УДК 004.89

СРАВНИТЕЛЬНОЕ ИССЛЕДОВАНИЕ МЕТОДОВ ПОВЫШЕНИЯ НАДЁЖНОСТИ API-ВЫЗОВОВ ЯЗЫКОВЫХ МОДЕЛЕЙ НА ОСНОВЕ ПОДХОДА ТОНКОЙ НАСТРОЙКИ GORILLA И ПОДХОДА С ДИНАМИЧЕСКИМ РАССУЖДЕНИЕМ REACT

В.Д. Колесникова[1]

[1]*Самарский университет, г. Самара*
e-mail: bulo45ka@gmail.com

Аннотация

В работе проведён сравнительный анализ двух методов повышения надёжности API-вызовов больших языковых моделей: подход на основе тонкой настройки Gorilla и подход на основе динамического рассуждения ReAct. Рассмотрены архитектурные особенности, преимущества и ограничения каждого метода. Выявлены критерии выбора подхода в зависимости от класса решаемых задач. Показано, что Gorilla эффективен для стабильных API, тогда как ReAct предпочтителен в условиях неопределённости.

Ключевые слова: Большие языковые модели (LLM), автономные агенты, глубокое обучение, нейросетевые модели, дообучение моделей, API-интеграция, интерактивное обучение, гибридные системы, внешние инструменты.

Введение

Современные большие языковые модели демонстрируют высокую эффективность в диалоге и генерации кода, однако остаются ограниченными статичным набором данных. Для выполнения практических задач требуется интеграция с внешними API. Ключевой проблемой являются галлюцинации при генерации вызовов. В данной работе проводится сравнительное исследование двух принципиально разных подходов к решению этой проблемы: метода тонкой настройки модели на форматы API на примере Gorilla и метода динамического рассуждения с обратной связью на примере ReAct.

1. Методы интеграции

Стремление преодолеть ограниченность статичных знаний LLM привело к формированию двух основных направлений интеграции моделей с внешними API. Первое направление предполагает модификацию самой модели для повышения точности формальных вызовов. Второе направление фокусируется на создании внешней управляющей надстройки, которая позволяет модели рассуждать о своих действиях. Рассмотрим данные подходы детально на примере архитектур Gorilla и ReAct.

1.1. Gorilla

Gorilla представляет собой дообученную версию LLaMA-7B, специализированную для трансляции инструкций естественного языка в вызовы API [1]. Обучение производится на наборе данных, автоматически сгенерированном методом Self-Instruct на основе документации API (Torch Hub, HuggingFace). В результате модель может сопоставить пользовательский запрос с конкретной функцией API и

сгенерировать синтаксически корректный вызов. Для задач, требующих актуальных данных, предусмотрен режим с поиском по документации, фрагмент которой добавляется в контекст модели. Схема работы Gorilla показана на рисунке 1.

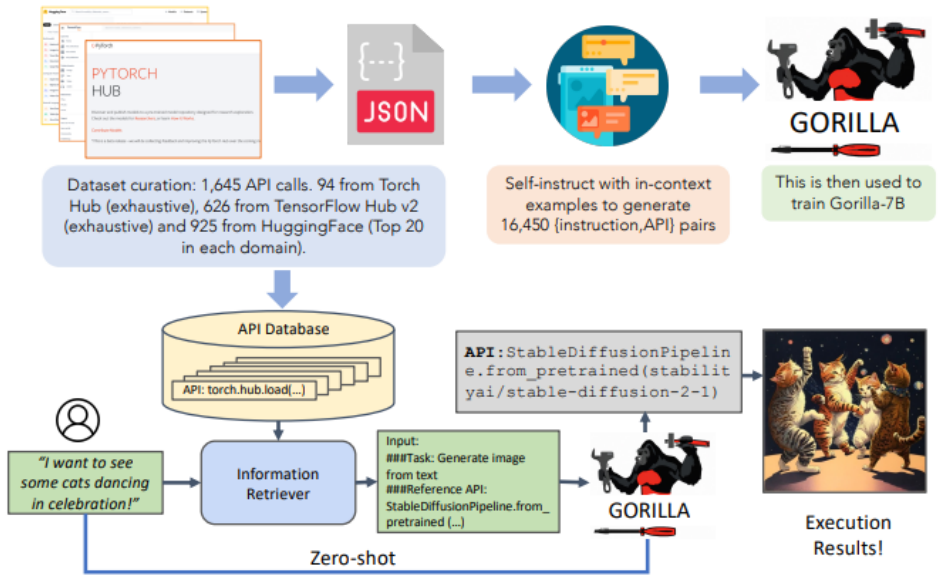

Рисунок 1 – Архитектура взаимодействия Gorilla с внешними API [1]

Ключевое преимущество подхода – высокая точность и низкое число галлюцинаций для API, представленных в обучающей выборке. Недостаток – статичность, ведь при изменении спецификации API требуется повторное дообучение модели.

1.2. ReAct

Метод ReAct реализует итеративный цикл «Мысль – Действие – Наблюдение» [2]. В отличие от Gorilla, генерирующей вызов за один проход, ReAct предполагает, что решение задачи требует последовательности шагов с явным вербальным анализом ситуации. На этапе «Мысль» модель рассуждает о том, какой инструмент следует применить и с какими параметрами. На этапе «Действие» формируется и исполняется вызов API. Полученный результат «Наблюдение» анализируется моделью для планирования следующего шага или формирования итогового ответа.

Преимущество ReAct заключается в гибкости и способности к самокоррекции – неудачный вызов может быть исправлен на следующей итерации цикла. Издержкой этих возможностей является повышенная вычислительная стоимость, обусловленная многократными обращениями к LLM.

2. Сравнительный анализ

Рассмотренные подходы Gorilla и ReAct представляют два принципиально разных способа интеграции языковых моделей с внешними API. Их сопоставление позволяет определить границы применимости каждого метода.

В таблице 1 представлено сравнение Gorilla и ReAct по пяти критериям: принцип работы, устойчивость к галлюцинациям, вычислительные затраты, гибкость при изменении API и интерпретируемость результата.

Таблица 1 – Сравнительный анализ подходов Gorilla и ReAct

Критерии	Gorilla	ReAct
Принцип работы	Однократная генерация синтаксически точного вызова API на основе дообучения	Итеративный цикл «Мысль – Действие – Наблюдение» с пошаговым рассуждением
Устойчивость к галлюцинациям	Высокая для API из обучающей выборки; возможны ошибки на незнакомых функциях	Средняя; ошибки возможны, но исправляются в следующих итерациях
Вычислительные затраты	Низкие; одно обращение к LLM на запрос	Высокие; 3-4 обращения к LLM на задачу
Гибкость при изменении API	Низкая; требуется повторное дообучение модели	Высокая; достаточно обновить описание API в контексте
Интерпретируемость	Низкая; модель выдаёт готовый вызов без объяснений	Высокая; трассировка мыслей делает процесс понятным

Представленные характеристики демонстрируют компромисс между специализацией Gorilla и адаптивностью ReAct. Gorilla достигает высокой точности и минимальной задержки за счёт дообучения, но теряет работоспособность при изменении API. ReAct обеспечивает самокоррекцию и интерпретируемость ценой сильного увеличения вычислительных затрат. Различие в устойчивости к галлюцинациям принципиально – Gorilla надёжна в пределах известного набора API, ReAct способен исправлять ошибки благодаря обратной связи.

Таким образом, Gorilla предпочтителен для стабильных промышленных API с жёсткими требованиями к задержке, ReAct же в условиях часто обновляемых спецификаций и при работе с произвольным набором инструментов.

3. Эмперический анализ данных

Для количественного обоснования сравнительного анализа, представленного в разделе 2, было проведено экспериментальное сравнение двух подходов. Основная серия экспериментов проводилась на англоязычном наборе данных, соответствующем условиям оригинальных исследований [1, 2]. Тестовый набор состоял из 100 задач, равномерно распределённых по трём категориям: получение погоды, математические вычисления и поиск информации. Эксперимент проводился в среде Google Colab с GPU Tesla T4. Методология соответствует подходам из обзорной работы по Tool Learning [3].

Таблица 2 – Количественные показатели эффективности

Критерии	Gorilla	ReAct	Разница
Точность, %	84	62	Gorilla лучше на 22%
Средняя задержка, сек	1,82	5,14	ReAct медленнее в 2,8 раз
Средний расход токенов	447	1283	ReAct больше в 2,9 раз
Среднее число генераций	1,0	3,1	–
Успешных самокоррекций	–	8 из 38 ошибок (21%)	–

Полученные данные подтверждают теоретические выводы. Подход Gorilla продемонстрировал высокую точность при минимальной задержке. Основные ошибки Gorilla были связаны с неоднозначными формулировками запросов. Подход ReAct

показал более низкую первичную точность, однако продемонстрировал способность к самокоррекции. Это подтверждает ключевое преимущество агентного подхода в условиях неопределённости.

Заключение

В рамках проведённого исследования был выполнен сравнительный анализ двух методов повышения надёжности API-вызовов языковых моделей: Gorilla и ReAct. Установлено, что Gorilla обеспечивает высокую точность и минимальные вычислительные затраты при работе со стабильным набором API. Подход ReAct демонстрирует большую гибкость и способность к самокоррекции. Практическая значимость работы заключается в формулировке критериев выбора метода интеграции LLM с внешними инструментами. Перспективным направлением дальнейших исследований представляется разработка гибридных архитектур.

Список использованных источников

1. Gorilla: Large Language Model Connected with Massive APIs / Shishir G. Patil, Tianjun Zhang, Xin Wang, Joseph E. Gonzalez. // ArXiv / Cornell University. – 2023. – URL: https://arxiv.org/abs/2305.15334 (дата обращения: 22.04.2026).

2. ReAct: Synergizing Reasoning and Acting in Language Models / Shunyu Yao, Jeffrey Zhao, Dian Yu [et al.] // ArXiv / Cornell University. – 2023. – URL: https://arxiv.org/abs/2210.03629 (дата обращения: 22.04.2026).

3. Tool Learning with Foundation Models / Y. Qin, S. Hu, Y. Lin [et al.] // ArXiv / Cornell University. – 2023. – URL: https://arxiv.org/abs/2304.08354 (дата обращения: 22.04.2026).

COMPARATIVE STUDY OF API CALL RELIABILITY ENHANCEMENT METHODS FOR LANGUAGE MODELS BASED ON THE GORILLA FINE-TUNING APPROACH AND THE REACT DYNAMIC REASONING APPROACH

V.D. Kolesnikova[1]

[1]Samara University, Samara
e-mail: bulo45ka@gmail.com

Abstract

This paper presents a comparative analysis of two methods for enhancing the reliability of API calls from large language models: the Gorilla fine-tuning-based approach and the ReAct dynamic reasoning-based approach. The architectural features, advantages, and limitations of each method are considered. Criteria for choosing the approach depending on the class of tasks are identified. It is shown that Gorilla is effective for stable APIs, while ReAct is preferable under conditions of uncertainty.

Keywords: Large Language Models (LLM), autonomous agents, deep learning, neural network models, model fine-tuning, API integration, interactive learning, hybrid systems, external tools.