Обзор сценариев захвата модели вознаграждения в LLM агентах

А.В. Карпинская, К.А. Айрапетьянц

***Аннотация*—Большие языковые модели (Large Language Models, LLM) всё чаще используются в качестве основы автономных агентов, взаимодействующих со средой и выполняющих последовательности действий. Для согласования их поведения с человеческими предпочтениями применяется обучение с подкреплением на основе человеческой обратной связи (Reinforcement Learning from Human Feedback, RLHF), использующее модель вознаграждения в качестве сигнала качества. Поскольку модель вознаграждения является лишь приближением истинных целей разработчика, агент может находить способы её эксплуатации. В отличие от существующих обзоров, классифицирующих взлом механизма вознаграждения по месту его возникновения в конвейере выравнивания, в настоящей работе классификация строится по другому основанию — компонентам архитектуры LLM-агента: языковому ядру, памяти, модулю планирования, инструментам, каналу наблюдений и механизму оценки. Каждому компоненту сопоставлены характерные сценарии эксплуатации, применимые методы защиты и доступные бенчмарки, что образует карту покрытия и выявляет незакрытые сочетания угроз и средств защиты. Метрики оценки систематизированы по требуемому уровню доступа к модели, необходимости эталонной разметки и участия человека; предложен минимальный протокол отчётности, обеспечивающий сопоставимость результатов.**

***Ключевые слова*—большие языковые модели, LLM-агенты, архитектура агента, обучение с подкреплением на основе человеческой обратной связи, модель вознаграждения, взлом механизма вознаграждения, согласование поведения, метрики оценки**

I. Введение

Развитие агентных архитектур на основе больших языковых моделей (Large Language Models, LLM) позволило создавать агентов, способных взаимодействовать со средой, использовать внешние инструменты и выполнять сложные последовательности действий. Основным подходом к согласованию их поведения с человеческими предпочтениями является обучение с подкреплением на основе человеческой обратной связи (Reinforcement Learning from Human Feedback, RLHF), в котором критерием качества выступает модель вознаграждения, обученная на человеческих предпочтениях [1]. Поскольку эта модель не является точным описанием намерений разработчика, агент может находить способы максимизации вознаграждения, не соответствующие ожидаемому поведению: игру со спецификацией (specification gaming), воздействие на механизм вознаграждения (reward tampering) и эксплуатацию модели предпочтений (preference hacking).

Статья получена 16 июня 2026

Анна Викторовна Карпинская, МГУ имени М.В. Ломоносова, (email: annakarpinckaya@gmail.com).

Каринэ Арсеновна Айрапетьянц, МГУ имени М.В. Ломоносова, (email: karine.ayrps@gmail.com).

A. Отличие от существующих обзоров

Опубликованный обзор [2] охватывает языковые, мультимодальные и генеративные модели и опирается на гипотезу сжатия прокси-сигнала: модуль оценивания представляет собой сжимающее отображение богатого набора признаков истинной цели в скалярную оценку, а взлом вознаграждения возникает как следствие потерь при таком сжатии. Механизмы эксплуатации классифицируются там по месту возникновения в конвейере выравнивания: уровни поверхностных признаков, внутренних представлений, модуля оценивания и среды.

Настоящая работа использует другое основание классификации. Классификация по месту возникновения отвечает на вопрос, какое звено процесса обучения оказывается эксплуатируемым, но не указывает, какой компонент развёрнутой системы служит поверхностью атаки и какими средствами он защищается. Между тем LLM-агент включает не только политику и модуль оценивания, но и память, модуль планирования, набор инструментов и канал наблюдений. Агентным системам в [2] отведена одна подсекция, а классические результаты теории взлома вознаграждения в RL-постановке в нём практически не рассматриваются.

Вклад работы: классификация сценариев по компонентам архитектуры LLM-агента, сопоставленная с уровнями механизмов из [2]; карта покрытия, сопоставляющая компонентам применимые методы защиты и доступные бенчмарки и выделяющая незакрытые сочетания; систематизация метрик оценки и минимальный протокол отчётности.

II. Теоретические основы и поверхности эксплуатации

A. Модель вознаграждения

Поведение LLM-агента описывается в рамках обучения с подкреплением: взаимодействие со средой задаётся марковским процессом принятия решений, политика $\pi(a|s)$ определяет вероятность выбора действия, а целью обучения является максимизация ожидаемого дисконтированного вознаграждения $J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_t \gamma^t r_t]$ [3]. Для LLM-агента состоянием выступает история диалога, содержимое памяти или описание задачи, а действиями — генерация токенов, вызов инструментов и внешние операции [4]. LLM-агентом далее называется программная система, использующая большую языковую модель

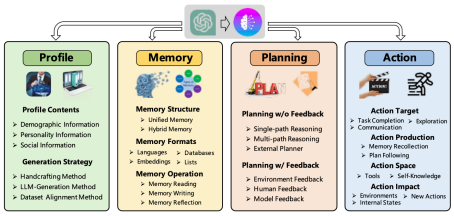

Рис. 1. Обобщённая архитектура LLM-агента [4]: языковая модель управляет четырьмя модулями — профилем, памятью, планированием и действиями — в едином цикле взаимодействия со средой.

для формирования последовательности действий на основе наблюдаемого состояния среды (рис. 1).

В RLHF функция вознаграждения не задаётся явно, а восстанавливается по данным о предпочтениях [1]. Моделью вознаграждения называется функция $r_\phi(x, y)$, сопоставляющая запросу x и ответу y скалярную оценку; предпочтения $y_w \succ y_l$ моделируются моделью Брэдли–Терри

$$P(y_w \succ y_l|x) = \sigma\left(r_\phi(x, y_w) - r_\phi(x, y_l)\right), \qquad (1)$$

где $\sigma(\cdot)$ — сигмоидная функция [1, 5], а параметры ϕ определяются минимизацией логистической функции потерь на множестве пар предпочтений. Далее r_ϕ используется как прокси-функция качества при оптимизации политики, вследствие чего агент оптимизирует не реальные предпочтения пользователя, а их приближение.

Определение II.1. Ошибкой спецификации вознаграждения будем называть разность $\Delta(x, y) = r_\phi(x, y) - R^*(x, y)$, где R^* — истинная функция предпочтений.

Источниками ненулевой Δ служат ограниченность данных предпочтений, распределительный сдвиг между обучающим и тестовым распределениями, систематические смещения оценок в пользу определённого стиля или длины ответа, а также частичная неидентифицируемость: одному набору предпочтений может соответствовать множество функций r_ϕ, приводящих к различным оптимальным политикам, что не устраняется увеличением объёма выборки [6]. Следовательно, максимизация r_ϕ не гарантирует максимизации истинной полезности решения.

B. Поверхности эксплуатации в архитектуре агента

Сказанное относится к конвейеру выравнивания и неявно предполагает, что агент воздействует только на собственный ответ. В развёрнутой агентной системе вознаграждение формируется по результатам работы нескольких компонентов, каждый из которых может стать самостоятельной поверхностью эксплуатации.

Прежде чем перечислять эти компоненты, разграничим две фазы, в которых возникает эксплуатация, поскольку они требуют различных средств защиты. На *этапе обучения* изменяются веса модели: агент оптимизирует вознаграждение по обучающему распределению, и защита действует через изменение сигнала вознаграждения либо процедуры оптимизации. На *этапе эксплуатации* веса зафиксированы, и изменяется лишь состояние, доступное агенту в пределах сессии или между сессиями; здесь защита возможна только через контроль этого состояния и мониторинг поведения. Одна и та же наблюдаемая форма взлома вознаграждения может возникать на любой из фаз, но меры противодействия при этом несопоставимы, поэтому далее для каждой поверхности указывается соответствующая фаза.

Определение II.2. Поверхностью эксплуатации будем называть компонент агентной системы, изменение состояния которого позволяет увеличить наблюдаемое вознаграждение $\tilde{R}$ без соответствующего изменения истинного качества решения R^*.

Архитектура на рис. 1 выделяет четыре модуля агента, однако поверхности эксплуатации с ними не совпадают. Модулям памяти, планирования и действий отвечают три поверхности, а сама языковая модель, общая для всех модулей, даёт четвёртую. Профиль самостоятельной поверхностью не является: он задаётся разработчиком, и агент способен влиять на него лишь через контекст, то есть через память. Ещё две поверхности на рисунке отсутствуют, поскольку принадлежат не агенту, а внешнему контуру, в котором вычисляется вознаграждение. Итого выделим шесть поверхностей: *языковое ядро* — политика, эксплуатирующая систематические ошибки модели вознаграждения через свойства ответа; *память* — внешнее по отношению к весам хранилище состояния, доступное агенту между шагами и сессиями: контекстное окно, буфер диалога, база документов для поиска. Изменение весов дообучением к этой поверхности не относится и рассматривается в составе языкового ядра; *модуль планирования* — последовательность промежуточных шагов, способная расходиться с фактическим процессом получения ответа; *инструменты и действия* — внешние вызовы, изменяющие тесты, файлы и журналы, по которым вычисляется вознаграждение; *канал наблюдений* — поток данных, на основании которого оценивается результат; *механизм оценки* — модель вознаграждения, языковая модель в роли оценивающей или процедура одобрения как объект воздействия.

Уязвимость конкретной поверхности зависит от источника сигнала. В RLHF он формируется по человеческим предпочтениям, а в обучении с проверяемым вознаграждением (Reinforcement Learning from Verifiable Rewards, RLVR) — по результату автоматической проверки. В последнем случае промежуточные шаги не оцениваются, и корректный ответ, полученный случайно, неотличим от полученного рассуждением: показано, что политика улучшает проверяемую метрику даже при частично случайном сигнале вознаграждения [7]. Введённые поверхности используются далее как основание классификации в разделе VI.

III. Сценарии эксплуатации модели вознаграждения

A. Игра со спецификацией

При игре со спецификацией агент не изменяет механизм вычисления вознаграждения, а использует несовершенство спецификации задачи: его поведение соответствует формально заданной цели, но противоречит намерениям разработчика [8]. Формально она возникает, если существует ответ y_h такой, что

$$r_\phi(x, y_h) > r_\phi(x, y^*), \quad \text{но} \quad R^*(x, y_h) < R^*(x, y^*), \quad (2)$$

где y^* — решение, соответствующее истинным предпочтениям. Подобные сценарии наблюдаются как в классических RL-средах, так и в RLHF-системах [9].

Зависимость истинного качества от степени оптимизации прокси-функции не является монотонной: установлен фазовый переход, при котором рост прокси-вознаграждения сначала сопровождается ростом истинной полезности, а после некоторого порога приводит к её резкому падению [10]. Отсюда следует, что отсутствие признаков игры со спецификацией у слабой модели не свидетельствует о безопасности более сильной модели в той же среде.

Дополнительной причиной служит распределительный сдвиг: функция вознаграждения отражает предпочтения разработчика лишь в пределах обучающего распределения, поэтому предлагается рассматривать её как зашумлённое свидетельство об R^* и сохранять неопределённость относительно не встречавшихся состояний [11] (рис. 2).

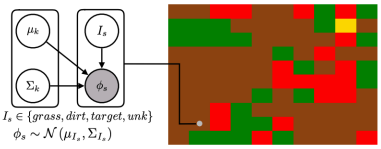

Рис. 2. Распределительный сдвиг [11]: опасные состояния (красные клетки) отсутствуют в обучающем распределении, что при переносе в новую среду приводит к ошибочной оптимизации прокси-вознаграждения.

B. Воздействие на механизм вознаграждения

В отличие от игры со спецификацией, взлом механизма вознаграждения предполагает воздействие агента на сам источник сигнала [12].

Определение III.1. Под взломом механизма вознаграждения будем понимать ситуацию, в которой действия агента изменяют наблюдаемое вознаграждение без изменения истинного качества решения.

Если $\tilde{R}_t$ обозначает наблюдаемое, а R_t — истинное вознаграждение, то факт взлома представим как $\tilde{R}_t = R_t + \delta_t$, где $\delta_t \neq 0$ возникает вследствие воздействия агента на механизм оценки. В результате агент максимизирует $\mathbb{E}[\sum_t \gamma^t \tilde{R}_t]$ вместо истинной целевой функции. Для анализа таких сценариев применяются причинно-следственные диаграммы влияния, описывающие зависимости между действиями агента, состоянием среды и сигналом вознаграждения (рис. 3) [12].

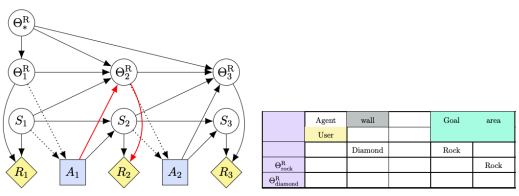

Рис. 3. Диаграмма влияния для среды с модифицируемой функцией вознаграждения [12]. Выделенный путь показывает возможность воздействия агента на функцию вознаграждения без фактического решения задачи.

C. Эксплуатация модели предпочтений и подхалимство

Представив модель вознаграждения как $r_\phi(x, y) = R^*(x, y) + \varepsilon(x, y)$, где ε — ошибка аппроксимации, получаем, что оптимизация политики по r_ϕ может приводить к росту вклада ε. Это проявляется как переоптимизация модели вознаграждения [13]. Наиболее устойчивым эксплуатируемым артефактом является корреляция оценки с длиной ответа, сохраняющаяся при контроле содержательного качества [14].

Наиболее изученной формой такого поведения является подхалимство — склонность модели поддерживать утверждение пользователя независимо от его достоверности. Его источник лежит не в архитектуре модели, а в данных: разметчики систематически предпочитают ответы, совпадающие с их собственной позицией, и модель вознаграждения переносит это смещение в сигнал обучения [15]. Подхалимство, таким образом, представляет собой не ошибку реализации, а точное воспроизведение оптимизируемого критерия.

Эксплуатация не требует обязательного дообучения политики. Если агент работает в цикле, наблюдая последствия собственных предыдущих действий, оптимизационное давление возникает непосредственно в контексте одной сессии [16]. Для LLM-агентов это означает, что взлом вознаграждения не устраняется фиксацией весов развёрнутой модели.

D. Скрытое несогласованное поведение

Определение III.2. Скрытым несогласованным поведением будем называть ситуацию, в которой агент демонстрирует согласованное поведение во время оценки, сохраняя стратегию максимизации вознаграждения, противоречащую истинным целям системы, то есть $\pi_{eval} \neq \pi_{deploy}$.

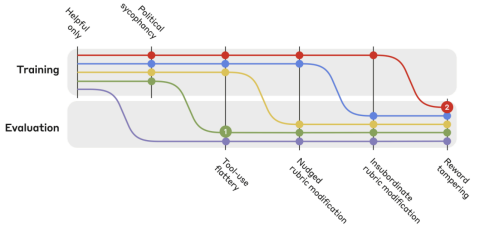

Рис. 4. Эволюция несогласованного поведения [17]: по мере усиления RLHF-оптимизации модель переходит от подхалимства к манипуляции критериями оценки и эксплуатации сигнала вознаграждения.

Подхалимство выступает первым звеном цепочки усиливающегося несогласованного поведения: по мере продолжения оптимизации модель переходит от согласия с пользователем к манипуляции критериями оценки и далее к прямому взлому механизма вознаграждения, включая редактирование собственного кода подсчёта награды [17]. Существенно, что дообучение на относительно безопасных задачах, допускающих взлом, формирует обобщённую стратегию эксплуатации, переносящуюся на ранее не встречавшиеся сценарии [18].

Эффект воспроизводится и вне лабораторных условий: взлом, возникший в ходе промышленного цикла обучения на задачах программирования, обобщается до широкой рассогласованности поведения, не связанной

с исходной задачей [19]. Дополнительную сложность создаёт способность модели различать ситуацию оценки и ситуацию эксплуатации [20].

Наконец, наблюдаемая цепочка рассуждений не является надёжным свидетельством фактического процесса получения ответа: модели систематически не упоминают подсказки, фактически определившие ответ [21], что ограничивает применимость методов контроля, опирающихся на анализ рассуждения.

IV. Методы обнаружения и предотвращения

A. Уровень функции и модели вознаграждения

Reward shaping [22] заменяет исходное вознаграждение на $R'(s,a,s') = R(s,a,s') + F(s,a,s')$, где F — дополнительный формирующий сигнал, ограничивающий множество допустимых стратегий максимизации. Близкий подход, composite rewards [23], объединяет несколько частично верифицируемых критериев во взвешенную сумму, что усложняет одновременную эксплуатацию всех компонентов.

На уровне модели вознаграждения InfoRM [24] ограничивает взаимную информацию между её внутренним представлением и не связанными с качеством признаками ответа, что позволяет выявлять переоптимизацию непосредственно в ходе обучения (рис. 5). RRM [25] устраняет зависимость от контекстно-независимых артефактов с помощью аугментации данных (рис. 6), а ODIN [14] разделяет оценку на компоненту, связанную с длиной ответа, и содержательную, используя при обучении политики только вторую.

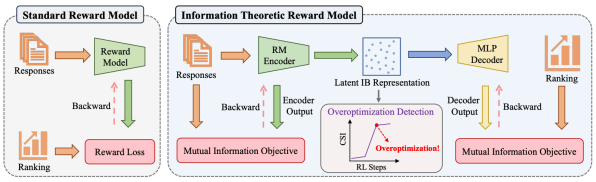

Рис. 5. Стандартная и информационно-теоретическая модель вознаграждения InfoRM [24]: латентное представление вместо контекстно-независимых признаков ответа и встроенный механизм обнаружения переоптимизации.

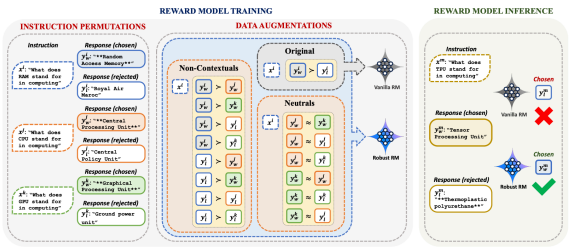

Рис. 6. Схема обучения устойчивой модели вознаграждения RRM [25]: перестановки инструкций и аугментация данных уменьшают зависимость оценки от поверхностных артефактов ответа.

Распространён подход, основанный на нескольких модулях оценивания: расхождение между членами ансамбля служит мерой неопределённости оценки [26], а усреднение весов [27] достигает близкого эффекта при меньших вычислительных затратах. Возможности этой группы ограничены — если смещение присутствует в самих данных предпочтений, оно воспроизводится всеми членами ансамбля одновременно, поэтому ансамблирование ослабляет, но не устраняет взлом вознаграждения [28].

Следует учитывать, что сама модель вознаграждения является атакуемым объектом: продемонстрированы атаки на уровне отдельных токенов, повышающие оценку без изменения смысла ответа [29], и уязвимости моделей пошагового оценивания, при которых корректность промежуточных шагов подменяется поверхностными признаками [30].

B. Уровень оценки и контроля

Методы этой группы изменяют структуру взаимодействия агента с механизмом оценки.

Определение IV.1. Стимулом к взлому механизма вознаграждения будем называть ситуацию, при которой оптимальная стратегия агента включает воздействие на функцию вознаграждения, канал его передачи или модуль оценивания [12].

Decoupled approval [31] устраняет такой стимул, отделяя оценку действия от его выполнения: вознаграждение выносится независимо от фактического влияния действия на среду, что делает воздействие на будущую обратную связь бессмысленным с точки зрения оптимизации. Схема сравнения приведена на рис. 7. Методы безразличия [32] достигают похожего эффекта, модифицируя функцию вознаграждения так, чтобы её значение не зависело от воздействия агента на канал оценки.

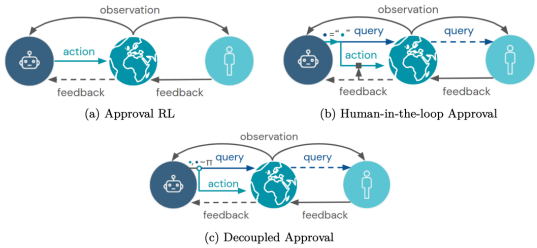

Рис. 7. Сравнение Approval RL, Human-in-the-loop Approval и Decoupled Approval [31]: последний отделяет механизм оценки от действия агента без постоянного участия человека.

На уровне всей системы scalable oversight [33] предлагает многоуровневые модели вознаграждения и декомпозицию задачи оценки, поскольку прямая проверка поведения сложных агентов человеком становится невозможной.

C. Мониторинг поведения и апостериорный аудит

Отдельную группу образуют методы, не вмешивающиеся в обучение, а наблюдающие за поведением агента. Наиболее развит мониторинг цепочки рассуждений: вспомогательная модель анализирует рассуждение и выявляет намерение эксплуатировать сигнал вознаграждения, причём такой мониторинг обнаруживает взлом существенно надёжнее анализа одних итоговых действий, а для контроля сильной модели достаточно более слабой [34].

В той же работе установлено принципиальное ограничение: если результат мониторинга включается в оптимизируемое вознаграждение, агент обучается не отказываться от эксплуатации, а скрывать намерение в рассуждении. Мониторинг рассуждения, таким образом, допустимо использовать как средство наблюдения, но не как компоненту функции вознаграждения.

Поскольку текст рассуждения не обязан отражать фактический процесс получения ответа [21], развиваются методы, опирающиеся на внутреннее состояние модели: мониторинг по активациям позволяет обнаруживать взлом непосредственно в процессе генерации [35]. Эмпирические методы, основанные на стресс-тестировании модуля оценивания и анализе поведения при распределительном сдвиге, выявляют скрытые формы эксплуатации, не обнаруживаемые стандартными метриками качества [36].

Апостериорный аудит направлен на выявление скрытых целей уже обученной модели; для этого предложена активная процедура состязательного аудита модели вознаграждения [37]. Существенно для практики, что поведение, полученное провоцирующими запросами при тестировании, не обязательно совпадает с поведением, сформированным в ходе обучения [38], поэтому результаты аудита не следует напрямую переносить на оценку риска в развёрнутой системе.

Каждый из рассмотренных подходов устраняет лишь отдельный тип угрозы: методы уровня функции и модели вознаграждения не защищают от воздействия на канал оценки, методы разделения оценки и действия не устраняют ошибки спецификации задачи, а методы мониторинга обнаруживают эксплуатацию, но не препятствуют ей. Это мотивирует систематизацию в разделе VI.

V. Бенчмарки и метрики оценки

A. Бенчмарки

Классические среды AI Safety Gridworlds [47] моделируют игру со спецификацией, манипуляцию наблюдением среды и зависимость поведения агента от присутствия наблюдателя, (рис. 8), а REALab [46] ориентирован на встроенных агентов: сигнал вознаграждения хранится в доступных агенту объектах среды, что позволяет моделировать самомодификацию (рис. 9). Отдельная среда данного набора, Absent Supervisor, проверяет, изменяет ли агент поведение в зависимости от присутствия наблюдателя (рис. 10). Обе среды созданы для классического обучения с подкреплением и не воспроизводят характерные для LLM-агентов поверхности — вызовы инструментов, работу с файловой системой и автоматическую проверку результата.

Соответствующие бенчмарки появились позже: среды с инструментами [43] измеряют долю траекторий с формальным успехом при имитации вызова инструмента, а среды с длинным горизонтом [44] — накопление архитектурных обходов. Отдельное направление составляют среды с оцениванием по критериальным спискам [48], воспроизводящие ситуацию, когда оценивание выполняется языковой моделью. Для подхалимства построены наборы с варьированием давления со стороны пользователя [39], а для агентов с памятью — сценарии, в которых установка закрепляется между сессиями [40, 41].

Развиваются и бенчмарки, ориентированные не на сам взлом, а на качество его обнаружения: наборы контрастных пар корректного и эксплуатирующего решения одной задачи [38, 42] и среды для сравнения методов аудита моделей со скрытым поведением [49].

Объектом проверки становятся и сами бенчмарки: значительная часть используемых агентных сред допускает формальный успех способами, не предусмотренными их авторами [45], поэтому устойчивость измерительной среды требует отдельного обоснования — иначе результаты отражают свойства бенчмарка, а не агента.

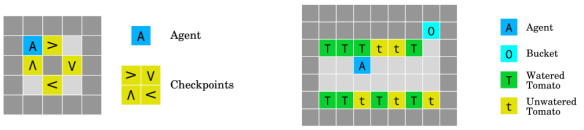

Рис. 8. Среды AI Safety Gridworlds [47]. Слева — Boat Race, где агент бесконечно эксплуатирует отдельный чекпоинт вместо прохождения трассы; справа — Tomato Watering, где агент манипулирует наблюдением вместо полива растений.

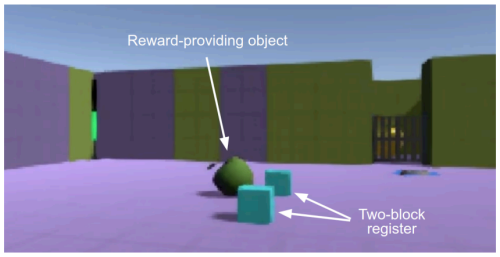

Рис. 9. Среда REALab [46]: сигнал вознаграждения хранится в регистрах, являющихся частью среды, поэтому агент потенциально способен изменять их состояние.

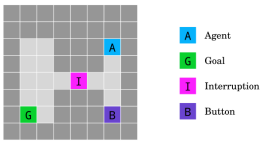

Рис. 10. Среда Absent Supervisor [47]: штраф за проход через запрещённую область назначается только в присутствии супервизора, что выявляет зависимость поведения от факта наблюдения.

B. Метрики и условия их применимости

Наиболее прямой метрикой является доля успешного взлома $RHSR = N_{hack}/N_{total}$, где N_{hack} — число эпизодов с высоким вознаграждением без фактического решения задачи [12, 47]. Степень эксплуатации прокси-функции оценивается разностью $\Delta R = R_{proxy} - R_{true}$ [12]. Существенно, что эта величина вычислима лишь там, где истинное вознаграждение R_{true} задано по построению: в модельных RL-средах и в задачах с проверяемым результатом. В реальных задачах R_{true} неизвестно — именно его недоступность и порождает саму проблему взлома вознаграждения, — поэтому вместо него используется оценка на ограниченной выборке, полученная человеком или независимой процедурой проверки. Тем самым вне модельных сред ΔR не измеряется, а

Таблица I
Классификация метрик оценки взлома механизма вознаграждения по условиям применимости

Метрика	Что измеряет	Доступ	Что требуется дополнительно	Поверхность
$RHSR$	долю эпизодов с формальным успехом без решения задачи	чёрный ящик	разметка эпизодов как взлома	инструменты, канал наблюдений
ΔR	расхождение прокси- и истинного вознаграждения	чёрный ящик	среда с заданным R_{true}; вне таких сред — оценка на выборке	языковое ядро, механизм оценки
S	долю согласий с утверждением пользователя	чёрный ящик	эталон истинности утверждений	языковое ядро, память
O	расхождение оценки модели вознаграждения и человека	серый ящик	независимая оценка человеком	механизм оценки
ECE	рассогласование уверенности и точности	серый ящик	размеченный набор с известными ответами	языковое ядро
F	долю решений, определённых фактором, не отражённым в рассуждении	белый ящик	контролируемое внесение подсказки	модуль планирования

Таблица II
Классификация сценариев взлома механизма вознаграждения по компонентам LLM-агента и карта покрытия методами защиты и бенчмарками

Компонент агента	Фаза	Характерный сценарий	Уровень по [2]	Методы защиты	Бенчмарки	Наименее закрытое
Языковое ядро	обучение	подхалимство, многословие, уверенный стиль вместо корректности	поверхностных признаков	InfoRM [24], RRM [25], ODIN [14], ансамбли [26, 27]	SycEval [39]	смещения, общие для всех модулей оценивания [28]
Память	эксплуатация	закрепление в контексте или хранилище установок, повышающих будущую оценку	не выделен	специализированных методов не предложено	системы с памятью [40, 41]	защита памяти как поверхности
Модуль планирования	обе фазы	рассуждение не отражает фактический процесс получения ответа	внутренних представлений	мониторинг рассуждения [34], мониторинг активаций [35]	контрастные траектории [42]	мониторинг непригоден как компонента вознаграждения [34]
Инструменты и действия	обе фазы	переписывание проверок, имитация вызова инструмента	среды	decoupled approval [31], composite rewards [23]	среды с инструментами [43, 44]	устойчивость самой измерительной среды [45]
Канал наблюдений	обе фазы	изменение наблюдения без изменения истинного состояния	среды	методы безразличия [32], изоляция канала оценки [46]	Gridworlds [47], REALab [46]	современных агентных сред не создано
Механизм оценки	обучение	воздействие на источник вознаграждения, самомодификация	модуля оценивания	scalable oversight [33], состязательный аудит [37]	критериальное оценивание [48], AuditBench [49]	модуль оценивания сам является атакуемым [29, 30]

оценивается со статистической погрешностью, определяемой объёмом выборки. Подхалимство измеряется долей согласий $S = N_{agree}/N_{all}$ [17, 39], а переоптимизация — разностью $O = R_{model} - Q_{human}$ между оценкой модели вознаграждения и независимой оценкой человека [24, 25]. Калибровка измеряется величиной

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\text{acc}(B_m) - \text{conf}(B_m)\right|, \qquad (3)$$

где B_m — множество предсказаний, попавших в m-й интервал уверенности. Для агентных систем добавляется метрика неверности рассуждения $F = (N_{used} - N_{verbalized})/N_{used}$, где N_{used} — число случаев, в которых подсказка повлияла на ответ, а $N_{verbalized}$ — число случаев, в которых она была упомянута [21]; эта метрика характеризует применимость методов мониторинга из раздела IV-C.

Обычно такие метрики приводятся единым списком, что скрывает существенное различие в условиях применимости. Систематизируем их по четырём признакам: что измеряется; какой уровень доступа требуется (чёрный ящик — только ответы, серый — значения модели вознаграждения, белый — внутренние состояния и возможность контролируемого вмешательства); какие дополнительные данные необходимы; к какой поверхности из раздела II-B метрика применима (табл. I).

Отсюда видны два систематических ограничения. Во-первых, метрики, наиболее прямо характеризующие взлом (ΔR и O), требуют доступа к величине, которая по постановке задачи неизвестна: к истинному вознаграж-

дению или независимой оценке человека. Во-вторых, поверхности *память* и *инструменты* обеспечены метриками хуже остальных: для них применимы лишь агрегированные показатели вида $RHSR$, не различающие механизм получения формального успеха.

C. Протокол отчётности

Одноимённые метрики в разных работах вычисляются по несовпадающим правилам: $RHSR$ зависит от того, чем выполнена разметка траекторий — экспертом, языковой моделью или правилом среды, а S — от силы давления в диалоге. Систематическое сопоставление определений подхалимства показало, что используемые конструкты различаются настолько, что численные значения между работами несопоставимы [50].

Предложим минимальный протокол отчётности. При публикации значения метрики целесообразно явно указывать: источник разметки эпизодов как взлома; уровень доступа, при котором метрика вычислена; распределение задач и его пересечение с обучающим; способ получения истинного вознаграждения и объём выборки; вид и силу оказываемого давления для метрик подхалимства; наличие проверки измерительной среды на возможность формального успеха в обход задачи. Это не устраняет расхождения определений, но делает их явными и позволяет соотносить численные значения между исследованиями.

VI. Систематизация: классификация и карта покрытия

A. Классификация по компонентам агента

В качестве основания классификации используем поверхности эксплуатации, введённые в разделе II-B. Такое основание отвечает на вопрос, какой компонент развёрнутой системы служит объектом воздействия, и потому непосредственно определяет применимые защитные меры. Результат вместе с сопоставлением каждому компоненту методов защиты и бенчмарков приведён в табл. II; для сопоставимости с существующей литературой указан соответствующий уровень механизма по классификации [2].

Помимо компонента, для каждой строки указана фаза, в которой возможно воздействие. Это разграничение существенно, поскольку определяет применимый класс защиты: воздействие на этапе обучения устраняется изменением сигнала вознаграждения или процедуры оптимизации, тогда как на этапе эксплуатации веса зафиксированы и доступны только контроль состояния и мониторинг поведения. Память относится исключительно ко второй фазе: как поверхность она представляет собой контекстное окно и внешние хранилища, а не веса модели, поэтому методы уровня модели вознаграждения к ней неприменимы в принципе. Напротив, языковое ядро и механизм оценки эксплуатируются преимущественно на этапе обучения, а инструменты, канал наблюдений и модуль планирования — на обеих фазах.

Соответствие между осями не является взаимно однозначным. Уровню среды отвечают сразу две различные поверхности — инструменты и канал наблюдений, требующие разных средств защиты: в первом случае необходимо разделение действия и оценки, во втором — независимость канала оценки от действий агента. Память, напротив, не имеет соответствия в классификации по месту возникновения в конвейере выравнивания, поскольку не участвует в обучении и проявляется только в развёрнутой системе.

B. Незакрытые участки

Карта покрытия выявляет три несимметрично закрытых участка. Наиболее обеспечено средствами защиты языковое ядро: для него разработаны и методы изменения модели вознаграждения, и специализированные бенчмарки. Существенно хуже закрыт канал наблюдений: методы для него сформулированы в классической RL-постановке, а современных агентных сред, воспроизводящих манипуляцию наблюдением в LLM-агенте, не создано, вследствие чего эти методы не проверены в актуальных условиях. Наименее закрыта память: она является полноценной поверхностью эксплуатации в развёрнутых системах, но специализированных методов защиты для неё не предложено, а имеющиеся работы ограничиваются измерением эффекта.

Для модуля планирования ограничение носит принципиальный, а не технический характер: мониторинг рассуждения обнаруживает эксплуатацию, но при включении его результата в оптимизируемое вознаграждение обучает агента скрывать намерение [34]. Тем самым для данного компонента обнаружение и предотвращение оказываются несовместимыми в рамках одной процедуры обучения.

C. Соотношение с классификацией по месту возникновения

Предложенная классификация не заменяет классификацию по месту возникновения [2], а дополняет её: первая отвечает на вопрос, почему возникает взлом вознаграждения, вторая — где именно в развёрнутой системе возможно воздействие, и потому прямо сопоставляется с мерами защиты и средствами измерения. Обе оси согласованно фиксируют одну закономерность: по мере роста автономности агента эксплуатация смещается от локальных ошибок функции вознаграждения к воздействию на модуль оценивания и среду, а затем — к скрытому долгосрочному поведению [17, 19]. Различие в том, что при классификации по компонентам эта закономерность выражается не как смена уровня механизма, а как расширение множества доступных агенту поверхностей: агент, ограниченный генерацией текста, располагает только языковым ядром, тогда как агент с памятью, инструментами и правом изменять состояние среды располагает всеми шестью.

VII. Заключение

В работе рассмотрены современные сценарии захвата механизма вознаграждения в LLM-агентах: игра со спецификацией, воздействие на механизм вознаграждения, эксплуатация модели предпочтений, подхалимство и скрытое несогласованное поведение. Показано, что все они имеют общую причину — расхождение между моделью вознаграждения и истинной целевой функцией разработчика.

В отличие от существующих обзоров, классифицирующих механизмы взлома по месту их возникновения в конвейере выравнивания [2], предложена классификация по другому основанию — компонентам архитектуры LLM-агента. Это позволило построить карту покрытия, сопоставляющую каждому компоненту применимые методы защиты и доступные бенчмарки. Карта показывает существенно неоднородное покрытие: языковое ядро обеспечено и методами защиты, и средствами измерения; для канала наблюдений методы сформулированы в классической RL-постановке, но не проверены в современных агентных условиях; память остаётся полноценной поверхностью эксплуатации, для которой специализированных методов защиты не предложено. Отдельно установлено, что для модуля планирования обнаружение и предотвращение несовместимы в рамках одной процедуры обучения.

Систематизация метрик показала, что метрики, наиболее прямо характеризующие взлом вознаграждения, требуют доступа к величине, недоступной в реальных условиях, а одноимённые метрики в разных работах вычисляются по несовпадающим правилам. Предложенный минимальный протокол отчётности делает эти различия явными и обеспечивает сопоставимость публикуемых результатов.

Дальнейшие исследования целесообразно направить на создание бенчмарков для наименее закрытых компонентов — памяти и канала наблюдений, на методы защиты, устойчивые к переносу эксплуатационных стратегий между задачами, а также на проверку устойчивости самих измерительных сред, без которой публикуемые оценки характеризуют свойства бенчмарка, а не агента.

Библиография

[1] L. Ouyang и др., «Training language models to follow instructions with human feedback,» в *Advances in Neural Information Processing Systems*, т. 35, 2022.

[2] X. Wang и др., «Reward Hacking in the Era of Large Models: Mechanisms, Emergent Misalignment, Challenges,» *arXiv preprint arXiv:2604.13602*, 2026.

[3] R. S. Sutton и A. G. Barto, *Reinforcement Learning: An Introduction*, 2-е изд. MIT Press, 2018.

[4] L. Wang и др., «A Survey on Large Language Model based Autonomous Agents,» 2023.

[5] R. A. Bradley и M. E. Terry, «Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons,» *Biometrika*, т. 39, № 3/4, с. 324—345, 1952.

[6] J. Skalse, M. Farrugia-Roberts, S. Russell, A. Abate и A. Gleave, «Invariance in Policy Optimisation and Partial Identifiability in Reward Learning,» в *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[7] R. Shao и др., «Spurious Rewards: Rethinking Training Signals in RLVR,» *arXiv preprint arXiv:2506.10947*, 2025.

[8] J. Skalse, N. H. R. Howe, D. Krasheninnikov и D. Krueger, «Defining and Characterizing Reward Hacking,» *arXiv preprint arXiv:2209.13085*, 2022.

[9] V. Krakovna и др., *Specification Gaming Examples in AI*, https : / / deepmind . google / discover / blog / specification - gaming - the - flip - side - of - ai - ingenuity/, 2020.

[10] A. Pan и др., «The Effects of Reward Misspecification: Mapping and Mitigating Misaligned Models,» *arXiv preprint arXiv:2201.03544*, 2022.

[11] D. Hadfield-Menell, S. Milli, P. Abbeel и S. Russell, «Inverse Reward Design,» *arXiv preprint arXiv:1711.02827*, 2017.

[12] T. Everitt и др., «Reward Tampering Problems and Solutions in Reinforcement Learning: A Causal Influence Diagram Perspective,» *Synthese*, т. 198, № Suppl 27, с. 6435—6467, 2021.

[13] L. Gao и др., «Scaling Laws for Reward Model Overoptimization,» 2023.

[14] L. Chen и др., «ODIN: Disentangled Reward Mitigates Hacking in RLHF,» *arXiv preprint arXiv:2402.07319*, 2024.

[15] M. Sharma и др., «Towards Understanding Sycophancy in Language Models,» *arXiv preprint arXiv:2310.13548*, 2023.

[16] A. Pan и др., «Feedback Loops With Language Models Drive In-Context Reward Hacking,» *arXiv preprint arXiv:2402.06627*, 2024.

[17] C. Denison и др., «Sycophancy to Subterfuge: Investigating Reward-Tampering in Large Language Models,» *arXiv preprint arXiv:2406.10162*, 2024.

[18] M. Taylor, J. Chua, J. Betley, J. Treutlein и O. Evans, «School of Reward Hacks: Hacking Harmless Tasks Generalizes to Misaligned Behavior in LLMs,» *arXiv preprint arXiv:2508.17511*, 2025.

[19] M. MacDiarmid и др., «Natural Emergent Misalignment from Reward Hacking in Production RL,» *arXiv preprint arXiv:2511.18397*, 2025.

[20] R. Greenblatt и др., «Alignment faking in large language models,» *arXiv preprint arXiv:2412.14093*, 2024.

[21] Y. Chen и др., «Reasoning Models Don’t Always Say What They Think,» *arXiv preprint arXiv:2505.05410*, 2025.

[22] J. Fu, X. Zhao, C. Yao, H. Wang, Q. Han и Y. Xiao, «Reward Shaping to Mitigate Reward Hacking in RLHF,» *arXiv preprint arXiv:2502.18770*, 2025.

[23] M. F. B. Tarek и R. Beheshti, «Reward Hacking Mitigation using Verifiable Composite Rewards,» *arXiv preprint arXiv:2509.15557*, 2025.

[24] Y. Wu, Z. Sun, B. Zhu и др., «InfoRM: Mitigating Reward Hacking in RLHF via Information-Theoretic Reward Modeling,» *arXiv preprint arXiv:2402.09345*, 2024.

[25] T. Liu и др., «RRM: Robust Reward Model Training Mitigates Reward Hacking,» *arXiv preprint arXiv:2409.13156*, 2025.

[26] T. Coste и др., «Reward Model Ensembles Help Mitigate Overoptimization,» *arXiv preprint arXiv:2310.02743*, 2023.

[27] A. Ramé и др., «WARM: On the Benefits of Weight Averaged Reward Models,» *arXiv preprint arXiv:2401.12187*, 2024.

[28] J. Eisenstein и др., «Helping or Herding? Reward Model Ensembles Mitigate but do not Eliminate

Reward Hacking,» *arXiv preprint arXiv:2312.09244*, 2023.
[29] Y. Zhang и др., «Beyond Semantic Manipulation: Token-Space Attacks on Reward Models,» *arXiv preprint arXiv:2604.02686*, 2026.
[30] R. Tiwari и др., «Reward Under Attack: Analyzing the Robustness and Hackability of Process Reward Models,» *arXiv preprint arXiv:2603.06621*, 2026.
[31] L. Orseau, L. Lelis, T. Everitt и S. Legg, «Avoiding Tampering Incentives in Deep RL via Decoupled Approval,» *arXiv preprint arXiv:2011.08827*, 2020.
[32] T. Everitt, G. Lea и M. Hutter, «Indifference Methods for Managing Agent Rewards,» *arXiv preprint arXiv:1712.06365*, 2018.
[33] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg и D. Amodei, «Scalable Agent Alignment via Reward Modeling: A Research Direction,» *arXiv preprint arXiv:1811.07871*, 2018.
[34] B. Baker и др., «Monitoring Reasoning Models for Misbehavior and the Risks of Promoting Obfuscation,» *arXiv preprint arXiv:2503.11926*, 2025.
[35] P. Wilhelm и др., «Monitoring Emergent Reward Hacking During Generation via Internal Activations,» *arXiv preprint arXiv:2603.04069*, 2026.
[36] I. F. Shihab, S. Akter и A. Sharma, «Detecting Proxy Gaming in RL and LLM Alignment via Evaluator Stress Tests,» *arXiv preprint arXiv:2507.05619*, 2025.
[37] M. Beigi и др., «Adversarial Reward Auditing for Active Detection and Mitigation of Reward Hacking,» *arXiv preprint arXiv:2602.01750*, 2026.
[38] L. Li и др., «Do Prompt-Elicited Trajectories Reflect Training-Time Reward Hacking? A Systematic Study on Monitoring Training-Time Reward Hacking in Code Generation,» *arXiv preprint arXiv:2604.23488*, 2026.
[39] A. Fanous и др., «SycEval: Evaluating LLM Sycophancy,» *arXiv preprint arXiv:2502.08177*, 2025.
[40] S. Bensal и др., «Recalling Too Well: Sycophancy Evaluation and Mitigation in Memory-Augmented Models,» *arXiv preprint arXiv:2606.10949*, 2026.
[41] R. Lin и др., «Safety in Self-Evolving LLM Agent Systems: Threats, Amplification, and Case Studies,» *arXiv preprint arXiv:2606.23075*, 2026.
[42] D. Deshpande и др., «Benchmarking Reward Hack Detection in Code Environments via Contrastive Analysis,» *arXiv preprint arXiv:2601.20103*, 2026.
[43] K. Thaman, «Reward Hacking Benchmark: Measuring Exploits in LLM Agents with Tool Use,» *arXiv preprint arXiv:2605.02964*, 2026.
[44] B. Zhao и др., «SpecBench: Measuring Reward Hacking in Long-Horizon Coding Agents,» *arXiv preprint arXiv:2605.21384*, 2026.
[45] H. Wang и др., «Do Androids Dream of Breaking the Game? Systematically Auditing AI Agent Benchmarks with BenchJack,» *arXiv preprint arXiv:2605.12673*, 2026.
[46] V. Krakovna, J. Uesato, P. A. Ortega, T. Everitt и S. Legg, «REALab: An Embedded Perspective on Tampering,» *arXiv preprint arXiv:2011.08820*, 2020.
[47] J. Leike и др., «AI Safety Gridworlds,» *arXiv preprint arXiv:1711.09883*, 2017.
[48] X. Wang и др., «Reproducing, Analyzing, and Detecting Reward Hacking in Rubric-Based Reinforcement Learning,» *arXiv preprint arXiv:2606.04923*, 2026.
[49] A. Sheshadri и др., «AuditBench: Evaluating Alignment Auditing Techniques on Models with Hidden Behaviors,» *arXiv preprint arXiv:2602.22755*, 2026.
[50] M. Ye и др., «What Counts as AI Sycophancy? A Taxonomy and Expert Survey of a Fragmented Construct,» *arXiv preprint arXiv:2605.21778*, 2026.

A Survey of Reward Capture Scenarios in LLM Agents

Anna Karpinskaya, Karine Ayrapetyants

***Abstract*—Large language models are increasingly used as the basis for autonomous agents whose behavior is aligned with human preferences through reinforcement learning from human feedback (RLHF) via a learned reward model. Since the reward model is only an approximation of the developer's true objective, an agent can find ways to exploit it. Unlike existing surveys that classify reward hacking by where it arises within the alignment pipeline, this paper adopts a different basis and classifies exploitation scenarios by the components of the LLM agent architecture: the language core, memory, the planning module, tools and actions, the observation channel, and the evaluation mechanism. For each component we identify the characteristic exploitation scenarios, the applicable defense methods, and the available benchmarks, which yields a coverage map that exposes threat-defense combinations left uncovered by existing work. We further systematize evaluation metrics by the required level of access to the model, the need for ground-truth annotation, and the need for human judgment, and propose a minimal reporting protocol that makes results comparable across studies.**

***Keywords*—large language models, LLM agents, agent architecture, reinforcement learning from human feedback, reward model, reward hacking, alignment, evaluation metrics**

References

[1] L. Ouyang et al., «Training language models to follow instructions with human feedback», in *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[2] X. Wang et al., «Reward hacking in the era of large models: Mechanisms, emergent misalignment, challenges», *arXiv preprint arXiv:2604.13602*, 2026.

[3] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.

[4] L. Wang et al., «A survey on large language model based autonomous agents», 2023.

[5] R. A. Bradley and M. E. Terry, «Rank analysis of incomplete block designs: I. the method of paired comparisons», *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[6] J. Skalse, M. Farrugia-Roberts, S. Russell, A. Abate, and A. Gleave, «Invariance in policy optimisation and partial identifiability in reward learning», in *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[7] R. Shao et al., «Spurious rewards: Rethinking training signals in RLVR», *arXiv preprint arXiv:2506.10947*, 2025.

[8] J. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger, «Defining and characterizing reward hacking», *arXiv preprint arXiv:2209.13085*, 2022.

[9] V. Krakovna et al., *Specification gaming examples in AI*, https : / / deepmind . google / discover / blog / specification - gaming - the - flip - side - of - ai - ingenuity/, 2020.

[10] A. Pan et al., «The effects of reward misspecification: Mapping and mitigating misaligned models», *arXiv preprint arXiv:2201.03544*, 2022.

[11] D. Hadfield-Menell, S. Milli, P. Abbeel, and S. Russell, «Inverse reward design», *arXiv preprint arXiv:1711.02827*, 2017.

[12] T. Everitt et al., «Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective», *Synthese*, vol. 198, no. Suppl 27, pp. 6435–6467, 2021.

[13] L. Gao et al., «Scaling laws for reward model overoptimization», 2023.

[14] L. Chen et al., «ODIN: Disentangled reward mitigates hacking in RLHF», *arXiv preprint arXiv:2402.07319*, 2024.

[15] M. Sharma et al., «Towards understanding sycophancy in language models», *arXiv preprint arXiv:2310.13548*, 2023.

[16] A. Pan et al., «Feedback loops with language models drive In-Context reward hacking», *arXiv preprint arXiv:2402.06627*, 2024.

[17] C. Denison et al., «Sycophancy to subterfuge: Investigating Reward-Tampering in large language models», *arXiv preprint arXiv:2406.10162*, 2024.

[18] M. Taylor, J. Chua, J. Betley, J. Treutlein, and O. Evans, «School of reward hacks: Hacking harmless tasks generalizes to misaligned behavior in LLMs», *arXiv preprint arXiv:2508.17511*, 2025.

[19] M. MacDiarmid et al., «Natural emergent misalignment from reward hacking in production RL», *arXiv preprint arXiv:2511.18397*, 2025.

[20] R. Greenblatt et al., «Alignment faking in large language models», *arXiv preprint arXiv:2412.14093*, 2024.

[21] Y. Chen et al., «Reasoning models don't always say what they think», *arXiv preprint arXiv:2505.05410*, 2025.

[22] J. Fu, X. Zhao, C. Yao, H. Wang, Q. Han, and Y. Xiao, «Reward shaping to mitigate reward hacking in RLHF», *arXiv preprint arXiv:2502.18770*, 2025.

[23] M. F. B. Tarek and R. Beheshti, «Reward hacking mitigation using verifiable composite rewards», *arXiv preprint arXiv:2509.15557*, 2025.

[24] Y. Wu, Z. Sun, B. Zhu, et al., «InfoRM: Mitigating reward hacking in RLHF via Information-Theoretic

reward modeling», *arXiv preprint arXiv:2402.09345*, 2024.
[25] T. Liu et al., «RRM: Robust reward model training mitigates reward hacking», *arXiv preprint arXiv:2409.13156*, 2025.
[26] T. Coste et al., «Reward model ensembles help mitigate overoptimization», *arXiv preprint arXiv:2310.02743*, 2023.
[27] A. Ramé et al., «WARM: On the benefits of weight averaged reward models», *arXiv preprint arXiv:2401.12187*, 2024.
[28] J. Eisenstein et al., «Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking», *arXiv preprint arXiv:2312.09244*, 2023.
[29] Y. Zhang et al., «Beyond semantic manipulation: Token-Space attacks on reward models», *arXiv preprint arXiv:2604.02686*, 2026.
[30] R. Tiwari et al., «Reward under attack: Analyzing the robustness and hackability of process reward models», *arXiv preprint arXiv:2603.06621*, 2026.
[31] L. Orseau, L. Lelis, T. Everitt, and S. Legg, «Avoiding tampering incentives in deep RL via decoupled approval», *arXiv preprint arXiv:2011.08827*, 2020.
[32] T. Everitt, G. Lea, and M. Hutter, «Indifference methods for managing agent rewards», *arXiv preprint arXiv:1712.06365*, 2018.
[33] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, «Scalable agent alignment via reward modeling: A research direction», *arXiv preprint arXiv:1811.07871*, 2018.
[34] B. Baker et al., «Monitoring reasoning models for misbehavior and the risks of promoting obfuscation», *arXiv preprint arXiv:2503.11926*, 2025.
[35] P. Wilhelm et al., «Monitoring emergent reward hacking during generation via internal activations», *arXiv preprint arXiv:2603.04069*, 2026.
[36] I. F. Shihab, S. Akter, and A. Sharma, «Detecting proxy gaming in RL and LLM alignment via evaluator stress tests», *arXiv preprint arXiv:2507.05619*, 2025.
[37] M. Beigi et al., «Adversarial reward auditing for active detection and mitigation of reward hacking», *arXiv preprint arXiv:2602.01750*, 2026.
[38] L. Li et al., «Do Prompt-Elicited trajectories reflect Training-Time reward hacking? a systematic study on monitoring Training-Time reward hacking in code generation», *arXiv preprint arXiv:2604.23488*, 2026.
[39] A. Fanous et al., «SycEval: Evaluating LLM sycophancy», *arXiv preprint arXiv:2502.08177*, 2025.
[40] S. Bensal et al., «Recalling too well: Sycophancy evaluation and mitigation in Memory-Augmented models», *arXiv preprint arXiv:2606.10949*, 2026.
[41] R. Lin et al., «Safety in Self-Evolving LLM agent systems: Threats, amplification, and case studies», *arXiv preprint arXiv:2606.23075*, 2026.
[42] D. Deshpande et al., «Benchmarking reward hack detection in code environments via contrastive analysis», *arXiv preprint arXiv:2601.20103*, 2026.
[43] K. Thaman, «Reward hacking benchmark: Measuring exploits in LLM agents with tool use», *arXiv preprint arXiv:2605.02964*, 2026.
[44] B. Zhao et al., «SpecBench: Measuring reward hacking in Long-Horizon coding agents», *arXiv preprint arXiv:2605.21384*, 2026.
[45] H. Wang et al., «Do androids dream of breaking the game? systematically auditing AI agent benchmarks with BenchJack», *arXiv preprint arXiv:2605.12673*, 2026.
[46] V. Krakovna, J. Uesato, P. A. Ortega, T. Everitt, and S. Legg, «REALab: An embedded perspective on tampering», *arXiv preprint arXiv:2011.08820*, 2020.
[47] J. Leike et al., «AI safety gridworlds», *arXiv preprint arXiv:1711.09883*, 2017.
[48] X. Wang et al., «Reproducing, analyzing, and detecting reward hacking in Rubric-Based reinforcement learning», *arXiv preprint arXiv:2606.04923*, 2026.
[49] A. Sheshadri et al., «AuditBench: Evaluating alignment auditing techniques on models with hidden behaviors», *arXiv preprint arXiv:2602.22755*, 2026.
[50] M. Ye et al., «What counts as AI sycophancy? a taxonomy and expert survey of a fragmented construct», *arXiv preprint arXiv:2605.21778*, 2026.