous_report.pdf', чтобы выявить критические изменения.",
 }]],
 "initial_config": {
   "GorillaFileSystem": {
    "workspace": {
     "type": "directory",
     "contents": {
      "document": {
       "type": "directory",
       "contents": {
        "final_report.pdf": {
         "type": "file",
         "content": "2024 год – это содержимое финального отчёта, включающее анализ бюджета и другие разделы."
        },
        "previous_report.pdf": {
         "type": "file",
         "content": "2023 год – это содержимое предыдущего отчёта с другим анализом бюджета."
        }
       }
```

Listing 3. Abbreviated example from Multi-turn Base subset of RFCB.

In Multi-turn setting we use default parameters: maximum 3 retries for each turn and total 20 turns for task. vLLM [8] framework used for inference on 2-4 Nvidia A100 with 48 Gb VRAM.

For training we implement a unified optimization pipeline on top of an FTRL [9] framework and adapt it to multi-path exploration for function calling. At each decision step within a turn, the current policy samples K=10 candidate action paths with sampling temperature set to 0.7. A lightweight success evaluator checks each turn by feedback from Python environment and assigns binary reward: 0 – incorrect, 1 – successful completion. We train LoRA [10] adapters with rank 8, alpha 16 and dropout 0.05 with batch size 16 across all settings.

5.1. Trace collection and filtering

We store full trajectories, including partial and failed branches, and apply filters to ensure training safety and stability: JSON-schema gating for arguments; de-duplication of near-identical calls; This yields a dataset of positive paths (successful states) and hard negatives (failed or inefficient branches).

5.2 Supervised fine-tuning (SFT) on golden sub-trajectories

For SFT we extract golden sub-trajectories from selected successful paths and train the policy to reproduce tool selection and argument construction. We optimize the SFT objective with learning rate 0.0005 for 3 epochs.

5.3 Direct Preference Optimization (DPO) from paired paths

For DPO [11], we form paired preferences per decision point: a preferred sample from the successful path and a dispreferred sample from failed/less efficient candidates. If after deduplication we have several positive and negative candidates, we randomly sample from 1 up to 5 different DPO pairs for each turn. We optimize the DPO objective with β=0.2 and learning rate 0.0005 for 3 epochs. Preferences reflect state-based success, fewer steps-to-success, and abstention correctness on irrelevance items.

5.4 Group Relative Preference Optimization (GRPO) with K-way ranking

For GRPO [12], we rank the N=8 candidates by the same feedback environment and optimize group-relative preferences with learning rate 0.00007, β=0.04, temperature sampling 0.7 for 3 epochs. This encourages robust improvements under exploration noise and aligns with multi-tool planning.

5.5 Reproducibility

The full evaluation code, translation prompts, and replication scripts are available in an open repository [13] ensuring full reproducibility of our results.

6. Results

Abbreviations used for the Single-turn categories are as follows: Sim. – Simple; Mul. – Multiple; Par. – Parallel; Par. Mul. – Parallel Multiple; Irr. – Irrelevance detection.

Table 1 shows evaluation of proprietary models where gpt-4.1 and gpt-4o from OpenAI generally show high EN and RU performance compare to YandexGPT from Yandex and GigaChat from Sber. Results on RU subset are lagging behind EN across all models all setups, however, gap in Irrelevance detection task is relatively low. GigaChat shows zero on Parallel and Parallel Multiple because either model or API interface does not support generation multiple function calling at once. Multi-turn evaluation, while valuable for studying dialogue consistency and function-call persistence, introduces substantial variability due to state propagation between turns and higher inference cost (~ $200 for one run gpt-4.1).

Table 2 shows evaluation of open-source Qwen 3 and 2.5 series models of different sizes from 8 up to 32 billion parameters in Single-turn setting. All models show high results in English and moderate results in Russian. Qwen 3 uses native reasoning trace before answer. Although the Qwen2.5-32B model does not reach the top English score, it excels on the Russian subset and exhibits the lowest cross-lingual lag.

Table 3 shows evaluation of open-source Qwen 3 and 2.5 series models in Multi-turn setting. Qwen 3 models are evaluated in thinking regime. While Single-turn tasks remain simpler, the Multi-turn evaluation poses a higher level of difficulty and demonstrates the benefits of scaling: larger models

achieve consistently better results in both Russian and English. Qwen2.5, in particular, attains the best overall performance with a significant margin. Notably, Qwen3-14B slightly improves on Miss-Funcs (Δ = +3 pp), confirming that certain reasoning sub-skills transfer well between languages when dialogue context is preserved.

Table 1. Proprietary models Single-Turn results by category (EN vs RU). Values in percent; Δ is difference between RU and EN results.

Model	Lang.	Sim.	Mul.	Par.	Par. Mul.	Irr.	Overall
gpt-4.1	EN	95.50	94.50	92.50	89.00	90.00	92.30
	RU	76.13	86.50	74.50	59.00	85.00	76.23
	Δ	-19.37	-8.00	-18.00	-30.00	-5.00	-16.07
gpt-4o	EN	96.25	95.00	94.00	83.50	89.58	91.67
	RU	74.62	86.50	74.50	52.00	86.25	74.77
	Δ	-21.63	-8.50	-19.50	-31.50	-3.33	-16.90
YandexGPT Pro 5	EN	40.00	38.50	26.50	26.50	92.92	44.88
	RU	24.62	26.50	17.50	14.50	91.25	34.87
	Δ	-15.38	-12.00	-9.00	-12.00	-1.67	-10.01
GigaChat2 Max	EN	53.75	35.00	0.00	0.00	87.92	35.33
	RU	45.48	32.00	0.00	0.00	86.25	32.77
	Δ	-8.27	-3.00	0.00	0.00	-1.67	-2.56

Table 2. Open-source models Single-Turn results by category (EN vs RU). Values in percent; Δ is difference between RU and EN results.

Model	Lang.	Sim.	Mul.	Par.	Par. Mul.	Irr.	Overall
Qwen3-8B	EN	95.00	95.00	90.50	89.00	86.25	91.15
	RU	66.58	68.50	65.00	47.50	85.42	66.60
	Δ	-28.42	-26.50	-25.50	-41.50	-0.83	-24.55
Qwen3-14B	EN	94.50	95.00	90.50	88.50	87.92	91.28
	RU	69.60	82.50	70.50	56.00	85.00	72.72
	Δ	-24.90	-12.50	-20.00	-32.50	-2.92	-18.56
Qwen3-32B	EN	95.50	95.50	89.50	91.00	85.42	91.38
	RU	67.09	70.50	64.50	48.50	84.17	66.95
	Δ	-28.41	-25.00	-25.00	-42.50	-1.25	-24.43
Qwen2.5-32B	EN	96.50	95.00	89.50	88.50	79.17	89.73
	RU	73.62	81.50	77.00	56.50	73.75	72.47
	Δ	-22.88	-13.50	-12.50	-32.00	-5.42	-17.26

Table 3. Open-source models Multi-Turn results by category (EN vs RU). Values in percent; Δ is difference between RU and EN results.

Model	Lang.	Base	Miss-Params	Miss-Funcs	Long-Ctx	Overall
Qwen3-8B	EN	5.50	4.00	9.00	3.50	5.50
	RU	6.00	5.03	5.50	2.50	4.76
	Δ	0.50	1.03	-3.50	-1.00	-0.74
Qwen3-14B	EN	13.50	13.00	15.00	12.50	13.50
	RU	7.50	8.00	18.00	10.00	10.88
	Δ	-6.00	-5.00	3.00	-2.50	-2.62
Qwen3-32B	EN	26.00	24.00	29.50	26.00	26.38
	RU	19.50	22.61	24.50	21.00	21.90
	Δ	-6.50	-1.39	-5.00	-5.00	-4.47
Qwen2.5-32B	EN	42.00	39.50	37.00	41.00	39.88
	RU	39.50	28.64	29.00	40.00	34.29
	Δ	-2.50	-10.86	-8.00	-1.00	-5.59

6.2 Effect of different training regimes

Table 4 presents the evaluation of models trained under different regimes - SFT, DPO, and GRPO - on the Single-turn subset. The SFT method shows a slight drop in overall accuracy, whereas DPO and GRPO preserve comparable quality across most categories and, in several cases, achieve noticeable improvements.

Table 5 shows evaluation of models trained with different regimes on the Multi-turn subset. RL-based methods (DPO, GRPO) consistently improve turn performance across languages. While SFT yields only minor gains and reduces RU accuracy, GRPO achieves the highest scores, with Russian results exceeding English by +6.5 pp overall, demonstrating effective cross-lingual generalization of reasoning and recovery strategies.

Table 4. Effect of different training regimes on Qwen3-8B Single-Turn results by category (EN vs RU). Values in percent; Δ is difference between RU and EN results.

Training	Lang.	Sim.	Mul.	Par.	Par. Mul.	Irr.	Overall
No	EN	95.00	95.00	90.50	89.00	86.25	91.15
	RU	66.58	68.50	65.00	47.50	85.42	66.60
	Δ	-28.42	-26.50	-25.50	-41.50	-0.83	-24.55
SFT	EN	56.50	59.50	66.00	53.00	87.92	64.58
	RU	36.68	48.50	48.50	30.00	84.17	49.57
	Δ	-19.82	-11.00	-17.50	-23.00	-3.75	-15.01
DPO	EN	96.75	95.00	90.00	88.00	87.92	91.53
	RU	57.04	67.50	62.00	45.00	88.33	63.97
	Δ	-39.71	-27.50	-28.00	-43.00	0.42	-27.56
GRPO	EN	93.75	95.50	90.00	89.00	88.33	91.32
	RU	63.82	75.00	66.50	53.00	82.92	68.25
	Δ	-29.93	-20.50	-23.50	-36.00	-5.42	-23.07

Table 5. Effect of different training regimes on Qwen3-8B Multi-Turn results by category (EN vs RU). Values in percent; Δ is difference between RU and EN results.

Training	Lang.	Base	Miss-Params	Miss-Funcs	Long-Ctx	Overall
No	EN	5.50	4.00	9.00	3.50	5.50
	RU	6.00	5.03	5.50	2.50	4.76
	Δ	0.50	1.03	-3.50	-1.00	-0.74
SFT	EN	9.50	4.50	4.00	9.00	6.75
	RU	3.00	2.51	2.50	3.50	2.88
	Δ	-6.50	-1.99	-1.50	-5.50	-3.87
DPO	EN	25.00	25.00	22.00	24.00	24.00
	RU	22.50	24.12	19.00	21.50	21.78
	Δ	-2.50	-0.88	-3.00	-2.50	-2.22
GRPO	EN	27.00	26.00	30.50	32.00	28.88
	RU	31.00	27.14	45.50	38.00	35.41
	Δ	4.00	1.14	15.00	6.00	6.53

6.3 Token usage and error analysis

Table 6 shows token usage of different models. Russian prompts systematically require more input tokens than English in single-turn (+150-240 tokens/model), while outputs grow only slightly (+3-22). In multi-turn, total token budgets rise sharply, but the RU-EN gap narrows (e.g., Qwen3-8B: +71 input / +2 output), indicating comparable dialog efficiency across languages. The "thinking" variant increases single-turn outputs (longer reasoning traces) but adds relatively little to multi-turn outputs, where the overhead is dominated by input/history.

Fig. 1 depicts error distribution for gpt-4.1 model on Single-turn subset. The Russian subset is dominated by parallel_function_checker_no_order:cannot_find_match and value_error:string: the model selects a parallel tool set but fails to match one of the required calls, and – when it does call string normalization mismatches remain frequent. irrelevance_error:decoder_success is higher in RU than EN, indicating occasional over-activation (a function is called when abstention was expected). Count errors (…:wrong_count) and AST decoder failures are comparatively minor in both languages.

Table 6. Token usage (average tokens per task) by language (EN vs RU). Values are average token counts; Δ is RU−EN difference (absolute tokens).

Model	Lang.	Single-Turn		Multi-Turn	
		Input	Output	Input	Output
gpt-4.1	EN	501.02	34.50	-	-
	RU	650.05	37.69	-	-
	Δ	149.03	3.19	-	-
Qwen3-8B (thinking)	EN	508.10	303.93	11090.56	451.76
	RU	686.12	325.47	11161.19	453.71
	Δ	178.02	21.54	70.63	1.95
Qwen2.5-32B	EN	508.10	40.23	6044.13	39.45
	RU	746.92	45.68	10434.65	43.69
	Δ	238.82	5.45	4390.52	4.24

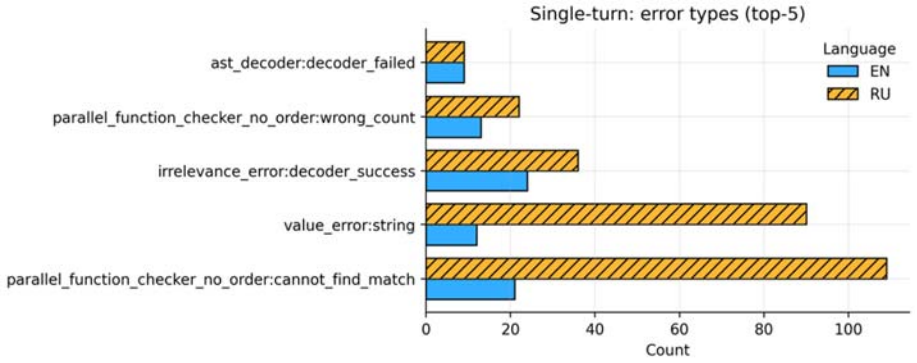

Fig. 1. Top-5 single-turn error distribution for gpt-4.1.

Fig. 2 depicts error distribution for Qwen3-8B model trained with GRPO regime. Most errors come from empty_turn_model_response: the model answers without a tool call when a call is required, i.e., fails to act at a necessary turn. The next most frequent class is force_terminated, when the model does not reach the goal within the allowed number of turns (generated trajectory length differs from

reference). By contrast, execution_response_mismatch is comparatively rare: calls are executed but cumulative responses miss some ground-truth outputs for that turn.

8. Discussion

In this paper, we presented RFCB, a Russian localization of the Berkeley Function-Calling Leaderboard (BFCL) designed for evaluating function-calling capabilities in both single-turn and multi-turn settings. Our experiments revealed several key insights into how models perform with Russian prompts compared to their English counterparts.

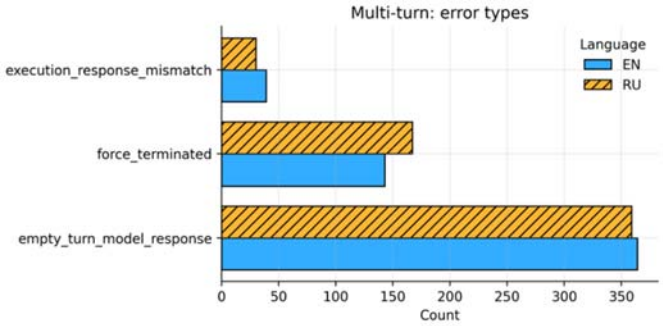

Fig. 2. Multi-turn errors distribution for GRPO trained Qwen3-8B.

Single-turn results in Russian consistently lag behind English models across all categories, even for proprietary systems trained on domestic data. The largest discrepancies are observed in Parallel and Parallel-Multiple tasks, where models struggle with multi-tool composition and argument alignment, largely due to morphological complexity and syntactic variation in Russian.

Multi-turn evaluation, in contrast, highlights significant benefits of scaling and reinforcement-based optimization (DPO/GRPO). Multi-turn models show substantial improvements in multi-step planning and error recovery, with GRPO producing the best overall results and effectively transferring tool-use abilities across languages.

The benchmark thus establishes a controlled yet realistic environment for measuring cross-lingual tool-use generalization

9. Conclusion

We introduced RFCB, the first schema-preserving Russian localization of the Berkeley Function-Calling Leaderboard. The two-stage translation pipeline- LLM-based automatic translation followed by expert post-editing - preserves evaluation semantics while ensuring linguistic and technical fidelity. The released dataset, evaluation scripts, and training pipeline make the benchmark fully reproducible [13].

Our evaluation reveals cross-lingual comparisons in single-turn tasks and highlights significant improvements in multi-turn scenarios when models are trained with reinforcement-based optimization (DPO/GRPO). These findings confirm that reinforcement learning not only enhances multi-step reasoning and tool planning but also helps close the cross-lingual gap between Russian and English models.

RFCB provides a solid foundation for future research on multilingual tool-use LLMs, offering a standardized framework for model evaluation, fine-tuning, and comparison.

Future work will focus on expanding the benchmark to other programming languages and live API environments, as well as integrating human-in-the-loop evaluation for long-context and robustness testing.

References

[1]. Patil S.G., Mao H., Yan F., Ji C.C.-J., Suresh V., Stoica I., Gonzalez J.E. The Berkeley Function Calling Leaderboard (BFCL): From Tool Use to Agentic Evaluation of Large Language Models. Proceedings of the 42nd International Conference on Machine Learning. Proceedings of Machine Learning Research, 2025, vol. 267. pp. 48371-48392. Available at: https://proceedings.mlr.press/v267/patil25a.html, accessed: 22.12.2025.

[2]. Nexusflow. Nexus Function Calling Leaderboard. Hugging Face Spaces, 2023. Available at: https://huggingface.co/spaces/Nexusflow/Nexus_Function_Calling_Leaderboard, accessed: 22.12.2025.

[3]. Li M., Zhao Y., Yu B., Song F., Li H., Yu H., Li Z., Huang F., Li Y. API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023. pp. 3102-3116. Available at: https://aclanthology.org/2023.emnlp-main.187/, accessed: 22.12.2025. DOI: 10.18653/v1/2023.emnlp-main.187.

[4]. Qin Y., Liang S., Ye Y., Zhu K., Yan L., Lu Y., Lin Y., Cong X., Tang X., Qian B., Zhao S., Hong L., Tian R., Xie R., Zhou J., Gerstein M., Li D., Liu Z., Sun M. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. International Conference on Learning Representations (ICLR 2024), 2024. Available at: https://openreview.net/forum?id=dHng2O0Jjr, accessed: 22.12.2025.

[5]. Guo Z., Cheng S., Wang H., Liang S., Qin Y., Li P., Liu Z., Sun M., Liu Y. StableToolBench: Towards Stable Large-Scale Benchmarking on Tool Learning of Large Language Models. Findings of the Association for Computational Linguistics: ACL 2024, 2024. pp. 11143-11156. Available at: https://aclanthology.org/2024.findings-acl.664/, accessed: 22.12.2025. DOI: 10.18653/v1/2024.findings-acl.664.

[6]. Styles O., Miller S., Cerda-Mardini P., Guha T., Sanchez V., Vidgen B. WorkBench: a Benchmark Dataset for Agents in a Realistic Workplace Setting. arXiv preprint arXiv:2405.00823, 2024. Available at: https://arxiv.org/abs/2405.00823, accessed: 22.12.2025. DOI: 10.48550/arXiv.2405.00823.

[7]. Yao S., Shinn N., Razavi P., Narasimhan K. tau-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. arXiv preprint arXiv:2406.12045, 2024. Available at: https://arxiv.org/abs/2406.12045, accessed: 22.12.2025. DOI: 10.48550/arXiv.2406.12045.

[8]. Kwon W., Li Z., Zhuang S., Sheng Y., Zheng L., Yu C.H., Gonzalez J.E., Zhang H., Stoica I. Efficient Memory Management for Large Language Model Serving with PagedAttention. Proceedings of the 29th Symposium on Operating Systems Principles (SOSP 2023), 2023, pp. 611-626. Available at: https://doi.org/10.1145/3600006.3613165, accessed: 22.12.2025. DOI: 10.1145/3600006.3613165.

[9]. Ye J., Jiang C., Du Z., Xu Y., Yao X., Xi Z., Fan X., Zhang Q., Gui T., Huang X., Chen J. Feedback-Driven Tool-Use Improvements in Large Language Models via Automated Build Environments. arXiv preprint arXiv:2508.08791, 2025. Available at: https://arxiv.org/abs/2508.08791, accessed: 22.12.2025. DOI: 10.48550/arXiv.2508.08791.

[10]. Hu E.J., Shen Y., Wallis P., Allen-Zhu Z., Li Y., Wang S., Wang L., Chen W. LoRA: Low-Rank Adaptation of Large Language Models. International Conference on Learning Representations (ICLR 2022), 2022. Available at: https://openreview.net/forum?id=nZeVKeeFYf9, accessed: 22.12.2025.

[11]. Rafailov R., Sharma A., Mitchell E., Manning C.D., Ermon S., Finn C. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. Advances in Neural Information Processing Systems 36 (NeurIPS 2023), 2023, pp. 53728-53741. DOI: 10.5555/3666122.3668460.

[12]. Shao Z., Wang P., Zhu Q., Xu R., Song J., Bi X., Zhang H., Zhang M., Li Y.K., Wu Y., Guo D. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv preprint arXiv:2402.03300, 2024. Available at: https://arxiv.org/abs/2402.03300, accessed: 22.12.2025. DOI: 10.48550/arXiv.2402.03300.

[13]. Ionov T.R., Malykh V.A. RFCB: Russian Function-Calling Benchmark. GitHub repository, 2024. Available at: https://github.com/sir-timio/RFCB, accessed: 22.12.2025.

Информация об авторах / Information about authors

Тимур Русланович ИОНОВ – аспирант института прикладных компьютерных наук Университета ИТМО и разработчик-исследователь в MWS AI.

Timur Ruslanovich IONOV is a postgraduate student at the Institute of Applied Computer Science at ITMO University and a research engineer at MWS AI.

Валентин Андреевич МАЛЫХ, кандидат технических наук, и.о. руководителя направления фундаментальных исследований в MWS AI. Сфера научных интересов: обработка естественного языка, большие языковые модели.

Valentin Andreevich MALYKH – Cand. Sci. (Tech.), Acting Head of Fundamental Research Department at MWS AI. Research interests: natural language processing, large language models.