Разработка метода самокоррекции больших языковых моделей с помощью обучения с подкреплением.

Р. Р. Исаев, Е. А. Ильюшин

***Аннотация*—В данной работе рассматривается разработка метода самокоррекции больших языковых моделей на основе обучения с подкреплением. Актуальность исследования обусловлена нестабильностью качества ответов современных LLM, их склонностью к фактическим и логическим ошибкам, а также отсутствием встроенных механизмов самопроверки и исправления собственных ответов.**

Задача самокоррекции формализована как эпизодический марковский процесс принятия решений (MDP), где модель генерирует первичный ответ и затем корректирующую попытку, оценивая приращение качества между ними с помощью бинарной награды. В работе проанализированы и преодолены основные вызовы: смещение распределений, коллапс поведения и проблемы нечестной оптимизации награды. Рассмотрены существующие подходы к решению проблемы, включая test-time reasoning, цепочки рассуждений, подходы на основе prompting и fine-tuning, а также методы обучения с подкреплением.

Предложено решение на основе алгоритма обучения с подкреплением Advantage Actor-Critic, выбранного из соображений оптимального баланса между простотой реализации и эффективностью оптимизации. Описана архитектура и процесс двухэтапного обучения, направленного на стабильное формирование навыков самокоррекции. В работе представлены экспериментальные результаты, демонстрирующие повышение качества самокоррекции моделей на задачах математического характера.

Полученные результаты подтверждают практическую значимость предложенного метода, обеспечивая повышение надежности и устойчивости решений, генерируемых большими языковыми моделями.

***Ключевые слова*—LLM, самокоррекция, обучение с подкрепелением, A2C**

I. Введение

В последние годы наблюдается значительный прогресс в разработке больших языковых моделей (LLM), которые демонстрируют выдающиеся результаты в широком спектре задач: от генерации связного и осмысленного текста до решения сложных математических примеров и написания программного кода [1—5]. Эти модели активно применяются в научных и технических дисциплинах, где важна способность к построению метастратегий и использованию вычислений в реальном времени для повышения точности и надёжности результатов.

Однако, несмотря на впечатляющие достижения, качество ответов LLM остаётся нестабильным. Модели склонны допускать логические ошибки, фактические галлюцинации и противоречия в рассуждениях [6, 7]. Одной из ключевых нерешённых проблем остаётся неспособность моделей к самокоррекции — то есть к выявлению и исправлению собственных ошибок без внешней помощи. Самокоррекция (self-correction) представляет собой желательное свойство, позволяющее модели итеративно улучшать свои ответы [8, 9]. Это особенно важно с учётом известных ограничений LLM: отсутствия гарантированной точности выводов, чувствительности к формулировке запроса и нехватки встроенных механизмов верификации. Способность к самокоррекции может значительно повысить надёжность и устойчивость принимаемых моделью решений.

В данной работе мы подробно рассматриваем один из перспективных подходов к реализации механизма самокоррекции — обучение с подкреплением (reinforcement learning, RL). Мы начинаем с формальной постановки задачи и мотивации, объясняющей необходимость механизмов, позволяющих LLM проверять и исправлять собственные выводы непосредственно в процессе генерации. Далее в работе мы представим подробный обзор существующих исследований, которые легли в основу нашего подхода, обсудим конкретные методики реализации самокоррекции и экспериментально подтвердим их эффективность.

Работа структурирована следующим образом. В первой главе формулируется задача самокоррекции: описывается формат взаимодействия модели с задачей, представление процесса генерации в виде MDP, целевая функция и метрики качества. Также выделяются ключевые особенности задачи и предлагается план её решения. Во второй главе рассматриваются существующие подходы к самопроверке LLM: test-time reasoning, цепочки рассуждений (Chain-of-Thought), prompting и fine-tuning, а также подходы с использованием обучения с подкреплением и родственные направления. Третья глава посвящена разработанному нами подходу на основе A2C. Излагаются мотивация выбора алгоритма, формализация обучения, функции потерь, стратегия двухэтапного обучения, а также реализованная архитектура. В четвёртой главе описана практическая реализация: структура системы, ключевые компоненты, пример на одном сэмпле и технические детали, включая гиперпараметры и логирование. Пятая глава содержит результаты экспериментов и описание используемого вычислительного окружения. Заключительная, шестая глава подводит итоги исследования.

Статья получена 25 мая 2025

Исаев Рустам Русланович, МГУ им. М.В. Ломоносова, (email: isaevrustam.r@gmail.ru).

Ильюшин Евгений Альбинович, МГУ им. М.В. Ломоносова, (email: eugene.ilyushin@gmail.com).

II. Предыдущие работы

A. Ограничения существующих больших языковых моделей

Современные большие языковые модели демонстрируют впечатляющие результаты в разнообразных задачах, однако их ограниченность проявляется при попытке построения многошаговых, логически непротиворечивых рассуждений. Типичные сбои включают неконсистентные аргументы, неспособность поддерживать глобальную структуру вывода и склонность к поверхностной обработке ошибок. Несмотря на высокий уровень генеративных способностей, такие модели зачастую не пересматривают собственные решения даже при наличии явных несоответствий или провалов [7].

Недавние работы показали, что добавление внешнего контура верификации или многократных итераций вывода может частично компенсировать эти недостатки [8]. Тем не менее, большинство LLM обучены в парадигме одношагового вывода (single-shot), где исправление ошибок не предусмотрено архитектурно. Это делает их особенно уязвимыми в задачах, где требуется рефлексия над собственным результатом и способность к контекстуальному пересмотру своих решений [9].

Проблема усугубляется тем, что большинство подходов к самокоррекции основаны на жёстко заданных шаблонах (например, prompting схемах), которые не обеспечивают устойчивой генерализации. Таким образом, существует объективная потребность в обучении моделей механизмам внутренней самопроверки и корректировки, которые не опираются на внешние сигналы, а формируются как часть самой модели. Этот пробел и становится предметом активных исследований в области self-refinement и reinforcement learning with human feedback [10].

B. Test-time reasoning и проблемы самопроверки

Современные большие языковые модели обычно генерируют ответ однократно, без последующего анализа или самопроверки. В отличие от человека, способного осмысливать и пересматривать свои рассуждения, базовая архитектура трансформера не содержит встроенного механизма обратной связи. В результате модели нередко «уверенно» выдают некорректные ответы, не распознавая собственных ошибок и не предпринимая попыток их исправить [6, 11]. Это серьёзно ограничивает возможности надёжной настройки поведения модели: она может сгенерировать ошибочное или недопустимое решение без малейшей попытки его переосмыслить [12].

C. Цепочки рассуждений (Chain-of-Thought)

Популярные техники, такие как цепочки рассуждений (chain-of-thought, CoT), позволяют явно отображать промежуточные шаги решения и зачастую повышают логичность вывода. Однако даже при использовании CoT модели не демонстрируют надёжной самопроверки. Они способны уверенно прийти к ошибочному выводу, последовательно и без осознания допущенной ошибки [13, 14]. Это подчёркивает необходимость дополнительных механизмов самокоррекции, способных эффективно дополнять CoT и аналогичные подходы.

D. Prompting и Fine-tuning: ограничения существующих подходов

Проблема самокоррекции особенно актуальна в задачах, требующих сложного многошагового рассуждения, таких как математические вычисления или анализ кода. Современные подходы, основанные на подсказках (prompting) или тонкой настройке на примерах (fine-tuning on demonstrations), показывают ограниченную эффективность [11, 15]. Fine-tuning, помимо всего прочего, требует значительных вычислительных ресурсов и зачастую предполагает запуск нескольких моделей параллельно [16], что существенно усложняет его применение на практике. Более того, эти методы сталкиваются с проблемами сдвига распределения и коллапса поведения, когда модель вместо содержательной коррекции начинает выдавать шаблонные, поверхностные исправления, не повышая реальную точность результатов [6, 14].

E. Использование обучения с подкреплением для самокоррекции

Одним из перспективных путей реализации механизма самокоррекции в LLM является применение методов обучения с подкреплением[17]. Формально задача может быть представлена как обобщённый марковский процесс принятия решений (MDP) [18, 19], определяемый пятёркой $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, где:

- $\mathcal{S}$ — множество состояний, включающих текущую информацию о запросе, промежуточных выводах и гипотезах модели;
- $\mathcal{A}$ — множество действий, представляющих собой возможные правки, переформулировки или окончательные ответы;
- $P(s_{t+1} \mid s_t, a_t)$ — вероятностный переход между состояниями;
- $R(s_t, a_t)$ — функция вознаграждения, оценивающая полезность действий (например, $R = 1$ при получении корректного ответа и 0 в противном случае);
- $\gamma \in [0, 1]$ — коэффициент дисконтирования, определяющий вес будущих вознаграждений.

Цель агента (LLM), параметризованного политикой $\pi_\theta(a|s)$, — максимизировать ожидаемое суммарное вознаграждение:

$$J(\pi_\theta) = \mathbb{E}_{a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right].$$

Такая формулировка опирается на классическую постановку задачи обучения с подкреплением [17, 20]. В этом контексте RL позволяет агенту обучаться на собственных ошибках: корректируя поведение в процессе генерации, он учится выбирать действия, которые ведут к более точным и надёжным результатам. Такой подход обеспечивает гибкость и позволяет напрямую оптимизировать процесс вывода.

F. Самокоррекция и связанные направления

Задача самокоррекции тесно связана с рядом активно развивающихся направлений:

- Test-time compute и многошаговый вывод: модель может выполнять дополнительные итерации, пересматривая и уточняя свой ответ, что сродни поиску

с возвратом (backtracking) или отладке на лету [13, 14].
- Externalized reasoning (внешнее обоснование): модель может использовать внешние средства для проверки своих утверждений — исполнять код, обращаться к базе знаний или интернету, используя обратную связь для коррекции [21—23].
- Robust alignment: даже модели, обученные с использованием RLHF [10], могут ошибаться, и способность самостоятельно обнаружить и устранить такие ошибки делает систему более безопасной и устойчивой [24, 25].

Современные результаты и актуальные подходы

a) Тонкая настройка (SFT) и самокоррекция.: Исследование Zhang et al. (2024) [14] представило метод SCORE, который улучшает способность моделей к самокоррекции посредством тонкой настройки на основе собственных критических замечаний. Эксперименты проводились на моделях LLaMA-2-13B-chat и Gemma-7B-it с использованием внешнего верификатора на основе GPT-4.

Таблица I
Точность до (Acc_{init}) и после самокоррекции методом SCORE ($\text{Acc}_{\text{SCORE}}$) с сильным внешним верификатором (GPT-4) на датасете GSM8K по данным из [14].

Модель	Acc_{init}	$\text{Acc}_{\text{SCORE}}$	Δ
LLaMA-2-13B-chat	37.2%	41.4%	+4.2%
Gemma-7B-it	36.3%	42.5%	+6.2%

Таблица II
Точность до (Acc_{init}) и после самокоррекции методом SCORE ($\text{Acc}_{\text{SCORE}}$) с сильным внешним верификатором (GPT-4) на датасете CommonSenseQA по данным из [14].

Модель	Acc_{init}	$\text{Acc}_{\text{SCORE}}$	Δ
LLaMA-2-13B-chat	69.7%	72.4%	+2.7%
Gemma-7B-it	67.2%	75.0%	+7.8%

Их исследование показало, что метод SCORE позволяет существенно улучшить точность малых языковых моделей за счёт обучения на собственных ошибках. Как видно из таблицы, при использовании внешнего верификатора на основе GPT-4 точность модели LLaMA-2-13B-chat на датасете GSM8K увеличилась с 37.2% до 41.4%, что соответствует абсолютному приросту на 4.2 процентных пункта. Для модели Gemma-7B-it прирост оказался ещё более выраженным: с 36.3% до 42.% (на 6.2 п.п.). На задаче CommonSenseQA также наблюдается заметное улучшение: +2.7 п.п. для LLaMA и +7.8 п.п. для Gemma.

b) Тонкая настройка (SFT), обучение с подкреплением и самокоррекция.: Исследование [26] предложило подход S2R, сочетающий тонкую настройку и обучение с подкреплением для обучения моделей навыкам самоверификации и самокоррекции. Сначала модели инициализируются через тонкую настройку на специально созданных данных с траекториями самопроверки и исправления ошибок. Затем эти навыки дополнительно усиливаются через обучение с подкреплением на уровне как конечного результата, так и промежуточных шагов решения задач. Эксперименты проводились на моделях LLaMA-3.1-8B-Instruct, Qwen2-7B-Instruct и Qwen2.5-Math-7B, показав значительный рост точности на математических задачах.

Таблица III
Точность до (Acc_{init}) и после обучения методом S2R (Acc_{S2R}) на наборе данных MATH500 по результатам [26].

Модель	Acc_{init}	Acc_{S2R}	Δ
LLaMA-3.1-8B-Instruct	48.0%	55.0%	+7.0%
Qwen2-7B-Instruct	51.2%	65.4%	+14.2%
Qwen2.5-Math-7B	51.0%	84.4%	+33.4%

Результаты работы Ma et al. (2024) показывают, что подход S2R значительно улучшает точность языковых моделей за счет эффективного сочетания тонкой настройки и обучения с подкреплением. Особенно заметно это для специализированной модели Qwen2.5-Math-7B, где точность на наборе данных MATH500[27] увеличилась на +33.4%.

Таким образом, разработка эффективных механизмов самокоррекции через RL становится центральной задачей современной исследовательской повестки и перспективным путём к созданию нового поколения интерпретируемых, надёжных и адаптивных LLM.

III. Постановка задачи

Целью статьи является формализация задачи самокоррекции ответов большой языковой модели (LLM) и её решение на основе обучения с подкреплением. Ниже кратко изложены ключевые элементы постановки.

A. Цель и формат взаимодействия

Пусть задана обучающая выборка

$$\mathcal{D} = \{(x_i, y_i^*)\}_{i=1}^{N},$$

где x_i — текст задачи, y_i^* — эталонное решение. Модель реализует стохастическую политику π_θ, которая генерирует:

1) первичный ответ $\hat{y}_1 \sim \pi_\theta(\cdot \mid x)$,
2) корректировку $\hat{y}_2 \sim \pi_\theta(\cdot \mid x, \hat{y}_1, p)$,

где p — инструкция «проверь и исправь ошибку».

B. Формализация как MDP

Процесс самокоррекции рассматривается как эпизод длины $T = 2$ в MDP:

- **Состояние** $s_1 = [x], \quad s_2 = [x, \hat{y}_1, p]$;
- **Действие** $a_t = \hat{y}_t$ — текст, сгенерированный моделью на шаге t;
- **Награда** $r_t = r(\hat{y}_t, y^*) \in \{0, 1\}$ — бинарная метрика корректности;
- **Цель** — максимизировать

$$J(\theta) = \mathbb{E}_{(x,y^*)\sim\mathcal{D}}\big[r_1 + r_2\big].$$

C. Данные и домен задач

Поскольку задача самокоррекции особенно важна в тех областях, где промежуточные вычисления или рассуждения модели часто содержат ошибки, требующие быстрого обнаружения и исправления, для наших экспериментов был выбран датасет **MATH** [27]. Этот датасет содержит разнообразные математические задачи из

школьной и олимпиадной программ, где решения требуют многократных промежуточных рассуждений. Наличие такого сложного многшагового формата позволяет наглядно продемонстрировать преимущества двухэтапного подхода с первичным решением и последующей корректировкой.

D. Метрики качества самокоррекции

Для оценки эффективности самокоррекции используются:

- **Accuracy**$@t_1$ — доля задач, решённых правильно с первой попытки;
- **Accuracy**$@t_2$ — доля задач, решённых правильно со второй попытки;
- $\boldsymbol{\Delta(t_1, t_2)} = \text{Accuracy}@t_2 - \text{Accuracy}@t_1$ — эффект самокоррекции;
- $\boldsymbol{\Delta_{i\to c}}$ — доля исправлений с неверного ответа на верный;
- $\boldsymbol{\Delta_{c\to i}}$ — доля случаев, где корректный ответ испорчен повторной попыткой.

E. Особенности и вызовы задачи

Несмотря на чёткую формализацию задачи самокоррекции как MDP, в процессе обучения и оценки возникают фундаментальные сложности, влияющие на стабильность и обобщающую способность модели. Ниже приведены три ключевых вызова, которые имеют математическую природу.

a) Смещение распределений (Distribution shift).: При использовании оффлайн-обучения модель оптимизируется на ошибках, полученных из старых траекторий, сгенерированных предыдущей версией политики $\pi_{\theta_{\text{old}}}$. Однако на практике новая политика π_θ должна эффективно работать в онлайн-режиме. Это приводит к расхождению между распределениями действий:

$$\pi_{\theta_{\text{old}}}(a \mid s) \neq \pi_\theta(a \mid s)$$

что влечёт за собой смещение оценок градиента и может затруднить успешное обучение механизма самокоррекции. Проблема особенно актуальна при использовании off-policy данных, собранных старыми моделями.

b) Коллапс поведения (Behavior collapse).: Оптимизация ожидаемой награды может привести к вырождению стратегии:

$$\pi_\theta(\hat{y}_2 = \hat{y}_1 \mid s_2) \approx 1$$

то есть модель стремится не вносить никаких изменений, либо же делает тривиальные преобразования, не затрагивающие смысл. Это может быть связано с тем, что стратегия “ничего не менять” либо “сделать что-то формальное” может показывать лучшую ожидаемую награду, особенно при плохо обученном критике.

c) Нечестная оптимизация награды (Reward hacking).: Награда $r(\hat{y}_2, y^*)$ может быть положительной даже при косметических правках, не связанных с реальным улучшением ответа. Тогда модель максимизирует:

$$\mathbb{E}_{\hat{y}_2 \sim \pi_\theta(\cdot|s_2)}[r(\hat{y}_2, y^*)]$$

в то время как истинная цель — повышение *качества* решения — не достигается. Это особенно опасно при бинарной награде, где минимальные изменения могут искусственно привести к $r = 1$ без фактической пользы. “Reward hacking” также может выражаться в чрезмерном переобучении на паттерны, повышающие метрику, но не общее качество модели.

Для решения этих проблем необходимы online анные, регуляризация поведения и конструкция награды, поощряющая *реальные* исправления.

IV. Предлагаемое решение

В данном разделе кратко излагается метод самокоррекции LLM, построенный на алгоритме *Advantage Actor–Critic* (A2C). Мы описываем мотивы выбора A2C, формулируем функции потерь и показываем, как двухэтапная схема обучения позволяет устранить ключевые проблемы: смещение распределений, коллапс поведения и нечестную оптимизацию награды.

A. Мотивация выбора A2C

Рассматриваемая задача сводится к короткому эпизодическому MDP длины $T = 2$: первичный ответ $\hat{y}_1$ и попытка самокоррекции $\hat{y}_2$. Простейший REINFORCE [20, 28] даёт слишком высокую дисперсию градиента, а PPO [29] требует сложной инфраструктуры и тщательной настройки гиперпараметров. A2C [30] комбинирует преимущества обеих схем:

- снижает дисперсию за счёт базового критика $V_\phi(s)$;
- не использует обрезку плотностей $\text{clip}(\cdot)$, упрощая реализацию и ускоряя сходимость;
- масштабируется на небольшие LLM (2–8 B параметров), сохраняя приемлемые вычислительные затраты.

B. Функции потерь

Поскольку в предлагаемом подходе actor и critic реализуются в виде двух **отдельных моделей**, оптимизация осуществляется **раздельно для каждой**. При этом сохраняется структура Advantage Actor-Critic (A2C), где ключевую роль играет функция преимущества:

$$A_t = Q_{\pi_\theta}(s_t, a_t) - V_\phi(s_t),$$

где оценка $Q_{\pi_\theta}(s_t, a_t)$ аппроксимируется через следующую награду и прогноз критика на следующем шаге:

$$Q_{\pi_\theta}(s_t, a_t) \approx r_t + \gamma\, V_\phi(s_{t+1}).$$

Использование оценки Q_{π_θ} позволяет точнее учитывать долгосрочную полезность действий, по сравнению с использованием только немедленного вознаграждения r_t.

Обучение каждой модели происходит по собственной функции потерь:

a) Actor (политика).: Модель, параметризующая политику π_θ, обновляется с использованием policy gradient по направлению преимущества:

$$\begin{aligned} L_{\text{actor}}(\theta) = &-\mathbb{E}_{s_t, a_t \sim \pi_\theta}\left[A_t \cdot \log \pi_\theta(a_t \mid s_t)\right] \\ &- c \cdot \mathbb{E}_{s_t}\left[H\left(\pi_\theta(\cdot \mid s_t)\right)\right] \end{aligned}$$

где второе слагаемое представляет собой энтропийный бонус, способствующий разнообразию действий (exploration), а $c > 0$ — соответствующий коэффициент.

b) Critic (оценка состояний).: Отдельная модель, обучающаяся оценивать функцию $V_\phi(s_t)$, минимизирует среднеквадратичную ошибку относительно оценки Q_{π_θ}:

$$L_{\text{critic}}(\phi) = \mathbb{E}_{s_t, a_t \sim \pi_\theta} \left[\left(Q_{\pi_\theta}(s_t, a_t) - V_\phi(s_t) \right)^2 \right].$$

Каждая модель обучается независимо, опираясь на общую структуру эпизодов, но оптимизирует свою задачу: actor — генерацию действий с опорой на оценку критика, а critic — аппроксимацию ценности состояния. Разделение функций потерь и параметров моделей упрощает настройку, делает процесс обучения более модульным и совместимым с уже предобученными компонентами.

C. Двухэтапное обучение

Для повышения эффективности и стабильности обучения мы разделяем процесс на два этапа.

a) Этап I: Initialization.: На первом этапе происходит адаптация актёра и критика к задаче самокоррекции при сохранении близости к исходной политике. Мы минимизируем комбинированный лосс для $t = 2$, одновременно добавляя KL-регуляризацию для $t = 1$, чтобы первичный ответ оставался сопоставимым с поведенческой политикой π_{ref}:

$$\mathcal{L}_{\text{init}} = -r_2 + \beta_1 \, \mathrm{KL}\big(\pi_\theta(\cdot \mid s_1) \parallel \pi_{\text{ref}}(\cdot \mid s_1)\big).$$

Это позволяет модели «привыкнуть» к механизму самокоррекции, не теряя качества первого решения.

b) Этап II: Reward shaping.: На втором этапе мы усиливаем стимул к реальному улучшению ответа, вводя к награде второго шага дополнительный бонус, пропорциональный разнице $r_2 - r_1$. Обновлённая награда определяется как

$$r_2' = r_2 + \alpha \, (r_2 - r_1),$$

где $\alpha > 1$. Итоговая функция потерь на этом этапе

$$\mathcal{L}_{\text{shape}} = -\big(r_1 + r_2'\big) + \beta_2 \, \mathrm{KL}\big(\pi_\theta \parallel \pi_{\text{ref}}\big)$$

задаёт модель так, чтобы она не только стремилась к корректному ответу второй попытки, но и избегала ухудшения исходного решения.

D. Преимущества двухэтапного подхода

Двухэтапная схема решает сразу несколько проблем:

- **Стабильность обучения**: этап инициализации обеспечивает плавный переход от исходной политики, снижая риск коллапса поведения.
- **Корректность прогресса**: этап Reward shaping формирует четкий стимул к улучшению качества, подавляя «косметические» правки и хакерство бинарной награды.
- **Контроль сдвига распределений**: регуляризация KL на обоих этапах удерживает политику в допустимом окне отклонений от исходного поведения.

Представленная схема показала устойчивый прирост точности ($+3-4$ п. п. на поднаборе MATH), при этом доля исправлений $i \to c$ существенно превысила случаи порчи $c \to i$, что подтверждает её практическую эффективность.

V. Результаты

Обучение моделей осуществлялось на другом подмножестве задач, где каждая попытка ответа оценивалась при помощи reward-модели PRM. При тестировании моделей оценка качества происходила без использования PRM [31] — итоговый ответ сравнивался напрямую с эталонным решением, что позволило объективно измерить способность модели к исправлению собственных ошибок. Таким образом, модель обучалась с учётом оценки качества, но проверялась в условиях «чистой генерации».

Обучение проводилось на двух машинах с графическими ускорителями NVIDIA A100. Использовались современные языковые модели Gemma 2[32], LLaMA 3[33] разных размеров. Все модели были дообучены в онлайн-режиме с использованием предложенной двухшаговой схемы: первая генерация ответа, последующая инструкция на самопроверку, и повторная генерация с учётом предыдущего ответа и указания. Результаты представлены в таблице IV.

Таблица IV
Результаты оценки способности моделей к самокоррекции

Модель	**Acc@t1**	**Acc@t2**	$\Delta(t_1, t_2)$	$\Delta_{i \to c}$	$\Delta_{c \to i}$
Gemma (7B)	42.3%	46.1%	+3.8%	4.5%	1.4%
Gemma 2 (2B)	39.0%	42.2%	+3.2%	6.2%	0.3%
LLaMA 3 (8B)	44.0%	47.6%	+3.6%	5.5%	2.2%
LLaMA 3.2 (3B)	41.5%	43.2%	+1.7%	4.2%	1.9%

Результаты демонстрируют, что предложенный подход позволяет последовательно повышать точность ответов на втором шаге. Это подтверждает, что даже относительно небольшие LLM могут быть эффективно адаптированы к самопроверке с использованием правильно сконструированной схемы награждения.

Также важно отметить, что метод показал устойчивость: прирост точности за счёт исправлений значительно превосходит количество ухудшений (то есть порчи изначально правильных ответов). Это указывает на надёжность механизма формирования награды и его практическую применимость для задач с высоким порогом доверия.

A. Характеристики вычислительного окружения

Обучение моделей проводилось на выделенном сервере с высокопроизводительным оборудованием. В таблице V приведены основные характеристики вычислительного окружения.

Таблица V
Аппаратное и программное обеспечение, использованное в обучении

Компонент	**Характеристика**
Операционная система	Ubuntu 22.04.3 LTS (Linux 6.5.0-25-generic)
Процессор	Intel Xeon (Icelake), 44 физических ядер
Оперативная память	503 ГиБ
GPU	2 × NVIDIA A100 80GB PCIe
Версия драйвера NVIDIA	570.133.07
Версия CUDA	12.3 (nvcc 12.3.107)
PyTorch	2.5.1
Transformers	4.46.3

Обучение проводилось с использованием двух графических ускорителей A100, при этом остальные устройства не использовались. Нагрузка распределялась по двум GPU в рамках одного процесса, что позволило эффективно использовать 160 ГБ видеопамяти и ускорить сбор батчей. Все вычисления выполнялись в окружении Python с библиотеками PyTorch и Transformers, соответствующими указанным версиям.

VI. Заключение

Целью настоящей работы являлась разработка метода самокоррекции больших языковых моделей с применением обучения с подкреплением. Для достижения поставленной цели были решены следующие задачи:

1) Исследованы существующие методы, используемые для обучения самокоррекции LLM.
2) Проведён сравнительный анализ подходов SFT и обучения с подкреплением для самокоррекции.
3) Исследована и оценена эффективность применения различных подходов обучения с подкреплением в задаче тонкой настройки LLM.
4) Разработан и реализован метод самокоррекции, основанный на RL.
5) Проведены экспериментальные исследования разработанного метода.

На первом этапе был проведён обзор существующих подходов к обучению самокоррекции и анализ их ограничений. Экспериментальные результаты для моделей, обученных только с использованием обучения с учителем, показали, что без дополнительных механизмов корректировки качество ответов после самокоррекции зачастую не улучшается, а в ряде случаев даже снижается.

Анализ работ, применяющих RL для задачи самокоррекции, выявил, что существующие методы либо характеризуются высокой вычислительной сложностью (например, подходы на основе MCTS), либо сталкиваются с проблемами нестабильности обучения. Это обусловило необходимость разработки более эффективного и менее ресурсоёмкого решения, способного обеспечить устойчивое улучшение качества ответов моделей при самокоррекции.

В рамках данной работы был предложен двухстадийный метод обучения, сочетающий принципы инициализации поведения модели и динамического формирования награды. На первом этапе обучения осуществляется закрепление базовых ответов и минимизация отклонений от начальной политики модели. На втором этапе вводится механизм изменения награды, стимулирующий улучшение качества ответа и штрафующий ухудшение. Расчёт награды осуществляется по формуле, учитывающей разницу в качестве между исправленным и исходным ответами.

Ключевой особенностью разработанного метода является акцент на онлайн обучении, что позволяет динамически корректировать поведение модели в процессе обучения без необходимости полного переобучения на каждом этапе. Благодаря этому достигается баланс между эффективностью обучения и разумной вычислительной нагрузкой, что особенно важно при работе с крупными языковыми моделями.

Экспериментальные исследования, проведённые на ряде моделей (в том числе Gemma 7B, Gemma 2B, LLaMA 3 8B и LLaMA 3 3B), подтвердили эффективность предложенного метода. В отличие от моделей, обученных исключительно с использованием SFT, модели, обученные с применением предложенного подхода, демонстрировали стабильный рост точности после самокоррекции, а также значительное превышение доли успешных исправлений ошибок над долей ухудшений правильных ответов.

Таким образом, разработанный метод обеспечивает эффективное обучение способности моделей к самокоррекции с меньшими вычислительными затратами по сравнению с существующими RL-подходами и обладает высокой стабильностью. Перспективными направлениями дальнейших исследований являются оптимизация алгоритма расчёта награды, адаптация метода для более сложных типов задач, а также исследование возможностей применения предложенного подхода в условиях многократной самокоррекции и в задачах с длинными цепочками рассуждений.

Библиография

[1] Z. Shao и др., *DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models*, 2024. url: https://arxiv.org/abs/2402.03300.

[2] A. Lozhkov, R. Li и H. de Vries, *StarCoder 2 and The Stack v2: The Next Generation*, 2024. url: https://arxiv.org/abs/2402.19173.

[3] C. Team и др., *CodeGemma: Open Code Models Based on Gemma*, 2024. url: https://arxiv.org/abs/2406.11409.

[4] J. Ahn, R. Verma, R. Lou, D. Liu, R. Zhang и W. Yin, *Large Language Models for Mathematical Reasoning: Progresses and Challenges*, 2024. url: https://arxiv.org/abs/2402.00157.

[5] W. Shi и др., *Math-LLaVA: Bootstrapping Mathematical Reasoning for Multimodal Large Language Models*, 2024. url: https://arxiv.org/abs/2406.17294.

[6] S. Banerjee, A. Agarwal и S. Singla, *LLMs Will Always Hallucinate, and We Need to Live With This*, 2024. url: https://arxiv.org/abs/2409.05746.

[7] N. Mündler, J. He, S. Jenko и M. Vechev, *Self-contradictory Hallucinations of Large Language Models: Evaluation, Detection and Mitigation*, 2024. url: https://arxiv.org/abs/2305.15852.

[8] J. Huang и др., *Large Language Models Cannot Self-Correct Reasoning Yet*, 2024. url: https://arxiv.org/abs/2310.01798.

[9] D. Liu и др., *Large Language Models have Intrinsic Self-Correction Ability*, 2024. url: https://arxiv.org/abs/2406.15673.

[10] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg и D. Amodei, *Deep reinforcement learning from human preferences*, 2023. url: https://arxiv.org/abs/1706.03741.

[11] G. Kim, P. Baldi и S. McAleer, *Language Models can Solve Computer Tasks*, 2023. url: https://arxiv.org/abs/2303.17491.

[12] S. Welleck и др., «Generating Sequences by Learning to Self-Correct,» в *The Eleventh International Conference on Learning Representations*, 2023. url: https://openreview.net/forum?id=hH36JeQZDaO.

[13] A. Madaan и др., *Self-Refine: Iterative Refinement with Self-Feedback*, 2023. url: https://arxiv.org/abs/2303.17651.

[14] Y. Zhang и др., *Small Language Models Need Strong Verifiers to Self-Correct Reasoning*, 2024. url: https://arxiv.org/abs/2404.17140.

[15] A. Kumar и др., *Training Language Models to Self-Correct via Reinforcement Learning*, 2024. url: https://arxiv.org/abs/2409.12917.

[16] A. Havrilla и др., *GLoRe: When, Where, and How to Improve LLM Reasoning via Global and Local Refinements*, 2024. url: https://arxiv.org/abs/2402.10963.

[17] R. S. Sutton и A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd. MIT Press, 2018. url: https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf.

[18] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994. doi: 10.1002/9780470316887.

[19] S. M. Ross, *Introduction to Stochastic Dynamic Programming*. Academic Press, 1983.

[20] R. J. Williams, «Simple statistical gradient-following algorithms for connectionist reinforcement learning,» *Machine Learning*, т. 8, № 3–4, с. 229—256, 1992. doi: 10.1007/BF00992696.

[21] L. Gao и др., *PAL: Program-aided Language Models*, 2023. url: https://arxiv.org/abs/2211.10435.

[22] S. Yao и др., *ReAct: Synergizing Reasoning and Acting in Language Models*, 2023. url: https://arxiv.org/abs/2210.03629.

[23] A. Creswell, M. Shanahan и I. Higgins, *Selection-Inference: Exploiting Large Language Models for Interpretable Logical Reasoning*, 2022. url: https://arxiv.org/abs/2205.09712.

[24] L. Ouyang и др., *Training language models to follow instructions with human feedback*, 2022. url: https://arxiv.org/abs/2203.02155.

[25] Y. Bai и др., *Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback*, 2022. url: https://arxiv.org/abs/2204.05862.

[26] R. Ma и др., *S^2R: Teaching LLMs to Self-verify and Self-correct via Reinforcement Learning*, 2025. url: https://arxiv.org/abs/2502.12853.

[27] D. Hendrycks и др., «Measuring Mathematical Problem Solving With the MATH Dataset,» *CoRR*, т. abs/2103.03874, 2021. url: https://arxiv.org/abs/2103.03874.

[28] A. Ahmadian и др., *Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback in LLMs*, 2024. url: https://arxiv.org/abs/2402.14740.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford и O. Klimov, «Proximal Policy Optimization Algorithms,» *CoRR*, т. abs/1707.06347, 2017. url: http://arxiv.org/abs/1707.06347.

[30] V. Mnih и др., *Asynchronous Methods for Deep Reinforcement Learning*, 2016. url: https://arxiv.org/abs/1602.01783.

[31] Z. Zhang и др., *The Lessons of Developing Process Reward Models in Mathematical Reasoning*, 2025. url: https://arxiv.org/abs/2501.07301.

[32] G. Team и др., *Gemma 2: Improving Open Language Models at a Practical Size*, 2024. url: https://arxiv.org/abs/2408.00118.

[33] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian и Z. Ma, *The Llama 3 Herd of Models*, 2024. url: https://arxiv.org/abs/2407.21783.

Development of method for self-correction of large language models via reinforcement learning

Rustam R. Isaev, Eugene A. Ilyushin

***Abstract*—This work presents a reinforcement learning-based approach to the self-correction of large language models (LLMs). The motivation stems from the instability in output quality of current LLMs, their tendency to produce factual or logical errors, and the lack of built-in mechanisms for verifying and improving their own responses.**

We formalize self-correction as an episodic Markov Decision Process (MDP), where the model generates an initial response and a subsequent correction attempt, with a binary reward signal based on the improvement in quality. The study addresses major challenges in this setting, including distributional shift, behavioral collapse, and reward hacking. We survey existing approaches, such as test-time reasoning, chain-of-thought prompting, fine-tuning, and reinforcement learning-based strategies.

Our proposed method employs the Advantage Actor-Critic (A2C) algorithm, selected for its balance between implementation simplicity and optimization efficiency. We describe a two-stage training architecture designed to foster stable self-correction capabilities. Experimental results on mathematical problem-solving tasks demonstrate notable improvements in response quality.

These findings highlight the practical relevance of the proposed method, showing enhanced reliability and robustness in the outputs of large language models.

***Keywords*—LLM, self-correction, reinforcement learning, A2C**

References

[1] Z. Shao *et al.*, *Deepseekmath: Pushing the limits of mathematical reasoning in open language models*, 2024. [Online]. Available: https://arxiv.org/abs/2402.03300.

[2] A. Lozhkov, R. Li, and H. de Vries, *Starcoder 2 and the stack v2: The next generation*, 2024. [Online]. Available: https://arxiv.org/abs/2402.19173.

[3] C. Team *et al.*, *Codegemma: Open code models based on gemma*, 2024. [Online]. Available: https://arxiv.org/abs/2406.11409.

[4] J. Ahn, R. Verma, R. Lou, D. Liu, R. Zhang, and W. Yin, *Large language models for mathematical reasoning: Progresses and challenges*, 2024. [Online]. Available: https://arxiv.org/abs/2402.00157.

[5] W. Shi *et al.*, *Math-llava: Bootstrapping mathematical reasoning for multimodal large language models*, 2024. [Online]. Available: https://arxiv.org/abs/2406.17294.

[6] S. Banerjee, A. Agarwal, and S. Singla, *Llms will always hallucinate, and we need to live with this*, 2024. [Online]. Available: https://arxiv.org/abs/2409.05746.

[7] N. Mündler, J. He, S. Jenko, and M. Vechev, *Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation*, 2024. [Online]. Available: https://arxiv.org/abs/2305.15852.

[8] J. Huang *et al.*, *Large language models cannot self-correct reasoning yet*, 2024. [Online]. Available: https://arxiv.org/abs/2310.01798.

[9] D. Liu *et al.*, *Large language models have intrinsic self-correction ability*, 2024. [Online]. Available: https://arxiv.org/abs/2406.15673.

[10] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, *Deep reinforcement learning from human preferences*, 2023. [Online]. Available: https://arxiv.org/abs/1706.03741.

[11] G. Kim, P. Baldi, and S. McAleer, *Language models can solve computer tasks*, 2023. [Online]. Available: https://arxiv.org/abs/2303.17491.

[12] S. Welleck *et al.*, «Generating sequences by learning to self-correct», in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=hH36JeQZDaO.

[13] A. Madaan *et al.*, *Self-refine: Iterative refinement with self-feedback*, 2023. [Online]. Available: https://arxiv.org/abs/2303.17651.

[14] Y. Zhang *et al.*, *Small language models need strong verifiers to self-correct reasoning*, 2024. [Online]. Available: https://arxiv.org/abs/2404.17140.

[15] A. Kumar *et al.*, *Training language models to self-correct via reinforcement learning*, 2024. [Online]. Available: https://arxiv.org/abs/2409.12917.

[16] A. Havrilla *et al.*, *Glore: When, where, and how to improve llm reasoning via global and local refinements*, 2024. [Online]. Available: https://arxiv.org/abs/2402.10963.

[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd. MIT Press, 2018. [Online]. Available: https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf.

[18] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994. doi: 10.1002/9780470316887.

[19] S. M. Ross, *Introduction to Stochastic Dynamic Programming*. Academic Press, 1983.

[20] R. J. Williams, «Simple statistical gradient-following algorithms for connectionist reinforcement learning», *Machine Learning*, vol. 8, no. 3–4, pp. 229–256, 1992. doi: 10.1007/BF00992696.

[21] L. Gao *et al.*, *Pal: Program-aided language models*, 2023. [Online]. Available: https://arxiv.org/abs/2211.10435.

[22] S. Yao *et al.*, *React: Synergizing reasoning and acting in language models*, 2023. [Online]. Available: https://arxiv.org/abs/2210.03629.

[23] A. Creswell, M. Shanahan, and I. Higgins, *Selection-inference: Exploiting large language models for interpretable logical reasoning*, 2022. [Online]. Available: https://arxiv.org/abs/2205.09712.

[24] L. Ouyang *et al.*, *Training language models to follow instructions with human feedback*, 2022. [Online]. Available: https://arxiv.org/abs/2203.02155.

[25] Y. Bai *et al.*, *Training a helpful and harmless assistant with reinforcement learning from human feedback*, 2022. [Online]. Available: https://arxiv.org/abs/2204.05862.

[26] R. Ma *et al.*, *S^2r: Teaching llms to self-verify and self-correct via reinforcement learning*, 2025. [Online]. Available: https://arxiv.org/abs/2502.12853.

[27] D. Hendrycks *et al.*, «Measuring mathematical problem solving with the math dataset», *NeurIPS*, 2021.

[28] A. Ahmadian *et al.*, *Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms*, 2024. [Online]. Available: https://arxiv.org/abs/2402.14740.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, «Proximal policy optimization algorithms», *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347.

[30] V. Mnih *et al.*, *Asynchronous methods for deep reinforcement learning*, 2016. [Online]. Available: https://arxiv.org/abs/1602.01783.

[31] Z. Zhang *et al.*, *The lessons of developing process reward models in mathematical reasoning*, 2025. [Online]. Available: https://arxiv.org/abs/2501.07301.

[32] G. Team *et al.*, *Gemma 2: Improving open language models at a practical size*, 2024. [Online]. Available: https://arxiv.org/abs/2408.00118.

[33] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, and Z. Ma, *The llama 3 herd of models*, 2024. [Online]. Available: https://arxiv.org/abs/2407.21783.