Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра алгоритмических языков

Лебедев Андрей Алексеевич

Методы дообучения больших языковых моделей инструкциям для повышения качества работы на русском языке

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
Канд. физ.-мат. наук
М.М. Тихомиров

Москва, 2025

Аннотация

Методы дообучения больших языковых моделей
инструкциям для повышения качества
работы на русском языке

Лебедев Андрей Алексеевич

В рамках данной работы рассмотрены современные подходы к дообучению больших языковых моделей инструкциям для повышения их качества при обработке текстов на русском языке. Основное внимание уделено исследованию метода обучения с учителем (Supervised Fine-Tuning, SFT) и методов оптимизации пользовательских предпочтений, таких как обучение с подкреплением на основе обратной связи от человека (Reinforcement Learning from Human Feedback, RLHF), а также альтернативных подходов, в том числе Direct Preference Optimization (DPO), Simple Preference Optimization (SimPO) и Kahneman-Tversky Optimization (KTO). Практическая часть работы включает применение методов обучения с учителем и SimPO для выравнивания моделей на специализированных датасетах, а именно реализацию процесса обучения и оценку полученных моделей. Представлены результаты экспериментов, демонстрирующие количественные и качественные улучшения в производительности моделей.

Содержание

1 Введение

Развитие больших языковых моделей (БЯМ) за последние годы существенно расширило возможности обработки естественного языка. Начиная с модели GPT-3 [1], масштабирование параметров и обучающего корпуса привело к появлению универсальных моделей, способных решать широкий круг задач — генерацию текстов, автоматический перевод, ответы на вопросы и написание программного кода — без явной настройки под каждую задачу. Модели семейства GPT, LLaMA [4] и другие продемонстрировали, что при достаточно большом размере и качестве данных возможно достичь впечатляющих результатов в самых различных сценариях применения.

Одна из ключевых особенностей этих моделей — универсальность. Они обучаются на больших объёмах неструктурированных текстов и могут демонстрировать высокое качество в задачах, для которых не проходили специального обучения. Например, Codex [7], модель, производная от GPT-3, показала способность решать программные задачи, а GPT-4 [3] демонстрирует стабильное качество перевода [6], сопоставимое с результатами профессиональных переводчиков. Эта универсальность делает БЯМ привлекательными как основу для создания различных приложений: чат-ботов, систем поддержки клиентов, инструментов анализа документов.

Особый интерес вызывает применение БЯМ к русскоязычным задачам, однако большинство современных моделей изначально обучаются преимущественно на англоязычных данных, и потому их поведение на русском языке часто характеризуется увеличением количества ошибок и меньшей лексической выразительностью. Это связано как с морфологической сложностью русского языка, так и с особенностями токенизации, где русские слова разбиваются на большее количество токенов. Такие проблемы мотивируют исследователей разрабатывать методы адаптации существующих моделей или строить специализированные решения. Среди таких подходов можно выделить непрерывное обучение моделей на русскоязычных корпусах, как в работе VikhrModels [10], или расширение словаря модели через встраивание новых токенов в работе Ruadapt [11].

Помимо лингвистических сложностей, важным аспектом является управление поведением модели. Языковые модели по своей природе склонны к генерации правдоподобного, но потенциально недостоверного или неуместного текста. Для повышения

надёжности и соответствия ответов ожиданиям пользователей были предложены методы выравнивания моделей с человеческими предпочтениями. Классическим подходом является Reinforcement Learning from Human Feedback (RLHF) [2], в котором используется модель награды и алгоритм оптимизации стратегии. Однако из-за сложности и вычислительной стоимости RLHF были разработаны более эффективные методы: Direct Preference Optimization (DPO) [14], Simple Preference Optimization (SimPO) [15], Kahneman-Tversky Optimization (KTO) [16] и другие, которые позволяют выравнивать модели в соответствии с предпочтениями без дополнительной модели оценки.

Для объективной оценки качества моделей на русском языке необходимы соответствующие бенчмарки. В последние годы были разработаны специализированные тестовые наборы, такие как MERA [8], содержащий задания на генерацию, анализ, логическое мышление и кодогенерацию; RuMMLU [5], адаптация популярного бенчмарка MMLU; и LIBRA [9], направленный на проверку способности модели к работе с длинными входными последовательностями. Эти ресурсы позволяют проводить комплексную оценку моделей и сравнивать различные методы обучения и адаптации.

Разработка эффективных языковых моделей, адаптированных к русскому языку и выровненных с предпочтениями пользователей, представляет собой актуальную задачу. Современные методы обучения, в сочетании с новыми подходами к оценке, дают возможность систематически улучшать качество моделей без необходимости создания дорогостоящих решений с нуля. Эффективное использование открытых моделей и доступных корпусов инструкций открывает путь к построению качественных и доступных решений для русскоязычных пользователей.

Настоящая работа направлена на исследование различных подходов к адаптации больших языковых моделей для повышения их эффективности в русскоязычных задачах. Рассматриваются современные методы дообучения на инструкциях и выравнивания моделей с предпочтениями пользователей. В практической части основное внимание уделяется реализации подходов SFT и SimPO, обработке обучающих данных, а также проведению всесторонней оценки обученных моделей с использованием специализированных бенчмарков.

2 Обзор существующих методов

Важной задачей при разработке больших языковых моделей (БЯМ) является их адаптация для выполнения конкретных задач или работы в специфических языковых доменах, таких как русский язык. Для этого используются методы дообучения и выравнивания моделей с предпочтениями пользователя.

2.1 Выравнивание моделей

Выравнивание — процесс настройки моделей таким образом, чтобы их поведение соответствовало ожиданиям и предпочтениям пользователей. Важно, чтобы модели не только генерировали осмысленные ответы, но и учитывали вопросы безопасности, полезности и достоверности.

Принцип выравнивания часто описывается через правило «Helpful, Honest & Harmless» (Полезность, Правдивость & Безвредность):

- Helpful (полезность): ответы модели должны быть максимально полезными и релевантными запросу пользователя. Это предполагает глубокое понимание контекста и способность модели адаптироваться к различным задачам.
- Honest (честность): модель должна предоставлять достоверную информацию и избегать преднамеренной или непреднамеренной дезинформации, в том числе галлюцинаций [**?**].
- Harmless (безвредность): модель должна избегать генерации оскорбительного или иного нежелательного контента. Это особенно важно в контексте публичного использования, где модель может взаимодействовать с широким кругом пользователей.

Процесс выравнивания включает несколько этапов:

1. Сбор данных, отражающих предпочтения пользователей. Это может включать обратную связь от людей, ранжирование ответов модели или другие формы взаимодействия.

2. Настройка модели с использованием методов обучения с подкреплением, таких как Reinforcement Learning from Human Feedback [2], где награда определяется на основе человеческой обратной связи.

3. Тестирование и оценка выровненной модели для оценки её соответствия ожиданиям от адаптации.

2.2 Supervised Fine-Tuning

Метод SFT представляет собой процесс дообучения языковых моделей с использованием размеченных данных, который позволяет адаптировать их под конкретные задачи или улучшить качество работы в определённых условиях. Подходы, подобные LIMA [17], показывают, что даже с небольшим объёмом данных можно добиться заметных улучшений за счёт высокого качества обучающего набора.

SFT основан на обучении с учителем, где модель корректирует свои параметры для минимизации ошибки между предсказаниями и правильными ответами из размеченного набора данных. Ключевые элементы метода включают размеченные датасеты, содержащие примеры входных запросов и целевых выходов. В качестве функции потерь выступает кросс-энтропия, которая оценивает разницу между предсказанным распределением вероятностей и целевым распределением. Для настройки параметров модели используются методы оптимизации, такие как Adam или его модификации, которые корректируют веса модели в сторону уменьшения значения функции потерь.

Примеры применения SFT включают:

- Решение прикладных задач: дообучение модели для задач, таких как перевод, суммаризация, написание программ и других, что позволяет повысить контроль за поведением модели в определенном домене.

- Адаптация под специфический язык: использование русскоязычных данных для улучшения производительности модели при работе с текстами на русском языке.

- Следование инструкциям: обучение модели не просто продолжать текст, а отвечать на запросы пользователей.

- Подготовка к выравниванию: используется как предварительный этап перед применением методов, таких как RLHF [2].

Таблица 1: Анализ сильных и слабых сторон метода SFT

Преимущества	**Ограничения**
Обеспечивает устойчивость и воспроизводимость результатов при наличии корректно размеченных данных.	Требует большого объёма высококачественных размеченных данных, получение которых может быть трудозатратным.
Обладает сравнительно простой архитектурой обучения и не требует дополнительных этапов оптимизации.	Не учитывает сложные или неоднозначные пользовательские предпочтения, выходящие за рамки обучающего корпуса.
Хорошо адаптируется к конкретным задачам при наличии специализированных датасетов.	Склонен к переобучению при избыточной специфичности обучающих данных.
Позволяет осуществлять обучение с умеренными вычислительными затратами.	Не использует сравнительную или иерархическую информацию об ответах, ограничивая глубину обучения.

Несмотря на то, что дообучение модели на новых данных позволяет адаптировать её к конкретной задаче или языку, оно также может привести к увеличению числа «галлюцинаций». В работе Gekhman Z. и соавторов [18] показано, что fine-tuning способен как усиливать, так и подавлять галлюцинации в зависимости от структуры данных и способа обучения. Это подтверждается также результатами исследования Wee P. и Baghdadi R. [19], где продемонстрировано, что небольшие модели, дообученные на данных от более крупных моделей, склонны к генерации недостоверной информации из-за рассогласования в объёме знаний.

Поскольку в данной работе используются открытые датасеты, полученные синтетически, важно учитывать их потенциальные ограничения. Как отмечают Kang K. и соавторы [20], даже без явных ошибок в разметке, наличие нестандартных или нетипичных инструкций может повышать риск галлюцинаций у модели после SFT. Это необходимо учитывать при интерпретации результатов.

Таким образом, SFT является важным этапом в адаптации языковых моделей, позволяющим подготовить их для дальнейшего использования и совершенствования, но

он не охватывает весь спектр задач, связанных с учётом субъективных предпочтений пользователей. В частности, в случаях, когда требуются более гибкие и тонкие механизмы ранжирования ответов или оптимизация модели на основе сравнительных сигналов, применяются методы, ориентированные на обучение с учётом предпочтений.

2.3 Обучение с подкреплением: общая теория и применение в БЯМ

Обучение с подкреплением — это подход, при котором агент учится принимать оптимальные решения, взаимодействуя с окружающей средой. Основная цель агента — максимизировать суммарное вознаграждение, которое он получает за свои действия. Формально задача Reinforcement Learning (RL) описывается как Марковский процесс принятия решений, который включает:

- **Пространство состояний (S)**: набор возможных состояний среды, в которых может находиться агент.
- **Пространство действий (A)**: множество доступных агенту действий в каждом состоянии.
- **Функция переходов (P)**: вероятность перехода среды из одного состояния в другое в результате выполнения действия.
- **Функция награды (R)**: скалярное значение, которое агент получает за выполнение действия в определённом состоянии.

Основная задача агента заключается в том, чтобы научиться стратегии $\pi(a|s)$, которая максимизирует ожидаемую суммарную награду:

$$G = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma \cdot r_{t+1}\right],$$

где:

- G — общее ожидаемое вознаграждение;
- r_{t+1} — награда, полученная агентом на шаге $t+1$;

- $\gamma \in [0, 1]$ — коэффициент дисконтирования, который определяет важность будущих наград: чем меньше γ, тем меньше учитывается значение долгосрочных вознаграждений.

Применение RL в контексте БЯМ связано с задачей согласования моделей с человеческими предпочтениями. RL позволяет дообучать модели на основе обратной связи от человека, что делает ответы более соответствующими ожиданиям пользователей. Одним из важных элементов RL в БЯМ является функция награды, которая должна быть тщательно спроектирована. Чаще всего она формируется на основе человеческих оценок или автоматически сгенерированных метрик. Эта функция определяет, насколько ответы модели релевантны, точны и понятны.

Применение RL позволяет решать множество задач, включая:

- улучшение качества генерации текста;
- учёт специфических запросов и требований пользователя;
- минимизацию нежелательных или неуместных ответов.

2.4 Reinforcement Learning from Human Feedback (RLHF)

Метод RLHF [2] для адаптации БЯМ изначально был предложен Christiano P. F. и соавторами [13] как способ обучения с использованием оценок человека в качестве сигнала вознаграждения. Он представляет собой подход к адаптации БЯМ с использованием обратной связи от людей. Основная цель RLHF заключается в улучшении качества генерации текста, чтобы ответы модели лучше соответствовали ожиданиям и предпочтениям пользователей.

RLHF включает следующие этапы:

1. Модель (обычно полученная после этапа SFT) генерирует ответы на запросы из заранее подготовленного набора.
2. Асессоры ранжируют сгенерированные ответы по качеству.
3. На основе оценок обучается модель награды R_ψ, которая предсказывает, насколько хорош ответ.

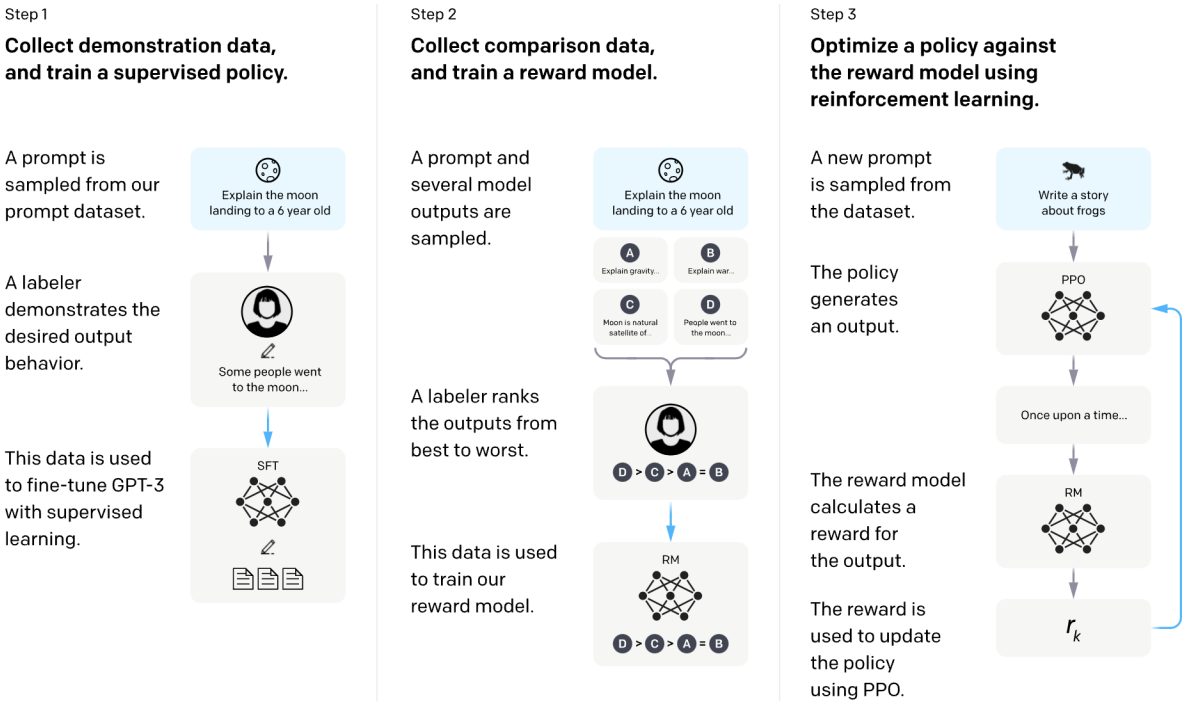

Рис. 1: Иллюстрация работы механизма RLHF.

4. Модель дообучается с использованием алгоритма Proximal Policy Optimization (PPO), чтобы максимизировать ожидаемую награду, предсказанную моделью награды.

Для обучения модели награды используется вероятность, что один ответ лучше другого. Это моделируется функцией:

$$P(a > b \mid s) = \sigma\big(r_\psi(s,a) - r_\psi(s,b)\big),$$

где s — запрос, a и b — два ответа, r_ψ — предсказанная награда, а σ — сигмоида. Параметры модели награды обучаются с использованием метода максимального правдоподобия, чтобы вероятность корректно отражала выбор асессоров.

Оптимизация языковой модели проводится с использованием метода Proximal Policy Optimization (PPO). Этот метод позволяет обучать стратегию π_θ так, чтобы она максимизировала ожидаемую награду:

$$J(\pi_\theta) = \mathbb{E}_{s\sim D,\, a\sim\pi_\theta}\big[r_\psi(s,a)\big],$$

где:

- $J(\pi_\theta)$ — целевая функция, которую необходимо максимизировать. Она выражает ожидаемую награду, получаемую агентом за действия, сгенерированные стратегией π_θ.

- $\pi_\theta(a|s)$ — стратегия с параметрами θ. Она определяет вероятность выбора действия a в состоянии s. В контексте БЯМ действие — это генерация следующего токена, а состояние — это контекст генерируемой последовательности.

- $s \sim D$ — запросы s выбираются из распределения D. Это означает, что запросы, на которые отвечает модель, поступают из некоторого заранее определённого набора данных.

- $a \sim \pi_\theta(a|s)$ — действия a генерируются на основе текущей стратегии π_θ.

- $r_\psi(s, a)$ — награда, предсказанная моделью награды R_ψ. Она отражает, насколько качественным считается действие a для состояния s.

Таблица 2: Анализ сильных и слабых сторон метода RLHF

Преимущества RLHF	**Ограничения RLHF**
Способность учитывать сложные предпочтения пользователей.	Высокая стоимость сбора данных обратной связи от асессоров.
Улучшение качества и согласованности ответов модели.	Сложность проектирования модели награды и алгоритма оптимизации.
Возможность работы с большим количеством запросов и сценариев.	Необходимость значительных вычислительных ресурсов.

Таким образом, RLHF представляет собой мощный инструмент для согласования больших языковых моделей с предпочтениями пользователей. Его использование особенно актуально в задачах, где требуется максимальная релевантность и надёжность ответов. Также он стал основой для развития других подходов к адаптации языковых моделей, которые в настоящее время используются для обучения многих популярных БЯМ.

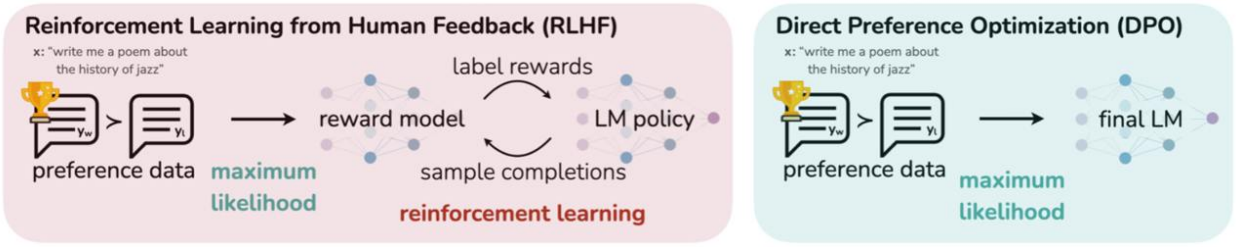

Рис. 2: Иллюстрация работы механизма DPO и сравнение с RLHF.

2.5 Direct Preference Optimization (DPO)

Метод DPO [14] представляет собой развитие RLHF и позволяет обучать модель напрямую на основе предпочтений пользователя без необходимости в отдельной модели награды или сложных алгоритмах обучения с подкреплением, таких как PPO.

В основе метода DPO лежит вероятностная модель предпочтений, которая позволяет вычислять функцию потерь напрямую на основе предпочтений пользователей. Главная идея заключается в оптимизации стратегии модели π_θ таким образом, чтобы она генерировала ответы, наиболее предпочтительные для пользователя. Для этого используется следующая формула для награды:

$$r_\theta(s, a) = \beta \log \frac{\pi_\theta(a|s)}{\pi_{\text{ref}}(a|s)},$$

где:

- s — запрос от пользователя.
- a — ответ, сгенерированный моделью.
- $\pi_\theta(a|s)$ — стратегия модели с параметрами θ.
- $\pi_{\text{ref}}(a|s)$ — референтная стратегия, обычно получаемая на этапе SFT.
- β — гиперпараметр, контролирующий масштаб награды.

Эта формула позволяет выразить предпочтение одного ответа над другим в терминах логарифма отношения вероятностей их генерации текущей моделью и референсной моделью.

DPO требует сбора такой же обучающей выборки, как и для модели награды в RLHF. Стратегия обучается классическим SFT, однако на специфический лосс. В отличие от PPO, в DPO нет необходимости генерировать данные во время обучения, как и нет необходимости учить отдельную модель награды. По сложности вычислений DPO всё же дороже обучения с учителем, потому что в функции ошибки участвуют вероятности не только обучаемой модели, но и SFT — её также требуется хранить в памяти. Тем не менее, DPO использует единый этап оптимизации и исключает модель награды, что упрощает процесс. Таким образом, DPO является более простой и экономичной альтернативой RLHF, подходящей для случаев, когда требуется минимизация вычислительных затрат при сохранении хорошего качества генерации и стабильности обучения.

Ограничения DPO:

- Эффективность метода сильно зависит от качества аннотаций и обучающей выборки.
- Приводит к несоответствию между генеративными моделями, используемыми при выводе, и моделью «награды».
- DPO использует две стратегии: π_θ и π_{ref}, что требует использования эталонной модели во время процесса обучения, увеличивая вычислительные потребности.
- Побочный эффект: модель начинает генерировать необоснованно длинные последовательности.

2.6 Simple Preference Optimization (SimPO)

Метод SimPO [15] представляет собой упрощённый подход к обучению языковых моделей, направленный на устранение некоторых ограничений DPO. SimPO фокусируется на прямой оптимизации предпочтений без использования сложных моделей награды и референсных стратегий, что делает его более эффективным и менее ресурсозатратным.

SimPO строится на упрощённой функции награды, основанной на метрике правдоподобия. Эта метрика позволяет оценивать качество ответа модели напрямую, исключая необходимость использования дополнительной эталонной стратегии. Награда задаётся

следующим образом:

$$r_{\text{SimPO}}(x, y) = \frac{\beta}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x,\ y_{<i}),$$

где:

- x — запрос, поступающий на вход модели.
- $y = (y_1, y_2, \ldots, y_{|y|})$ — сгенерированный ответ.
- $\pi_\theta(y_i \mid x, y_{<i})$ — вероятность предсказания токена y_i при условии запроса x и предыдущих токенов ответа $y_{<i}$.
- β — гиперпараметр, определяющий масштаб награды.
- $|y|$ — длина ответа.

Эта формула обладает некоторыми преимуществами по сравнению с DPO. Во-первых, награда прямо пропорциональна вероятностной функции, используемой для генерации ответов, что делает процесс обучения более согласованным. Во-вторых, исчезает необходимость в эталонной модели (π_{ref}), что уменьшает вычислительные затраты и снижает требования к памяти.

Для улучшения способности модели различать предпочтительные ответы вводится целевая «маржа» γ, которая регулирует степень предпочтения одного ответа над другим:

$$p(y_w \succ y_l \mid x) = \sigma\big(r(x, y_w) - r(x, y_l) - \gamma\big),$$

где:

- y_w и y_l — «победивший» и «проигравший» ответы, соответственно.
- σ — сигмоида, преобразующая разницу наград в вероятность.
- γ — гиперпараметр, целевая маржа, регулирующая требуемую разницу в наградах для предпочтения одного ответа над другим.

Маржа определяет, насколько один ответ должен превосходить другой, чтобы считаться лучше. Увеличение маржи приводит к улучшению обобщения, то есть к более чёткому различию между качественными и некачественными ответами.

Преимущества SimPO относительно DPO:

- Награда прямо пропорциональна метрике, которая фактически управляет генерацией.
- Отсутствует необходимость в эталонной модели, что приводит к большей эффективности вычислений и меньшим требованиям к памяти.
- Устойчивость к ошибкам в данных и гиперпараметрам благодаря упрощённой формулировке задачи.

Подводя итог, SimPO представляет собой значительное упрощение по сравнению с DPO, сохраняя при этом конкурентоспособные результаты в задачах выравнивания языковых моделей. Метод особенно полезен в условиях ограниченных ресурсов или необходимости быстрого прототипирования.

2.7 Kahneman-Tversky Optimization (KTO)

Метод KTO [16] основан на принципах теории перспектив, предложенной Канеманом и Тверски. Эта теория утверждает, что люди воспринимают потери и выигрыши асимметрично: потери вызывают более сильное эмоциональное воздействие, чем выигрыши той же величины. KTO адаптирует эту идею для обучения языковых моделей, делая акцент на минимизации вероятности нежелательных или неподходящих ответов.

KTO направлен на то, чтобы модель избегала ошибок с большей вероятностью, чем стремилась к генерации идеальных ответов. Это достигается за счёт использования бинарного сигнала обратной связи и модификации стандартных функций оптимизации.

Бинарная функция награды:

$$R(x) = \begin{cases} +1, & \text{если } x \text{ является приемлемым;} \\ -1, & \text{если } x \text{ является неприемлемым.} \end{cases}$$

Здесь x — это выходной ответ модели.

Для ограничения нежелательных результатов используется модифицированная формула KL-дивергенции:

$$D_{\mathrm{KL}}(P \,\|\, Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)},$$

где $P(x)$ — распределение вероятностей модели, а $Q(x)$ — эталонное распределение на основе человеческой обратной связи. Так, в KTO увеличивается штраф за повышение вероятности нежелательных ответов.

Преимущества KTO:

- Требуется только бинарная обратная связь от пользователя, что делает данные менее подверженными ошибкам аннотации и упрощает сбор обучающей выборки.
- Устойчивость: модель становится более надёжной в условиях неопределённости.

Метод KTO особенно полезен для задач, где критически важно минимизировать вероятность вредных или токсичных ответов. Например, он может быть использован в системах, предназначенных для взаимодействия с уязвимыми группами пользователей, или в контекстах, где ошибки могут иметь серьёзные последствия. Таким образом, KTO представляет собой подход, фокусирующийся на безопасности и надёжности языковых моделей, делая их более предсказуемыми и минимизируя риски в реальных сценариях использования.

2.8 Анализ методов обучения языковых моделей

Рассмотренные методы адаптации больших языковых моделей демонстрируют различные подходы к решению задачи выравнивания моделей с предпочтениями пользователей. Каждый из них имеет свои сильные и слабые стороны, которые определяют область их применения.

Общие ограничения методов:

- Все методы требуют высокого качества и большого объёма обучающих данных.
- Параметры требуют тщательной настройки, что может быть трудоёмким процессом.
- Методы могут снижать способность модели генерировать высококачественные ответы в пользу того, на что модель выравнивается.
- Сложно контролировать длину ответа модели.

Таблица 3: Сравнение методов адаптации больших языковых моделей

Метод	**Преимущества**	**Ограничения**
SFT	Простота реализации, достаточно высокая эффективность	Сильная зависимость от объёма и качества размеченных данных
RLHF	Учёт сложных предпочтений, улучшение качества	Высокая стоимость сбора данных, крупные вычислительные расходы
DPO	Простота обучения, отсутствие отдельной модели награды	Зависимость от эталонной стратегии и чувствительность к гиперпараметрам
SimPO	Минимизация вычислительных затрат, простота реализации	Требует разметки данных
KTO	Устойчивость, простота в сборе данных	Требует разметки данных

Среди современных методов адаптации языковых моделей к пользовательским предпочтениям выделяется SimPO, который демонстрирует превосходство над альтернативными подходами, включая DPO, KTO и SFT. В недавнем исследовании [15] метод SimPO был протестирован на моделях Mistral (7B) и Llama 3 (8B) на ряде бенчмарков, включая AlpacaEval 2, Arena-Hard и MT-Bench. По итогам экспериментов SimPO показал наибольшую долю предпочтений, отданных его ответам в сравнении с другими методами, что указывает на высокую согласованность ответов модели с оценками пользователей.

Дополнительным преимуществом данного подхода является отказ от использования отдельной референсной модели, что упрощает архитектуру обучения и повышает его устойчивость. В рамках данной работы метод SimPO выбран в качестве одного из основных для исследования методов оптимизации предпочтений. Для формирования обучающего корпуса есть возможность сгенерировать положительные примеры с помощью более сильной модели-эксперта, а отрицательные с использованием еще не оптимизированной модели.

3 Постановка задачи

Целью данной работы является изучение и применение современных подходов дообучения больших языковых моделей для повышения их качества работы на русском языке, а также поиск оптимального сочетания методов и данных для максимизации выбранных метрик качества. Для достижения этой цели необходимо решить следующие задачи:

- Изучить существующие подходы к выравниванию БЯМ, включая Supervised Fine-Tuning, Reinforcement Learning from Human Feedback, Direct Preference Optimization, Simple Preference Optimization и Kahneman-Tversky Optimization.
- Выполнить анализ доступных датасетов для адаптации языковых моделей на русский язык.
- Разработать и реализовать программное решение для дообучения выбранной модели.
- Провести эксперименты с различными методами дообучения.
- Найти, проанализировать и реализовать программно метрики, позволяющие эффективно оценивать качество получаемых моделей.
- Решить прикладную задачу повышения качества генерации при работе с длинным контекстом.
- Определить оптимальную комбинацию методов и данных, наиболее максимизирующую выбранные метрики.

В результате выполнения практической составляющей работы предполагается проведение сравнительного анализа различных методов выравнивания на инструкциях и данных на русском языке и выявление наиболее эффективного.

4 Практическая часть

4.1 Введение

На данном этапе исследования была выполнена программная реализация методов адаптации языковых моделей с использованием специализированных наборов данных и подходов обучения. В качестве базовой модели использовалась Qwen2.5–3B–Instruct — открытая и компактная языковая модель, обладающая сопоставимым качеством с более крупными архитектурами при значительно меньших вычислительных требованиях.

Основное внимание в экспериментальной части работы было уделено обучению моделей методом Supervised Fine-Tuning (SFT), как ключевому подходу дообучения БЯМ. Дополнительно был реализован и протестирован метод Simple Preference Optimization (SimPO), ориентированный на выравнивание модели по предпочтениям пользователя.

Целью экспериментов являлось сравнение различных стратегий дообучения моделей на корпусах с различной структурой и степенью качества, а также выявление влияния методологических решений на итоговую производительность моделей по широкому спектру метрик. Отдельным направлением стало решение прикладной задачи — повышение качества генерации при работе с длинным контекстом, что имеет важное значение для практического использования языковых моделей в диалоговом взаимодействии.

Далее представлены используемые датасеты и методы их подготовки, метрики оценки, а также описание хода экспериментов и анализ полученных результатов.

4.2 Датасеты

Для обучения и оценки языковых моделей в данной работе были использованы четыре открытых датасета, каждый из которых выполняет отдельную функцию в структуре экспериментов. Наборы различаются по типу аннотации, качеству и назначению: от прямого обучения с учителем до обучения с предпочтениями. Важной задачей на этом этапе являлась не только загрузка и предобработка данных, но и отбор релевантных примеров, необходимых для построения наиболее качественных обучающих выборок.

4.2.1 GrandMaster-PRO-MAX

Данный датасет был разработан в лаборатории Vikhrmodels и содержит около 156 000 уникальных пар «инструкция–ответ», синтетически созданных с использованием модели *GPT-4-Turbo*.

Подготовка данных включала:

- Перевод пар «инструкция–ответ» в шаблон, необходимый для загрузки в модель.
- Разделение на обучающую и валидационную выборки, перемешивание.
- Использование доли в 60 %, 80 % и 100 % от общего объёма данных для оценки влияния объёма на производительность модели.

4.2.2 Saiga Scored

Датасет был разработан Ильёй Гусевым и содержит 36 000 пар «инструкция–ответ», снабжённых оценками качества *opus_ score*, оценкой сложности и другими важными характеристиками. Оценка *opus_ score* была проведена с помощью сильной модели Claude 3 Opus по шкале от 1 до 10.

Подготовка данных включала:

- Перевод пар «инструкция–ответ» в шаблон, необходимый для загрузки в модель.
- Фильтрацию записей с `opus_score` ≥ 8 для обеспечения высокого качества обучающих данных.
- Разделение на обучающую и валидационную выборки, перемешивание.
- Использование 20 %, 40 %, 60 % и 100 % от общего объёма данных для оценки влияния объёма на производительность модели.

4.2.3 Grounded-RAG-RU-v2

Данный датасет создан в лаборатории Vikhrmodels и содержит синтетические данные с ответами на вопросы с опорой на внешний контекст (RAG-подход). Датасет содержит 50 000 примеров, разделенных на 2 типа: ood (out of distribution) – примеры,

где модели не дан в контексте ответ на вопрос, и она должна ответить соответствующе, а также good (обычные) примеры.

Подготовка данных включала:

- Оценка качества каждого примера в датасете с помощью сильной модели Qwen2.5-72B-Instruct по шкале от 1 до 5. Это дало возможность выбирать из датасета лучшие экземпляры.
- Перевод пар «инструкция–ответ» в шаблон, необходимый для загрузки в модель.
- Разделение на обучающую и валидационную выборки, перемешивание.

4.2.4 Saiga Preferences

Этот датасет был опубликован Ильёй Гусевым и содержит около 30 000 троек «запрос – плохой ответ – хороший ответ».

Подготовка данных включала:

- Перевод пар «инструкция–ответ» в шаблон, необходимый для загрузки в модель.
- Разделение на обучающую и валидационную выборки, перемешивание.

4.3 Методики оценки

4.3.1 LLM-as-a-Judge на наборе RU Arena Hard

Подход *LLM-as-a-Judge* на наборе *RU Arena Hard* заключается в автоматизированной сравнительной оценке пары ответов с использованием крупной языковой модели в роли судьи. Метод обеспечивает высокую корреляцию с человеческими оценками и позволяет масштабировать процедуру тестирования без привлечения аннотаторов.

Корпус RU Arena Hard представляет собой русифицированную версию бенчмарка Arena Hard Auto, включающего 500 заданий различной тематики, в том числе генерацию текстов, логическое рассуждение, математические задачи и код.

В качестве модели-судьи в рамках данной работы использовалась *Qwen2.5–72B–Instruct*. Для каждого задания судья анализировал пару ответов, один — от исследуемой модели, второй — от одной из референсных, и определял предпочтительный вариант.

В качестве референсных моделей, с которыми проводилось сравнение, были выбраны:

- `gigachat_max_26.20_uncen`
- `gpt-4o-mini`
- `gpt-4-1106-preview`
- `Qwen2.5-32B-Instruct`

Промпты для оценки были адаптированы под формат на русском языке, а результатом выступала доля случаев, в которых ответ исследуемой модели был признан предпочтительным. Данная метрика позволила количественно зафиксировать улучшение качества генерации на русском языке, полученное в результате различных стратегий дообучения.

Параллельно с оценкой качества проводился анализ средней длины сгенерированных ответов. Важно, что иногда более длинные ответы получают более высокие оценки *LLM-as-a-Judge*, поэтому это позволяло контролировать, насколько успешно модель справляется с оценкой с учётом изменения длины ответа.

4.3.2 Бенчмарк LIBRA

Для оценки способности моделей работать с длинным контекстом использовался бенчмарк *LIBRA* (Long Input Benchmark for Russian Analysis), разработанный командой AI Forever. LIBRA представляет собой агрегированный корпус из 21 поднабора, ориентированных на задачи анализа и извлечения информации из текстов на русском языке. Его особенностью является наличие специально адаптированных версий известных QA-наборов, в которых ключевые факты распределены по длинным отрывкам текста, что требует от модели устойчивого внимания и способности к многошаговому рассуждению.

В рамках данной работы использовались два поднабора LIBRA:

- **ruBABILongQA1–5** — пять поднаборов, представляющих собой адаптированные на русский язык версии задач из датасета Babi Long, ориентированных на оценку способности моделей к обработке и запоминанию длинного контекста. Каж-

дый поднабор моделирует специфический тип рассуждения (например, причинно-следственные связи, отслеживание переменных, извлечение информации), при котором корректный ответ может быть получен только при учёте всей входной последовательности, содержащей до нескольких тысяч токенов.

- **ruQasper** — русскоязычная версия набора Qasper, представляющего собой корпус научных статей и сопутствующих вопросов к ним. Модель должна найти релевантный фрагмент текста в длинном документе и сформулировать осмысленный и информативный ответ.

Методика оценки включала прямой запуск модели на тестовых примерах каждого из поднаборов с использованием формализованных шаблонов запросов.

Методика оценки включала прямой запуск модели на тестовых примерах каждого из поднаборов с использованием формализованных шаблонов запросов. В частности, для поднаборов *ruBABILongQA1–5* тестирование проводилось отдельно для групп примеров с различной длиной входного контекста: 4 000, 8 000, 16 000 и 32 000 токенов. Это позволяет оценить, как поведение модели изменяется при увеличении объёма входной информации. Итоговая оценка для каждого поднабора вычислялась как среднее значение точности по всем четырём категориям длины. Такой подход обеспечивает более полное представление об устойчивости модели при работе с длинным контекстом.

Результаты оценивались по точности генерации (точность, F1-score или полнота в зависимости от поднабора), согласно принятой методике в рамках LIBRA. Выбор бенчмарка обусловлен его способностью не только проверять корректность ответов, но и выявлять деградацию качества генерации в зависимости от сложности и объёма входных данных — что актуально в контексте прикладной задачи, поставленной в данной работе.

4.3.3 Бенчмарк MMLU и его русскоязычной версии

Для оценки общего уровня фундаментальных знаний моделей использовался бенчмарк *MMLU* (Massive Multitask Language Understanding), а также его русскоязычная адаптация *MMLU-ru*, подготовленная сообществом NLP Core Team. MMLU представляет собой стандартную задачу множественного выбора с четырьмя вариантами ответа,

охватывающую 57 различных дисциплин, включая историю, математику, юриспруденцию, медицину, физику и др.

Бенчмарк ориентирован на проверку способности модели к обобщению и решению задач, выходящих за пределы поверхностного сопоставления текстов. Все задания разработаны таким образом, чтобы быть нечувствительными к вероятностной релевантности и требовать минимального контекста для получения ответа.

В русскоязычной версии *MMLU-ru* использовался автоматический перевод оригинального корпуса. Общий объём составляет около 16 000 заданий. Используемая метрика — *accuracy@1*, то есть доля правильно выбранных ответов.

Тестирование производилось в режиме zero-shot, без предоставления модели дополнительных примеров. Такой режим позволяет объективно сравнивать производительность моделей, обученных на разных данных и по различным стратегиям.

Использование MMLU и MMLU-ru в рамках настоящей работы выполняло две функции. Во-первых, как независимый и универсальный индикатор уровня знаний модели. Во-вторых, как контроль возможного ухудшения результатов вследствие дообучения. Поскольку подобное дообучение может приводить к катастрофической забывчивости (catastrophic forgetting), особенно при агрессивной адаптации под узкие задачи, MMLU служил в качестве проверочной точки на предмет деградации обобщённых когнитивных способностей модели.

4.3.4 Оценка качества перевода на наборе FLORES из бенчмарка Darumeru

Для оценки способности моделей к машинному переводу использовался поднабор *FLORES* из бенчмарка *RefalMachine/darumeru*. Он представляет собой русифицированную версию корпуса `FLORES-101` и включает пары предложений на русском и английском языках. Подсет охватывает широкий спектр тематик и синтаксических конструкций, что делает его репрезентативным для оценки генеративного качества на двух языках.

Оценка проводилась в двух направлениях:

- перевод с английского на русский (`en` $\rightarrow$ `ru`);
- перевод с русского на английский (`ru` $\rightarrow$ `en`).

В качестве основной метрики использовались значения *ROUGE-1* и *ROUGE-2*, вычисляемые между сгенерированным переводом и референсным. Итоговым значением служило среднее арифметическое между ними. Такая схема позволяет оценить как общее лексическое совпадение (ROUGE-1), так и точность восстановления фразовой структуры (ROUGE-2).

Метрика применялась для анализа влияния дообучения на способность модели к сохранению качества работы на двух языках.

4.4 Цель экспериментов

Экспериментальная часть работы направлена на изучение влияния различных стратегий дообучения языковых моделей на качество генерации текстов на русском языке. Основной задачей являлось сопоставление эффективности метода SFT и метода оптимизации пользовательских предпочтений SimPO при обучении на различных наборах данных.

Особое внимание уделялось анализу влияния состава и объёма данных на итоговые характеристики модели: её способность генерировать качественные, содержательные и полезные ответы, а также сохранять фундаментальные знания, не переобучаясь на специфических датасетах. Отдельно рассматривался вопрос устойчивости модели при работе с длинным контекстом и способности к переносу выученных способностей между этапами обучения.

Каждая из моделей оценивалась по набору автоматических метрик, охватывающих как качество ответов с точки зрения пользовательских предпочтений, так и сохранность базовых знаний, способность к переводу и устойчивость при увеличении объёма входных данных.

4.5 Значения метрик исходной модели

В качестве исходной использовалась модель *Qwen2.5–3B–Instruct*. Она представляет собой открытую БЯМ с дообучением на инструкциях, но не адаптированную специально под русскоязычные задачи.

Значения метрик для исходной модели приведены в таблице 4.

Каждая из метрик в таблице характеризует отдельный аспект качества модели. По-

Таблица 4: Метрики для модели Qwen2.5–3B–Instruct (baseline)

Метрика	**Baseline (Qwen2.5–3B–Instruct)**
LLM-as-a-Judge	0.1355
Средняя длина ответа	418
ruMMLU	0.556
enMMLU	0.670
LIBRA: ruBABILongQA1	0.648
LIBRA: ruBABILongQA2	0.260
LIBRA: ruBABILongQA3	0.138
LIBRA: ruBABILongQA4	0.440
LIBRA: ruBABILongQA5	0.835
LIBRA: ruQasper	0.166
FLORES: en $\rightarrow$ ru	0.519
FLORES: ru $\rightarrow$ en	0.430

казатель *LLM-as-a-Judge* отражает долю пар, в которых ответы модели оказались предпочтительнее по сравнению с ответами эталонных моделей по мнению судьи – другой сильной LLM на наборе RU Arena Hard. *Средняя длина ответа* вычисляется как среднее количество токенов в сгенерированных ответах.

Метрики *ruMMLU* и *enMMLU* представляют собой точность (*accuracy@1*) при решении задач множественного выбора и измеряют базовые знания модели на русском и английском языках соответственно. Значения по поднаборам *LIBRA* (*ruBABILongQA1–5*, *ruQasper*) отражают среднюю точность по различным длинам входного контекста (4 000, 8 000, 16 000 и 32 000 токенов) и служат индикатором устойчивости модели при работе с длинными текстами. Наконец, метрики *FLORES* характеризуют способность модели к переводу и вычисляются как среднее между значениями *ROUGE-1* и *ROUGE-2*, сравнивающими сгенерированный перевод с эталонным на уровне лексических и фразовых совпадений.

Полученные значения метрик позволили зафиксировать точку отсчёта и впоследствии оценивать влияние различных стратегий дообучения.

4.6 SFT на датасете GrandMaster-PRO-MAX

Данный этап экспериментов был направлен на изучение влияния объёма качественных русскоязычных обучающих данных на производительность модели. В качестве датасета использовался GrandMaster-PRO-MAX, содержащий синтетически сгенерированные пары «инструкция–ответ» на основе модели *GPT-4-Turbo*. Дообучение проводилось методом Supervised Fine-Tuning на долях 60 %, 80 % и 100 % от общего объёма корпуса.

Полученные результаты по всем основным метрикам приведены в таблице 5.

Таблица 5: Результаты моделей, дообученных на GrandMaster-PRO-MAX

Метрика	**Baseline**	**+ 60 % GM**	**+ 80 % GM**	**+ 100 % GM**
LLM-as-a-Judge	0.1355	**0.1695**	0.1514	0.1480
Средняя длина ответа	418	568	549	551
ruMMLU	**0.556**	0.550	0.545	0.550
enMMLU	**0.670**	0.659	0.657	0.659
LIBRA: ruBABILongQA1	0.648	0.635	**0.657**	0.647
LIBRA: ruBABILongQA2	0.260	0.340	**0.365**	0.363
LIBRA: ruBABILongQA3	0.138	**0.165**	0.160	**0.165**
LIBRA: ruBABILongQA4	0.440	**0.478**	0.443	0.458
LIBRA: ruBABILongQA5	0.835	0.838	0.837	**0.845**
LIBRA: ruQasper	0.166	0.175	**0.190**	0.184
FLORES: en → ru	0.519	**0.520**	0.518	**0.520**
FLORES: ru → en	0.430	**0.451**	0.450	**0.451**

По столбцам расположены модели, полученные дообучением Qwen2.5–3B–Instruct на различных объёмах датасета GrandMaster-PRO-MAX (сокращенно GM).

Анализ. Дообучение на датасете GrandMaster-PRO-MAX демонстрирует стабильное улучшение по ключевым метрикам по сравнению с исходной моделью. Особенно заметен рост по *LLM-as-a-Judge*, что отражает улучшение качества генерации на русском языке. В то же время увеличение доли обучающих данных с 60 % до 100 % не даёт линейного прироста: наибольшая оценка *LLM-as-a-Judge* наблюдается при 60 % данных, а затем снижается, несмотря на рост средней длины ответа.

Показатели на *ruMMLU* и *enMMLU* слегка снижаются. Метрики по поднаборам LIBRA *ruBABILongQA1* и *ruQasper* демонстрируют сильный рост. Это говорит о том, что модель улучшает способность работать с длинным контекстом.

Результаты на *FLORES* также повышаются, что говорит следующее: дообучение на русскоязычном корпусе оказывает позитивное влияние на навыки перевода. Таким образом, SFT на GrandMaster-PRO-MAX обеспечивает рост практически по всем метрикам, что особенно заметно у модели, обученной на полном датасете.

4.7 SFT на датасете Saiga Scored

На следующем этапе была изучена эффективность дообучения на датасете *Saiga Scored*. Данный набор включает около 36 000 пар «инструкция–ответ» с автоматически сгенерированной оценкой качества каждого примера (*opus score*) по шкале от 1 до 10, полученной с помощью модели Claude 3 Opus. В экспериментах использовались только те примеры, для которых $opus\_score$ был не ниже 8 для обеспечения высокого качества обучающего набора с сохранением наибольшей доли датасета. По предварительным экспериментам, $opus\_score \geq 7$ и $opus\_score \geq 9$ давали результат хуже.

Дообучение проводилось сначала напрямую на Saiga Scored для оценки его влияния на исходную модель, а далее в комбинации с лучшей предварительно обученной на GrandMaster-PRO-MAX моделью. Использовались различные объёмы данных Saiga Scored (20 %, 40 %, 60 %, 100 %).

Полученные результаты по всем основным метрикам приведены в таблицах 6 и 7.

Анализ. Дообучение на Saiga Scored приводит к снижению на метрике *LLM-as-a-Judge*. Это связано с тем, что в датасете преобладают короткие и лаконичные ответы, что снижает среднюю длину генераций и ограничивает их содержательность.

Добавление Saiga Scored к модели, уже дообученной на GrandMaster-PRO-MAX, в среднем объёме (40–60 %) приводит к улучшению по большинству метрик: растут значения *MMLU*, *LIBRA* и особенно *FLORES*. Это говорит о положительном эффекте датасета на способности модели целом.

Таким образом, данные эксперименты подтверждают важность сбалансированного включения корпуса Saiga Scored в обучение. Комбинация с GrandMaster-PRO-MAX позволяет объединить сильные стороны обоих подходов: естественность и полноту — с

Таблица 6: Результаты моделей, дообученных на Saiga Scored (часть 1)

Метрика	**Baseline**	**+ 100% Saiga**	**+ GM + 20% Saiga**
LLM-as-a-Judge	**0.1355**	0.0865	0.1132
Средняя длина ответа	418	410	480
ruMMLU	0.556	0.552	0.557
enMMLU	0.670	**0.671**	0.667
LIBRA: ruBABILongQA1	0.648	0.640	**0.663**
LIBRA: ruBABILongQA2	0.260	0.373	0.362
LIBRA: ruBABILongQA3	**0.138**	0.075	0.082
LIBRA: ruBABILongQA4	0.440	0.455	0.490
LIBRA: ruBABILongQA5	0.835	**0.850**	0.840
LIBRA: ruQasper	0.166	0.196	**0.203**
FLORES: en → ru	0.519	0.521	0.520
FLORES: ru → en	0.430	0.453	0.454

Таблица 7: Результаты моделей, дообученных на Saiga Scored (часть 2)

Метрика	**+ GM + 40% Saiga**	**+ GM + 60% Saiga**	**+ GM + 100% Saiga**
LLM-as-a-Judge	0.1061	0.1076	0.1048
Средняя длина ответа	470	456	438
ruMMLU	0.560	0.557	**0.563**
enMMLU	0.667	0.668	0.668
LIBRA: ruBABILongQA1	0.653	0.652	0.645
LIBRA: ruBABILongQA2	**0.380**	0.378	0.350
LIBRA: ruBABILongQA3	0.092	0.075	0.080
LIBRA: ruBABILongQA4	0.480	**0.492**	0.482
LIBRA: ruBABILongQA5	0.845	0.837	0.830
LIBRA: ruQasper	0.195	0.190	0.193
FLORES: en → ru	**0.523**	0.521	**0.523**
FLORES: ru → en	0.455	**0.457**	**0.457**

одной стороны, и знания и краткость — с другой.

4.8 SFT на датасете Grounded-RAG-RU-v2

Следующий этап экспериментов был направлен на изучение влияния корпуса *Grounded-RAG-RU-v2* (далее и в таблице сокращённо RAG) на производительность модели. Данный датасет состоит из инструкций и ответов и включает два типа примеров: *good* — когда ответ может быть построен на основе предоставленного контекста, и *ood* (out-of-distribution) — когда в контексте отсутствует информация, необходимая для ответа.

Каждый пример в датасете был оценён с использованием модели Qwen2.5–72B-Instruct по пятибалльной шкале. Для обучения использовались только примеры с оценкой 4 или 5, а также отфильтрованные `ood`-примеры в небольшом объёме. Часть `ood`-данных сохранено для формирования у модели поведения отказа от ответа при отсутствии релевантной информации.

В таблице 8 представлены метрики для четырёх конфигураций: исходная модель, модель, дообученная только на RAG, и модели, в которые RAG был добавлен после этапов обучения на двух предыдущих датасетах.

Таблица 8: Результаты моделей, дообученных на Grounded-RAG-RU-v2

Метрика	**Baseline**	**+ RAG**	**+ GM + Saiga + RAG (4/5, 10% ood)**	**+ GM + Saiga + RAG (5/5, 5% ood)**
LLM-as-a-Judge	0.1355	**0.1846**	0.1175	0.1421
Средняя длина ответа	418	467	481	503
ruMMLU	**0.556**	0.553	0.546	0.551
enMMLU	0.670	**0.674**	0.667	0.671
LIBRA: ruBABILongQA1	0.648	0.642	0.660	**0.690**
LIBRA: ruBABILongQA2	0.260	0.240	**0.368**	0.330
LIBRA: ruBABILongQA3	0.138	**0.158**	0.120	0.135
LIBRA: ruBABILongQA4	0.440	0.465	0.480	**0.488**
LIBRA: ruBABILongQA5	**0.835**	0.815	**0.835**	0.827
LIBRA: ruQasper	0.166	0.175	**0.183**	0.178
FLORES: en → ru	0.519	0.516	0.512	**0.519**
FLORES: ru → en	0.430	0.446	**0.456**	0.455

Анализ. Добавление RAG без других корпусов приводит к росту по метрике *LLM-as-a-Judge*, а также увеличению средней длины ответов. При этом значения *ruMMLU* и *enMMLU* практически не изменяются, что свидетельствует об отсутствии деградации фундаментальных знаний.

Добавление RAG к лучшей модели, ранее обученной на GrandMaster-PRO-MAX и Saiga Scored, сохраняет общий рост длины ответа и повышает точность по *LIBRA*, особенно в поднаборе *ruBABILongQA1*, что отражает способность модели работать с длинным контекстом. Несмотря на невысокое значение *LLM-as-a-Judge* модель GM + Saiga + RAG (5/5, 5% ood) демонстрирует сбалансированное поведение по всем метрикам и может рассматриваться как универсальное решение.

Таким образом, Grounded-RAG-RU-v2 является ценным дополнением при обучении моделей на задачах, требующих интерпретации контекстной информации, и полезен при генерации с учётом наличия или отсутствия релевантных источников.

4.9 SimPO на датасете Saiga Preferences

Заключительный этап экспериментов был посвящён применению метода SimPO (Simple Preference Optimization) на специализированном датасете *Saiga Preferences*. Этот корпус представляет собой набор троек вида «запрос — плохой ответ — предпочтительный ответ», размеченных автоматически. Такой формат позволяет напрямую обучать модель различать хорошие и плохие генерации, не прибегая к отдельной модели награды, как в RLHF, или эталонной модели, как в DPO.

Для обучения использовался реализованный в библиотеке `trl` оптимизатор CPO (Contrastive Preference Optimization), который реализует процедуру оптимизации с использованием разности логарифмов правдоподобия предпочтительного и отклонённого ответа. Важными гиперпараметрами обучения являются:

- `cpo_alpha` — коэффициент масштабирования функции потерь. Более высокие значения усиливают штраф за неправильное ранжирование пар, делая обучение агрессивнее.
- `max_length_ratio` — максимальное допустимое отношение длины предпочтительного ответа к длине отклонённого. Этот параметр ограничивает включение в обу-

чение пар, в которых один ответ значительно длиннее другого, предотвращая доминирование длины как фактора предпочтения.

SimPO применялся к лучшей модели, полученной на этапах SFT. Таким образом, процедура служила финальным этапом выравнивания модели по пользовательским предпочтениям.

Результаты эксперимента приведены в таблицах 9 и 10.

Таблица 9: Результаты моделей, дообученных на Saiga Preferences (часть 1)

Метрика	**Baseline**	**Best SFT**	**SP 40% (0.2, 1.7)**	**SP 60% (0.2, 1.7)**
LLM-as-a-Judge	0.1355	0.1421	0.1543	0.1713
Средняя длина ответа	418	503	540	581
ruMMLU	**0.556**	0.551	0.553	0.554
enMMLU	0.670	**0.671**	0.670	0.670
ruBABILongQA1	0.648	**0.690**	0.675	0.675
ruBABILongQA2	0.260	0.330	0.333	0.330
ruBABILongQA3	0.138	0.135	0.145	0.143
ruBABILongQA4	0.440	0.488	0.490	**0.493**
ruBABILongQA5	**0.835**	0.827	0.827	0.825
ruQasper	0.166	0.178	0.173	0.184
Flores ru $\rightarrow$ en	**0.519**	**0.519**	0.518	0.518
Flores en $\rightarrow$ ru	0.430	**0.455**	0.454	0.454

В таблицах в скобках к SimPO-версиям моделей указаны значения гиперпараметров обучения *cpo_alpha* и *max_length_ratio* соответственно.

Анализ. Метод SimPO позволяет значительно повысить качество генерации, измеряемое по *LLM-as-a-Judge*, а также приводит к росту средней длины ответа. При этом метрики по задачам знаний (ru/enMMLU) и работе с длинным контекстом (LIBRA) либо сохраняются, либо улучшаются, без признаков деградации. Это делает SimPO эффективным завершающим этапом адаптации модели к пользовательским предпочтениям.

Таблица 10: Результаты моделей, дообученных на Saiga Preferences (часть 2)

Метрика	**SP 100% (0.2, 1.7)**	**SP 100% (0.2, 1.5)**	**SP 100% (0.1, 1.5)**
LLM-as-a-Judge	**0.3137**	0.2130	0.2749
Средняя длина ответа	772	684	762
ruMMLU	0.551	0.552	0.552
enMMLU	0.669	0.670	0.669
ruBABILongQA1	0.645	0.643	0.628
ruBABILongQA2	0.335	0.335	**0.348**
ruBABILongQA3	**0.163**	0.155	0.147
ruBABILongQA4	0.485	0.475	0.477
ruBABILongQA5	0.827	0.823	0.818
ruQasper	0.183	**0.187**	0.183
Flores ru → en	0.508	0.508	0.506
Flores en → ru	0.447	0.447	0.444

4.9.1 Дообучение с использованием собственных генераций модели

В рамках дополнительного эксперимента был исследован альтернативный способ формирования обучающих примеров для метода SimPO. Вместо использования исходного столбца *rejected* в датасете Saiga Preferences, содержащего отклонённые ответы, сгенерированные сильными БЯМ, была проведена замена этих ответов на генерации той же модели, которую планировалось дообучать. То есть на генерации лучшей SFT-моделью. Таким образом, обучающая выборка отражала реальные слабые стороны самой модели, а не внешнюю оценку, что потенциально должно было привести к более точному выравниванию по её собственным ошибкам.

Были подготовлены две версии модифицированного датасета, отличающиеся параметрами генерации отклонённых ответов:

- **Вариант A**: генерации с `temperature = 0.3`, `top_p = 0.9`;
- **Вариант B**: генерации с `temperature = 1.0`, `top_p = 1.0`.

Обе версии использовались в обучении модели методом SimPO. Результаты представлены в таблице 11.

Таблица 11: Результаты дообучения с использованием собственных генераций в качестве отклонённых ответов

Метрика	**Baseline**	**t=0.3, top-p=0.9**	**t=1.0, top-p=1.0**
LLM-as-a-Judge	**0.1355**	0.1219	0.1317
Средняя длина ответа	418	449	507
ruMMLU	**0.556**	0.552	0.548
enMMLU	**0.670**	0.663	0.663
ruBABILongQA1	**0.648**	0.618	0.620
ruBABILongQA2	0.260	0.312	**0.353**
ruBABILongQA3	0.138	0.103	**0.145**
ruBABILongQA4	**0.440**	0.368	0.405
ruBABILongQA5	**0.835**	0.810	0.810
ruQasper	0.166	**0.183**	0.167
Flores ru → en	**0.519**	0.512	0.505
Flores en → ru	0.430	**0.451**	0.432

Вопреки изначальным ожиданиям, модифицированные датасеты не привели к улучшению качества генерации: все показатели оказались ниже, чем в предыдущих экспериментах с оригинальной версией Saiga Preferences, и даже ниже относительно исходной модели. Таким образом, замена отклонённых ответов на собственные генерации модели может привести к деградации качества.

4.10 Выводы на основе экспериментов

Проведённые эксперименты позволили всесторонне оценить влияние различных методов дообучения и конфигураций датасетов на качество работы на русском языке исходной модели. Основные наблюдения и выводы представлены ниже:

- SFT на синтетических данных позволяет существенно улучшить модель по большинству бенчмарков по сравнению с исходной моделью *Qwen2.5-3B-Instruct*. Лучшая конфигурация (100% GM + 60% Saiga + 100% RAG) демонстрирует высокое качество генерации и высокие результаты на длинных контекстах.

- SimPO показал значительное улучшение, особенно по метрике *LLM-as-a-Judge*. Максимальное значение в 0.3137 (SP 100%) более чем в два раза превышает результат базовой модели.
- Оптимальный компромисс по большинству метрик достигается в конфигурации SimPO 60% (0.2, 1.7): наблюдается стабильный рост по LLM-as-a-Judge, высокая точность на ru/en MMLU и отсутствие деградации на традиционных задачах.

5 Инструментальные средства

Все эксперименты по обучению и оценке моделей проводились на вычислительном кластере *МГУ–270*. Для выполнения задач использовалось 16 графических ускорителей *NVIDIA A100* с объёмом видеопамяти 80 ГБ каждая.

Программная реализация была выполнена на языке Python с использованием фреймворка `Hugging Face Transformers`. Структура кода была разделена на независимые модули, отвечающие за подготовку данных, обучение, оценку моделей и управление конфигурациями. Основной стек технологий включал:

- `Transformers`, `Datasets` — загрузка моделей, токенизация, работа с датасетами, обучение;
- `llmtf_open` – фреймворк для эффективной оценки моделей;
- `LoRA` (Low-Rank Adaptation) — техника, в основе которой обучаемые низкоранговые матрицы; позволяет адаптировать модель с меньшими вычислительными затратами [12]
- `trl` — реализация методов обучения с учётом предпочтений (SimPO);
- `BitsAndBytes` — загрузка моделей в 4-bit и 8-bit формате для экономии ресурсов;
- `vLLM` и кастомные интерфейсы — ускоренная генерация и интерфейс взаимодействия с LLM при оценке;
- `WandB`, `TensorBoard` — логгирование и визуализация процесса обучения.

Для обучения моделей методом Supervised Fine-Tuning (SFT) был реализован модульный класс `SFTTrainer`, обеспечивающий:

- автоматическую фильтрацию и формирование подвыборок обучающих данных;
- использование шаблонов диалога с кастомными chat-template;
- настройку токенизатора и модели, включая PEFT-конфигурации (LoRA);
- сохранение и объединение весов модели после обучения.

Для обучения с предпочтениями методом SimPO использовался отдельный модуль с применением класса `CPOTrainer` из библиотеки `trl`, а также специализированной подготовки датасета с фильтрацией, выравниванием и балансировкой пар «предпочитаемый — отклонённый» ответ. Были реализованы механизмы предотвращения дисбалансы длины, проверки допустимости пар и контроля полноты данных.

Для оценки моделей применялся централизованный скрипт, объединяющий все метрики в едином интерфейсе: *LIBRA* (*ruBABILongQA1–5*, *ruQasper*), *ruMMLU* и *enMMLU*, FLORES и *LLM-as-a-Judge* на *RU Arena Hard*.

6 Заключение

В рамках данной дипломной работы были исследованы современные методы дообучения больших языковых моделей с фокусом на русскоязычные задачи. Основное внимание уделялось подходам Supervised Fine-Tuning (SFT) и Simple Preference Optimization (SimPO), а также их влиянию на качество генерации текстов и способность моделей следовать предпочтениям пользователя при работе на русском языке.

Проведённые эксперименты подтвердили, что классический подход SFT на качественно подготовленных синтетических данных способен значительно улучшить базовую модель, существенно повышая её эффективность на различных специализированных задачах и бенчмарках.

Метод SimPO показал способность заметно улучшить воспринимаемое качество генерации текстов за счёт оптимизации модели по предпочтениям без использования референсной модели или модели награды. Результаты экспериментов свидетельствуют о том, что модели, дообученные на SimPO, значительно превосходят исходные по метрике LLM-as-a-Judge, однако требуют тонкой настройки гиперпараметров для поиска оптимальной комбинации и избежания чрезмерного увеличения длины ответов.

Таким образом, данный подход позволил эффективно совместить сильные стороны традиционного дообучения (SFT) и оптимизации предпочтений (SimPO), продемонстрировав высокий потенциал для дальнейших исследований и внедрения в практические приложения.

Список литературы

[1] **Brown T. B. et al.** Language Models are Few-Shot Learners // *Advances in Neural Information Processing Systems.* – 2020.

[2] **Ouyang L. et al.** Training language models to follow instructions with human feedback // *arXiv preprint* arXiv:2203.02155. – 2022.

[3] **OpenAI (Achiam J. et al.)** GPT-4 Technical Report // *arXiv preprint* arXiv:2303.08774. – 2023.

[4] **Touvron H. et al.** LLaMA: Open and Efficient Foundation Language Models // *arXiv preprint* arXiv:2302.13971. – 2023.

[5] **Hendrycks D. et al.** Measuring Massive Multitask Language Understanding // *arXiv preprint* arXiv:2009.03300. – 2021.

[6] **Yan J. et al.** Benchmarking GPT-4 against Human Translators // *arXiv preprint* arXiv:2411.13775. – 2024.

[7] **Chen M. et al.** Evaluating Large Language Models Trained on Code // *arXiv preprint* arXiv:2107.03374. – 2021.

[8] **Fenogenova A. et al.** MERA: A Comprehensive LLM Evaluation in Russian // *arXiv preprint* arXiv:2401.04531. – 2024.

[9] **Churin I. et al.** LIBRA: Long Input Benchmark for Russian Analysis // *arXiv preprint* arXiv:2408.02439. – 2024.

[10] **Nikolich A. et al.** Vikhr: The Family of Open-Source Instruction-Tuned LLMs for Russian // *arXiv preprint* arXiv:2405.13929. – 2024.

[11] **Tikhomirov M.; Chernyshev D.** Facilitating LLM Russian adaptation with Learned Embedding Propagation // *arXiv preprint* arXiv:2412.21140. – 2024.

[12] **Hu E. J. et al.** LoRA: Low-Rank Adaptation of Large Language Models // *arXiv preprint* arXiv:2106.09685. – 2021.

[13] **Christiano P. F. et al.** Deep reinforcement learning from human preferences // *Advances in Neural Information Processing Systems*. – 2017. – T. 30.

[14] **Rafailov R. et al.** Direct Preference Optimization: Your Language Model is Secretly a Reward Model // *Advances in Neural Information Processing Systems*. – 2024. – T. 36.

[15] **Meng Y.; Xia M.; Chen D.** SimPO: Simple Preference Optimization with a Reference-Free Reward // *Advances in Neural Information Processing Systems*. – 2024. – T. 37.

[16] **Ethayarajh K. et al.** KTO: Model Alignment as Prospect Theoretic Optimization // *arXiv preprint* arXiv:2402.01306. – 2024.

[17] **Zhou C. et al.** LIMA: Less is More for Alignment // *Advances in Neural Information Processing Systems*. – 2024. – T. 36.

[18] **Gekhman Z. et al.** Does Fine-Tuning LLMs on New Knowledge Encourage Hallucinations? // *arXiv preprint* arXiv:2405.05904. – 2024.

[19] **Wee P.; Baghdadi R.** Exploring the Knowledge Mismatch Hypothesis: Hallucination Propensity in Small Models Fine-tuned on Data from Larger Models // *arXiv preprint* arXiv:2411.00878. – 2024.

[20] **Kang K. et al.** Unfamiliar Finetuning Examples Control How Language Models Hallucinate // *arXiv preprint* arXiv:2403.05612. – 2024.