

RUADAPT: ВЫЧИСЛИТЕЛЬНО ЭФФЕКТИВНАЯ ЯЗЫКОВАЯ АДАПТАЦИЯ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ

© 2025 г. М. М. Тихомиров^{1,*}, Д. И. Чернышев^{1,**}

Представлено

Поступило 01.04.2022

После доработки 31.04.2022

Принято к публикации 27.05.2022

Мультиязычные большие языковые модели (LLM) часто показывают сниженную производительность для языков, отличных от английского, из-за несбалансированности обучающих данных. Прямая адаптация таких моделей к новому языку, например, русскому, сопряжена с риском катастрофического забывания исходных знаний и требует значительных вычислительных ресурсов. В этой работе мы представляем Ruadapt — комплексную и вычислительно эффективную методологию для языковой адаптации LLM с заменой токенизации. Полная адаптация одного варианта Qwen3-8B модели по нашей методологии требует менее 2000 GPU-часов, а последующая адаптация других вариантов в 10 раз меньше, благодаря разделимости результатов каждого шага процедуры. Оптимальная конфигурация процедуры адаптации позволяет добиться до 80% ускорения генерации с полным сохранением навыков работы с длинным контекстом и незначительными потерями эффективности анализа пользовательских инструкций. Мы приводим подробное эмпирическое исследование каждого шага адаптации с целью определения оптимальных гиперпараметров, а также ключевых этапов и их влияния на итоговое качество. Выработанные рекомендации применяются в текущем поколении моделей Ruadapt, включая RuadaptQwen3-32B-Hybrid. Мы публикуем наши модели, код и наборы данных в открытом доступе, предлагая научному сообществу проверенную и экономически целесообразную стратегию создания высококачественных языковых моделей.

Ключевые слова и фразы: Большие языковые модели, языковая адаптация, русский язык

DOI: 10.31857/S2686954322040117

ВВЕДЕНИЕ

Современные большие языковые модели (LLM) обладают обширными знаниями, однако их применение для языков, отличных от английского, ограничено в силу обучающих данных в пользу английского языка. По этим причинам модели могут упускать культурные и стилистические особенности других языков, а также менее эффективно представляют текст на иных языках при обработке.

Проблемы локализации делают актуальной задачу обучения специальных вариантов больших языковых моделей, учитывающих необходимые особенности целевого языка. Для русского языка наиболее эффективными показали себя модели [5, 6], полученные посредством "обучения с нуля" на качественных корпусах русского языка. Однако было также продемонстрировано, что подобные результаты могут быть достигнуты адаптацией мультиязычных больших языковых моделей посредством дообучения на русскоязычных инструкциях [2] и заменой словаря генерации на более морфологически корректный [3].

Адаптация существующей мультиязычной большой языковой модели несет меньше экономических рисков чем "обучение с нуля" собственной, поскольку совокупная стоимость процедуры на порядки ниже. В то же время реализация процедуры адаптации усложняется тем, что существующие модели имеют высокую плотность знаний в силу чего любая процедура дообучения приводит к непредсказуемой потере полезных навыков [13, 14, 12]. Самым простым вариантом адаптации считается дообучение большой языковой модели на примерах выполнения пользовательских инструкций [2] на целевом языке. Более продвинутый вариант - предварительное дообучения большой языковой модели на общих

¹Московский государственный университет им. М. В. Ломоносова; научно-исследовательский вычислительный центр, Москва, Россия

*E-mail: tikhomirov.mm@gmail.com

**E-mail: chdanorbis@yandex.ru

корпусах целевого языка на задаче языкового моделирования (как правило предсказания следующего токена), а затем уже обучение и калибровка предпочтений под инструкции пользователей целевого языка. Наиболее совершенный подход [1] дополняет эту схему шагом адаптации словаря токенизации к эффективной обработке и анализу слов и словосочетаний, что позволяет также повысить вычислительную эффективность обработки документов.

В данной работе исследуются проблемы организации процедуры адаптации на примере вычислительно эффективного подхода Ruadapt. Основной вклад работы заключается в следующем:

- Мы представляем **практическое руководство по адаптации мультязычных LLM** к русскому языку, позволяющее достигать лучших результатов с экономией вычислительных ресурсов в 10 раз.
- Мы проводим эмпирическое исследование оптимальных гиперпараметров, реализаций шагов и синтеза набора данных процедуры языковой адаптации, подкрепляя его также анализом вычислительных затрат.
- Мы приводим анализ сохраняемых навыков после процедуры адаптации и определяем оптимальные для больших языковых моделей конфигурации, позволяющие максимизировать понимание пользовательских запросов и качество работы с длинным контекстом.
- Весь актуальный код методологии открыт, а наилучшие из обученных моделей выложены в открытый доступ.

МЕТОДОЛОГИЯ

Серия русскоязычных больших языковых моделей Ruadapt основана на обобщенной процедуре адаптации генеративных языковых моделей, которая может быть описана следующими шагами:

1. Построение нового словаря токенизации
2. Обучение эмбеддингов для новых элементов словаря токенизации
3. Выравнивание базовой языковой модели в соответствии с новым словарем токенизации
4. Перенос новых языковых знаний на целевую версию исходной модели посредством метода Learned Embedding Propagation (LEP)
5. Калибровка адаптированной целевой версии исходной модели на примерах решения задач на целевом языке

Построение нового словаря токенизации Обновление словаря токенизации - важный и дорогой шаг. Важность заключается в возможности повышения общей эффективности языковой модели как с точки зрения аналитических способностей, так и общей скорости генерации текста. Дорогой же он по причине необходимости последующей адаптации модели к новым токенам словаря.

Наиболее эффективный метод [1] построения словаря - расширение, с возможным удалением лишних элементов (например, для исключения возможности генерации на ненужных языках). Новые элементы словаря могут быть получены алгоритмами BPE или Unigram. BPE используется в большинстве доступных больших языковых моделей, поэтому в контексте расширения обычно подразумевают его. Однако Unigram позволяет гораздо точнее воспроизводить морфологию русского языка [3].

Для совмещения исходного BPE словаря с Unigram расширением в Ruadapt используется следующая процедура:

1. В исходный список токенов добавляются все новые.
2. Вероятность каждого токена t Unigram переводится в частоту по формуле $freq(t) = 1e6 \cdot P(t)$.
3. В порядке убывания частоты дерево слияния (merges) дополняется новыми ветками построения добавленных токенов по алгоритму BPE.

Обучение эмбеддингов для новых элементов словаря токенизации В контексте генеративных языковых моделей под эмбеддингами подразумевается входной слой нейронной сети, переводящий токен в векторное представление, и выходной (LM head), выполняющий обратное преобразование скрытого представления в токен. Для обучения эмбеддингов новых токенов используют подход локального дообучения модели на корпусах целевого языка [3]. Сначала эмбеддинги нового токена инициализируются посредством усреднения эмбеддингов токенов, на которые разбивается новый токен согласно старому словарю токенизации. Затем эти эмбеддинги обучаются на языковом моделировании (предсказании следующего токена) русского языка. При этом другие параметры модели остаются без изменений.

Выравнивание базовой языковой модели Для адаптации внутренних слоев модели к новой токенизации можно провести процедуру выравнивания, посредством одновременного дообучения эмбеддингов и слоев модели на задаче моделирования целевого языка. Дообучение может проводиться

RUADAPT: ВЫЧИСЛИТЕЛЬНО ЭФФЕКТИВНАЯ ЯЗЫКОВАЯ АДАПТАЦИЯ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ 3 разными способами, однако в наших [4] экспериментах наиболее вычислительно эффективным оказался вариант дообучения посредством LoRA адаптеров на дополнительном корпусе русскоязычных текстов.

Перенос новых языковых знаний посредством метода LEP После выравнивания большой языковой модели возможны 2 сценария: непосредственная калибровка модели на инструктивных данных или предварительный перенос приобретенных языковых знаний на другую версию с последующей калибровкой. Второй сценарий полезен когда новые эмбединги обучались для некоторой базовой модели, на базе которой ранее уже были обучены сложновоспроизводимые инструктивные версии, поскольку перенос позволит существенно сократить расходы на воспроизведение способностей.

Для переноса эмбедингов применяется метод Learned Embedding Propagation (LEP) [1], который аппроксимирует процедуру адаптации модели к новому словарю посредством линейных операторов. Если выравнивание модели проводилось посредством адаптеров, таких как LoRA, то результат этого шага также может быть перенесен на целевую модель.

Калибровка адаптированной версии модели Калибровка модели на решение задач на целевом языке является главным этапом процедуры адаптации. С одной стороны, метод LEP позволяет применять итоговую модель [1] по назначению сразу после переноса, но с другой, дополнительная калибровка на качественных примерах позволяет сгладить ошибки аппроксимации и тем самым сократить потери навыков и знаний.

Процедура калибровки модели представляет собой классическую процедуру дообучения (supervised fine-tuning, SFT) на размеченных примерах. Дообучаться могут все параметры модели, отдельные слои или специальные адаптеры. Последний вариант является наименее вычислительно требовательным и одновременно наиболее надежным, поскольку адаптеры, такие как LoRA, имеют ограниченное влияние на параметры в силу работы с низкоранговыми приближениями исходных параметрических матриц и как следствие изменяют только высокоуровневые закономерности [15].

ДАННЫЕ

Данные обучения и выравнивания В качестве набора данных для продолженного предобучения мы использовали комбинацию из rulm [7] и Fineweb-2 [8], общим объемом ≈ 150 ГБ текстовых данных или ≈ 18 миллиардов токенов. При этом данные были разделены на 2 части: обучения эмбедингов (80 ГБ) и выравнивания модели (70 ГБ).

Данные калибровки Потенциальным источником данных процедуры калибровки может быть любая коллекция примеров целевой задачи, однако в случае наличия в этой коллекции примеров, противоречащих знаниям или стилю генерации адаптируемой модели, эффект от процедуры может быть незначительный или даже отрицательный. Поэтому наиболее надежным и эффективным вариантом является специальный синтез примеров калибровки с учетом присутствующих в модели закономерностей.

Для этапа калибровки рассматривались 3 разных набора:

- **hybrid** - наш набор данных ruadapt_hybrid_instruct¹, состоящий из ≈ 80 тыс. примеров, синтезированных моделью Qwen3-235B-A22B.
- **instruct_2507** - наш набор данных ruadapt_instruct_2507², состоящий из ≈ 100 тыс. примеров, которые синтезированы моделью Qwen3-235B-A22B-Instruct-2507.
- **T-Wix** - готовый набор данных T-Wix [16], состоящий из 500 тыс. высококачественных синтезированных примеров, является расширенным аналогом **hybrid**.

Для синтеза наших наборов использовались запросы из коллекции GrandMaster-PRO-MAX[2]. Для каждого запроса генерировались по 3 примера, из которых выбирался самый короткий, который соответствует языку запроса и не содержит иероглифов. Набор данных hybrid отличает от instruct содержанием рассуждений (блоки <think>...</think>).

МЕТОДОЛОГИЯ ОЦЕНКИ

Оценка качества в работе производилась с использованием фреймворка llmtf³. Логи оценки всех инструктивных датасетов (вместе с промптами и генерациями моделей) доступны на платформе huggingface⁴.

¹https://huggingface.co/datasets/RefalMachine/ruadapt_hybrid_instruct

²https://huggingface.co/datasets/RefalMachine/ruadapt_instruct_2507

³https://github.com/RefalMachine/llmtf_open

Базовые модели оценивались во few-shot режиме на отдельных наборах: NEREL [9], Summ⁵, MultiQ, USE, взятые из MERA [10], Text Copy (диагностика устойчивости генерации), FLORES (ru-en и en-ru) [11], а также enMMLU и ruMMLU.

При оценке инструктивных версий использовались данные llmtf категорий ifeval, knowledge, long, math, ner, sentiment, summary, translate. При этом для всех датасетов кроме mmlu (в данном случае и zero-shot и few-shot) использовался zero-shot режим.

ЭКСПЕРИМЕНТЫ

Главный вопрос во всей методологии адаптации больших языковых моделей к особенностям целевого языка - какие этапы действительно необходимы? Смена словаря токенизации (шаг 1) позволяет ускорить генерацию текстов, однако нет понимания, что действительно нужны другие действия кроме последующей калибровки модели (шаг 5). Для ответа на вопрос был проведен анализ влияния параметров конфигурации процедуры адаптации на итоговую эффективность обработки русскоязычных запросов для моделей серий Qwen3-8B и Qwen3-4B.

Обучение эмбеддингов для новых элементов словаря токенизации По сути единственным существенным гиперпараметром обучения эмбеддингов при фиксированном вычислительном бюджете (размер батча и число примеров) является скорость обучения (lr).

Согласно Таблице 1, оптимальная скорость обучения (lr) для эмбеддингов составила 5e-4. Однако даже при оптимальной lr наблюдались существенные потери качества по сравнению с базовой моделью, что подтверждает необходимость последующих шагов адаптации.

ТАБЛИЦА 1. Сводная таблица результатов экспериментов.

Подход	embeds	rank	alpha	lr	mean	NEREL	Summ	MultiQ	USE	Text Copy	FLORES	MMLU (en)	MMLU (ru)		
Qwen3-8B-Base	-	-	-	-	60.2	64.3	25.2	64.0	22.7	100.0	53.7	79.3	72.2		
+ Обучение эмбеддингов	-	-	-	1.0E-4	44.1	41.5	22.2	54.4	15.4	22.0	51.3	78.7	67.6		
	-	-	-	3.0E-4	57.0	56.2	24.1	66.7	15.9	90.5	53.4	78.4	70.4		
	-	-	-	5.0E-4	57.7	57.7	24.6	67.8	16.8	92.0	53.6	78.4	70.6		
+ Выравнивание	lora	128	x 1	2.5E-5	57.6	56.0	24.7	69.8	15.4	93.0	53.5	78.1	70.6		
			x 1	5.0E-5	57.5	55.6	24.5	69.6	15.6	92.5	53.6	78.1	70.9		
			x 1	1.0E-4	57.6	56.2	24.8	68.4	15.4	93.0	53.7	78.3	70.9		
			x 2	2.5E-5	57.5	55.9	24.8	69.2	15.1	92.5	53.6	78.2	70.9		
			x 2	5.0E-5	57.6	56.5	24.7	68.5	15.4	92.5	53.6	78.2	71.2		
			x 4	5.0E-5	57.4	54.7	24.8	66.4	16.3	93.5	53.8	78.1	71.4		
	fft	128	x 1	2.5E-5	57.6	56.2	24.7	69.7	15.2	92.5	53.5	78.2	70.7		
			x 1	5.0E-5	57.7	55.4	24.9	69.4	15.7	93.5	53.5	78.3	70.9		
			x 1	1.0E-4	57.7	56.4	24.8	68.3	15.7	93.5	53.7	78.3	71.2		
			x 2	2.5E-5	57.7	56.0	24.8	69.7	15.5	92.5	53.6	78.2	71.2		
			x 2	5.0E-5	57.5	56.0	24.8	68.4	15.5	92.5	53.5	78.2	71.2		
					512	x 2	5.0E-5	57.6	56.1	24.7	66.4	16.0	93.5	53.7	78.5
Дополнительные эксперименты															
Продолжение обучения эмбеддингов	fft	-	-	-	57.9	57.9	24.7	69.0	16.7	91.5	53.6	78.5	71.1		
Пропуск шага обучения эмбеддингов	fft	128	1	1.0E-4	54.5	54.6	24.2	61.3	9.2	86.5	51.6	78.5	70.5		
			1	3.0E-4	56.2	58.0	25.0	61.7	15.4	91.5	48.2	78.7	71.3		
			1	5.0E-4	56.6	57.9	24.8	64.7	16.0	93.0	47.1	78.4	71.2		
			2	5.0E-5	53.8	52.2	23.8	61.8	7.8	85.5	50.7	78.6	70.2		

Выравнивание базовой языковой модели Выравнивание модели подразумевает коррекцию работы внутренних слоев модели для учета всех особенностей новых матриц эмбеддингов. Однако возможно основные языковые знания модели сконцентрированы в слоях эмбеддингов, тогда коррекция не даст эффекта. Чтобы это проверить были протестированы различные конфигурации процедуры выравнивания.

Результаты тестирования приведены в Таблице 1. Под моделью "Обучение эмбеддингов", поверх которого идет "Выравнивание", подразумевается вариант первого шага, полученный с lr=5.0e-4. В случае полноценной процедуры выравнивания целесообразно применять адаптеры LoRA только

⁴https://huggingface.co/datasets/RefalMachine/llmtf_bench_data

⁵<https://huggingface.co/datasets/IlyaGusev/gazeta>

для обучения внутренних слоев, а внешние эмбединги продолжать обучать полноценным образом (`embeds=fft`). Ранг матриц LoRA (`rank`) не имеет смысла брать больше 128, поскольку дополнительные затраты по памяти с рангом 512 не дают значительных улучшений. Устоявшаяся формула коэффициента слияния адаптеров $\alpha = rank \times 2$ также оказывается оптимальной в этом случае. В то же время в контексте адаптации модели Qwen3-8B продолжение обучения только эмбедингов (этапа 1) на данных выравнивания имеет большую эффективность. Эксперимент с пропуском шага обучения эмбедингов (сразу выравнивание модели) показывает, что причина такого результата скорее всего связана с неполной насыщенностью параметров новых токенов: устойчивость процедуры выравнивания напрямую зависит от информативности поступающих в промежуточные слои эмбедингов токенов, поскольку иначе задача обучения усложняется необходимостью дополнительной расшифровки неоднозначных последовательностей.

Перенос языковых знаний и калибровка модели После настройки слоев эмбедингов можно приступить к сборке целевого варианта. На этом этапе возникают вопросы:

- Нужен ли перенос знаний или стоит сразу калибровать?
- Какие данные оптимальные для калибровки модели?

Влияние переноса знаний В качестве базовых моделей для этапа LEP использовались три варианта, отличающиеся глубиной предварительной адаптации:

- `part1`: Модель после обучения эмбедингов без выравнивания (`lr=5e-4`).
- `emb`: Модель после обучения эмбедингов на всем объеме данных без выравнивания (`part 1 + part 2` с `lr=5e-4`).
- `full`: Модель, прошедшая полную процедуру адаптации, включая дообучение LoRA-адаптеров (`rank=128`, `alpha=256`, `lr=5e-5`) и полное дообучение эмбедингов.

Результаты этих экспериментов подробно представлены в Таблице 2 и на Рисунке 1. Здесь уже видны существенные различия между подходами выравнивания. Во-первых, перенос методом LEP во всех случаях имеет кардинальный эффект и даже больший объем обучающей коллекции (T-Wix) не способен компенсировать его отсутствие. Во-вторых, при малом объеме данных калибровки (`hybrid`) продолжение обучения только эмбедингов (`emb`) не так эффективно как выравнивание (`full`), а в случае большего они и вовсе эквивалентны. В третьих, полный пропуск выравнивания (без продолжения обучения эмбедингов) приводит к существенному падению эффективности процедуры выравнивания.

ТАБЛИЦА 2. Влияние переноса и числа примеров в рамках калибровки Qwen3-8B.

LEP	Base	Dataset	mean	ifeval	knowledge	long	math	ner	sentiment	summary	translate
No	emb	hybrid	35.7	54.7	59.2	40.0	14.4	26.4	8.8	22.0	60.1
	emb	T-Wix	37.6	52.1	60.7	44.2	20.3	29.3	10.8	22.5	60.5
	full	hybrid	36.1	56.1	60.2	39.9	13.6	26.7	9.9	22.4	60.1
	full	T-Wix	38.0	53.6	61.2	42.8	21.1	29.4	12.5	22.6	60.3
	part1	T-Wix	37.2	52.6	60.6	43.4	18.7	28.3	10.7	22.5	60.6
Yes	emb	hybrid	43.6	72.8	59.2	52.8	40.9	29.6	11.4	22.0	59.9
	emb	T-Wix	44.2	74.3	59.5	53.8	41.6	30.6	11.7	22.1	59.9
	full	hybrid	44.0	73.3	59.1	54.2	42.2	30.0	11.0	22.2	59.9
	full	T-Wix	44.4	73.8	59.6	55.0	40.9	30.7	13.0	22.1	60.3
	part1	T-Wix	43.8	73.6	59.4	54.0	40.3	29.6	11.9	21.9	59.9

Влияние формата данных Для оценки влияния формата была проведена адаптация Qwen3-4B-Base (аналогичная варианту `full` для 8B модели), для которой существуют две инструктивные версии:

- Qwen3-4B-Instruct-2507
- Qwen3-4B

. Для этих версий были выполнены перенос знаний и калибровка модели (Таблица 3). Исходя из результатом можно сделать следующие выводы:

1. LEP эффективен для любых моделей полученных от единой базовой модели
2. Дообучение на данных в «родном» формате обеспечивает более качественную адаптацию модели к целевому стилю инструкций

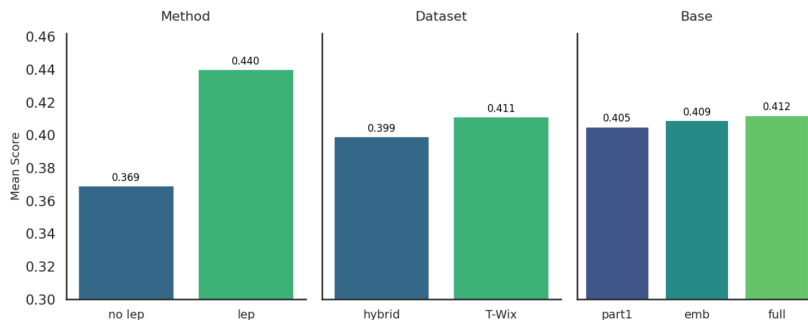


Рис. 1. Сравнительный анализ влияния компонентов на итоговую метрику Mean Score.

ТАБЛИЦА 3. Результаты переноса адаптации Qwen3-4B-base на другие модели.

LEP to	dataset	"родной" формат?	mean	ifeval	knowledge	long	math	ner	sentiment	summary	translate
Qwen3-4B	T-Wix	да	41.8	68.4	55.9	50.1	37.3	28.6	13.1	22.2	58.7
	instruct_2507	нет	40.5	68.3	54.8	50.5	29.9	28.5	11.5	22.3	58.4
Qwen3-4B-Instruct-2507	T-Wix	нет	41.7	72.4	57.3	52.4	26.7	28.8	14.9	21.5	59.2
	instruct_2507	да	44.2	73.9	57.1	53.4	46.5	29.1	13.5	21.3	58.9

Устойчивость моделей Устойчивость генерации оценивалась по доле ответов, не содержащих повторяющихся циклов. В качестве теста использовались математические задачи, на которых слабые модели склонны заикаться. Результаты представлены в Таблицах 4 и 5. Насколько можно видеть, в случае экспериментов с 8B моделью LEP снижает устойчивость, но в то же время без него модели почти в 3 раза реже решают задачи.

ТАБЛИЦА 4. Показатели устойчивости генерации для Qwen3-8B

LEP	Base	Dataset	Score	No Loop	Mean tokens
No	emb	hybrid	0.14	1.00	1900
	emb	T-Wix	0.20	1.00	1315
	full	hybrid	0.14	1.00	1916
	full	T-Wix	0.21	1.00	1100
	emb	hybrid	0.41	0.88	5349
	emb	T-Wix	0.42	0.96	6045
Yes	full	hybrid	0.42	0.85	4848
	full	T-Wix	0.41	0.94	5337

РЕЗУЛЬТАТЫ ПОЛНОЙ ПРОЦЕДУРЫ ЯЗЫКОВОЙ АДАПТАЦИИ

Итоговые результаты Для понимания общей эффективности процедуры мы сравнили полученные по описанной методологии Ruadapt модели с существующими открытыми аналогами. Результаты представлены в Таблице 6. Как можно видеть из таблицы, проблема снижения ifeval является общей для всех адаптируемых моделей с заменой токенизации. Как следствие оценки способностей моделей оказывается занижена, так как модели не соблюдают формат генерации ответа, как например, на sentiment категории для T-pro-it-2.0 [17].

Ускорение генерации за счет замены токенизации Для того, чтобы наглядно продемонстрировать эффект от замены токенизации мы произвели 2 типа замеров на русском языке: суммарное время на генерацию ответов к 500 запросам, а также время на обработку длинного запроса (300 тыс. символов). Результаты можно видеть на Рисунке 2. Переход на более подходящую токенизацию позволяет получить ускорение от 40% до 80% в зависимости от длины контекста.

АНАЛИЗ ВЫЧИСЛИТЕЛЬНОЙ СТОИМОСТИ

Для анализа экономии ресурсов мы также приводим сведения о затраченных вычислительных ресурсах в нормализованных (по оборудованию) GPU-часах. Под нормализованным GPU-часом подра-

ТАБЛИЦА 5. Показатели стабильности генерации для Qwen3-4B.

LEP to	dataset	"родной" формат?	Score	No Loop	Mean tokens
Qwen3-4B	T-Wix	да	0.37	0.89	5243
	instruct_2507	нет	0.30	0.62	1960
Qwen3-4B-Instruct-2507	T-Wix	нет	0.27	0.61	997
	instruct_2507	да	0.46	0.91	4774

ТАБЛИЦА 6. Оценка качества обученных Ruadapt моделей, исходных моделей и других существующих адаптаций.

Model	Tokenizer	mean	ifeval	knowledge	long	math	ner	sentiment	summary	translate
32B										
Qwen3-32B	original	53.6	79.3	68.0	58.0	63.5	50.0	24.3	23.1	62.5
RuadaptQwen3-32B-Hybrid	adapted	49.3	74.3	68.2	59.1	45.2	39.5	21.3	24.2	62.7
T-pro-it-2.0	adapted	49.0	66.8	69.3	56.3	61.3	43.0	10.5	21.9	62.8
8B										
Qwen3-8B	original	47.1	77.1	59.4	54.0	58.4	31.8	13.6	22.5	60.0
QVikhr-3-8B-Instruction	original	46.7	77.0	59.7	53.0	55.5	31.7	14.2	22.2	60.0
RuadaptQwen3-8B-Hybrid	adapted	44.4	73.8	59.6	55.0	40.9	30.7	13.0	22.1	60.3
4B										
Qwen3-4B	original	44.2	74.2	54.9	52.3	45.3	31.2	14.2	22.9	58.3
QVikhr-3-4B-Instruction	original	42.0	70.6	55.2	51.4	32.6	30.5	15.9	21.5	58.0
RuadaptQwen3-4B-Hybrid	adapted	40.5	68.3	54.8	50.5	29.9	28.5	11.5	22.3	58.4
Qwen3-4B-Instruct-2507	original	45.0	77.9	56.7	50.0	48.8	31.7	14.7	21.5	58.7
RuadaptQwen3-4B-Instruct	adapted	44.2	73.9	57.1	53.4	46.5	29.1	13.5	21.3	58.9

зумеается час работы в переводе на видеокарту Nvidia A100. Общие данные приведены в Таблице 7. Стоимость оптимальной конфигурации адаптации (отмечены ✓) составляет менее 2000 GPU-часов, что в случае использования Nvidia DGX H100 (≈ 16 A100) позволяет провести адаптацию менее чем за неделю. Калибровка модели составляет всего 6-10% от стоимости всей процедуры, что означает применение метода LEP позволяет сократить стоимость адаптации второй и последующих моделей семейства (образованных от единой базовой модели) как минимум в **10 раз**.

Мы также применили описанную методологию Ruadapt к адаптации Qwen3-32B модели, что потребовало ≈ 10000 нормализованных GPU часов. При этом, аналогичная адаптированная модель T-pro-it-2.0 по различным оценкам потребовала бы ≈ 40000 GPU часов (40 миллиардов токенов и полное, не LoRa, обучение модели).

Касательно подготовки данных, если брать скорость генерации 1500 токенов в секунду (для 8 A100) для Qwen3-235B-A22B, то полная стоимость синтеза hybrid (650 млн. токенов) и instruct_2507 (420 млн. токенов) составляет 960 и 620 GPU-часов соответственно. Подготовка более масштабного набора, такого как T-Wix (5.6 млрд токенов), в тех же условиях потребует от 8.3 тыс. GPU-часов.

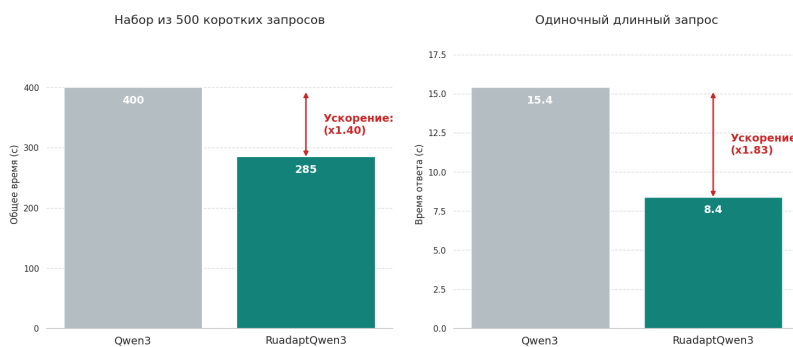


Рис. 2. Сравнение скорости генерации Qwen3 vs RuadaptQwen3

Таблица 7. Стоимость шагов адаптации Qwen3-8B.

	GPU-часы	
Обучение эмбеддингов	✓	995
Продолжение обучения эмбеддингов	✓	812
Выравнивание		1340
Калибровка (hybrid)	✓	108
Калибровка (T-Wix)		263
Общая стоимость процедуры		
Оптимальная (✓)		1915
Максимальная		2598

ЗАКЛЮЧЕНИЕ

В настоящей работе была предложена и экспериментально исследована комплексная методология Ruadapt для вычислительно эффективной языковой адаптации больших языковых моделей к русскому языку с заменой токенизации. Методология универсальна и работает для современных LLM любых размеров и семейств. Использование адаптированного токенизатора обеспечивает значительное ускорение генерации русскоязычного текста (от 40% до 80%) в символах по сравнению с исходными моделями. Проведенные эксперименты позволяют сформулировать ряд ключевых выводов и практических рекомендаций.

Основной эффект при адаптации достигается на этапе обучения эмбеддингов нового словаря токенизации, где критически важен подбор скорости обучения. Дополнительное обучение внутренних слоев модели (выравнивание) является опциональным, особенно если планируется использовать метод LEP (Learned Embedding Propagation) для переноса знаний.

Метод LEP оказывает существенное позитивное влияние на сохранение инструктивных навыков исходной модели. Однако для достижения максимального качества после его применения требуется дообучение (калибровка) на данных, формат которых соответствует стилю исходной модели. Показано, что хотя базовые адаптированные модели могут уступать в качестве исходным, этот разрыв полностью нивелируется после этапа калибровки на инструкциях. При этом наблюдается общая для всех адаптированных версий деградация на задачах категорий math и ifeval, однако она может быть решена дополнительными шагами обучения с подкреплением после адаптации, так как исходно эти навыки были интегрированы в модели именно за счет методов обучения с подкреплением.

При адаптации языковых моделей крайне важно оценивать уверенность и стабильность, которая часто не проявляется напрямую в бенчмарках. Так, например, несмотря на отсутствие существенной разницы между использованием датасетов hybrid и T-Wix, разницу можно увидеть в тесте на стабильность генераций, который показал, что в случае T-Wix модель намного реже уходит в циклы.

Предложенная методология делает языковую адаптацию LLM экономически целесообразной, снижая барьеры для исследовательских групп с ограниченными ресурсами. Полная адаптация 8B-модели занимает менее 2000 GPU-часов, а создание новых инструктивных версий на той же основе с помощью LEP обходится в 10 раз дешевле полной адаптации.

Хотя работа была сфокусирована на русском языке, наш подход является независимым от языка и может быть применен к другим языкам с алфавитной письменностью. Весь код адаптации доступен в открытом доступе.⁶

БЛАГОДАРНОСТИ

Работа выполнялась с использованием суперкомпьютера «МГУ-270» МГУ имени М.В. Ломоносова.

ФИНАНСИРОВАНИЕ РАБОТЫ

Исследование выполнено за счет гранта Российского научного фонда № 25-11-00191, <https://rscf.ru/project/25-11-00191/>

⁶<https://github.com/RefalMachine/ruadapt>

СПИСОК ЛИТЕРАТУРЫ

- [1] Tikhomirov M., Chernyshev D. Facilitating large language model russian adaptation with learned embedding propagation //Journal of Language and Education. – 2024. – Т. 10. – №. 4 (40). – С. 130-145.
- [2] Nikolich A. et al. Vikhr: Constructing a state-of-the-art bilingual open-source instruction-following large language model for Russian //Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024). – 2024. – С. 189-199.
- [3] Tikhomirov M., Chernyshev D. Impact of tokenization on llama russian adaptation //2023 Ivannikov Ispras Open Conference (ISPRAS). – IEEE, 2023. – С. 163-168.
- [4] Tikhomirov M. M., Chernyshev D. I. Improving Large Language Model Russian adaptation with preliminary vocabulary optimization //Lobachevskii Journal of Mathematics. – 2024. – Т. 45. – №. 7. – С. 3211-3219.
- [5] Mamedov V. et al. GigaChat Family: Efficient Russian Language Modeling Through Mixture of Experts Architecture //Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations). – 2025. – С. 93-106.
- [6] Fenogenova A. et al. MERA: A Comprehensive LLM Evaluation in Russian //Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). – 2024. – С. 9920-9948.
- [7] Ilya Gusev. rulm: A toolkit for training neural language models, 2023.
- [8] Penedo G. et al. FineWeb2: One Pipeline to Scale Them All—Adapting Pre-Training Data Processing to Every Language //arXiv preprint arXiv:2506.20920. – 2025.
- [9] Loukachevitch N. et al. NEREL: A Russian Dataset with Nested Named Entities, Relations and Events //Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021). – 2021. – С. 876-885.
- [10] Fenogenova A. et al. MERA: A Comprehensive LLM Evaluation in Russian //Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). – 2024. – С. 9920-9948.
- [11] Goyal N. et al. The flores-101 evaluation benchmark for low-resource and multilingual machine translation //Transactions of the Association for Computational Linguistics. – 2022. – Т. 10. – С. 522-538.
- [12] Fujii K. et al. Continual pre-training for cross-lingual llm adaptation: Enhancing japanese language capabilities //arXiv preprint arXiv:2404.17790. – 2024.
- [13] Cui Y., Yang Z., Yao X. Efficient and effective text encoding for chinese llama and alpaca //arXiv preprint arXiv:2304.08177. – 2023.
- [14] Cui Y., Yao X. Rethinking llm language adaptation: A case study on chinese mixtral //arXiv preprint arXiv:2403.01851. – 2024.
- [15] Hu E. J. et al. Lora: Low-rank adaptation of large language models //ICLR. – 2022. – Т. 1. – №. 2. – С. 3.
- [16] T-Технологии. Datasets T-Wix. URL: <https://huggingface.co/datasets/t-tech/T-Wix>.
- [17] T-Технологии. Модель T-pro-it-2.0. URL: <https://huggingface.co/t-tech/T-pro-it-2.0>.

RUADAPT: COST-EFFECTIVE LARGE LANGUAGE MODEL LINGUAL ADAPTATION

M. M. Tikhomirov^{1,*}, D. I. Chernyshev^{1,**}

¹Lomonosov Moscow State University, Research Computing Center,
Moscow, Russian Federation

Presented by Academician of the RAS B. S. Kashin

Multilingual Large Language Models (LLMs) often exhibit degraded performance for languages other than English due to the imbalance in their training data. Directly adapting these models to a new language, such as Russian, carries the risk of catastrophic forgetting their original capabilities and demands significant computational resources. In this paper, we introduce Ruadapt — a comprehensive and computationally efficient methodology for the language adaptation of LLMs, featuring tokenizer replacement. A full adaptation of a single Qwen3-8B model variant with our methodology requires less than 2000 GPU-hours, while subsequent adaptations of other variants are up to 10 times less resource-intensive due to the modular nature of the procedure's steps. An optimal configuration achieves up to an 80% speed-up in generation, with full preservation of long-context capabilities and only a minor degradation in instruction-following performance. We conduct a detailed empirical study of each adaptation step to identify optimal hyperparameters and to assess the impact of each key stage on the final quality. These resulting guidelines are implemented in the current generation of Ruadapt models, such as RuadaptQwen3-32B-Hybrid. We are open-sourcing our models, code, and datasets to provide the research community with a validated and cost-effective strategy for developing high-quality, language-specific models.

Keywords: large language models, language adaptation