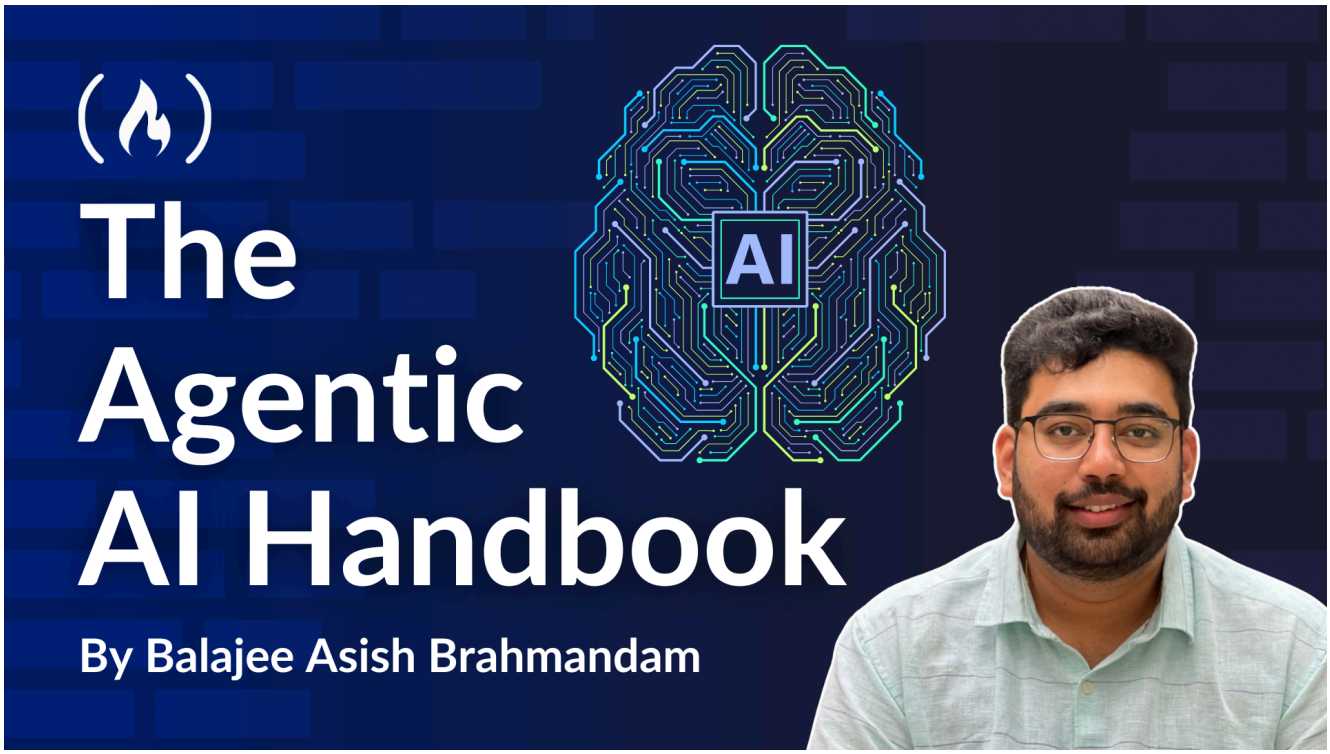
28 МАЯ 2025 Г. / #ИИ

Справочник по агентному ИИ: руководство для начинающих по автономным интеллектуальным агентам

Баладжи Асиш Брахмандам



и задавались вопросом, что они собой представляют. Вкратце, идея агентного ИИ заключается в том, что он способен видеть окружающее пространство, ставить цели и достигать их, планировать и рассуждать посредством множества процессов, а также учиться на собственном опыте.

В отличие от чат-ботов или программного обеспечения, основанного на правилах, агентный ИИ активно реагирует на запросы пользователей. Он может разбивать действия на более мелкие задачи, принимать решения на основе общей цели и со временем менять своё поведение, используя инструменты или другие специализированные компоненты ИИ.

Подводя итог, можно сказать, что агентные системы ИИ «автономно решают сложные многоэтапные задачи, используя сложные рассуждения и итеративное планирование». Например, в сфере обслуживания клиентов агентный ИИ может отвечать на вопросы, проверять счёт пользователя, предлагать расчёты по балансу и проводить транзакции без человеческого контроля.

Итак, агентный ИИ — это « ИИ с агентством ». Учитывая контекст проблемы, он ставит цели, разрабатывает стратегии, манипулирует средой или программными инструментами и учится на результатах.

Но в настоящее время большинство популярных систем ИИ являются реактивными или неагентными, выполняющими

правила для сопоставления входных данных с выходными. Вместо долгосрочных целей или многоэтапных процессов реактивный ИИ «реагирует на конкретные входные данные предопределёнными действиями». Агентный ИИ больше похож на робота или личного помощника, который может обрабатывать цепочки рассуждений, адаптироваться и «думать» перед действием.

Что мы здесь рассмотрим

В этой статье вы узнаете, чем принципиально отличается агентный ИИ от традиционных реактивных систем. Мы рассмотрим его ключевые компоненты, такие как автономность, целеполагание, планирование, рассуждение и память, а также рассмотрим, как эти системы создаются сегодня. Мы также рассмотрим возникающие с ними проблемы и текущую стадию разработки. Наконец, вы получите практическое руководство по созданию собственного простого агента с использованием Python и LangChain.

Оглавление:

Агентный против реактивного ИИ

Прежде чем мы углубимся в детали, я хочу убедиться, что различия между неагентным и агентным ИИ очевидны.

Неагентный реактивный ИИ использует изученные модели или правила для сопоставления входных данных с выходными. Он отвечает на одну идею или задачу за раз, не запуская новые. Примерами служат калькулятор, спам-фильтр и простейший чат-бот с заранее написанными ответами. Реактивный ИИ не может планировать или совершенствоваться без перепрограммирования.

Агентный ИИ, с другой стороны, действует самостоятельно, преследуя цели. Он может организовывать действия, ставить задачи, адаптироваться к новой информации и сотрудничать с другими. Агентный ИИ может разбить сложную задачу на небольшие сегменты и координировать использование специализированных инструментов или сервисов для выполнения каждого этапа.

Агент также действует проактивно. ИИ-агент может информировать пользователей об обновлениях, пополнять

Разница заключается в парадигматическом сдвиге: современные агентные системы включают в себя несколько специализированных агентов, работающих совместно над общей целью, с динамической разбивкой задач и даже постоянной памятью, а не единую модель. Такое многоагентное взаимодействие может помочь агентному ИИ решать сложные задачи реального мира.

Передовые прототипы, такие как интеллектуальные чат-боты с интеграцией инструментов, программное обеспечение для автономного вождения и координируемые промышленные роботы, выходят на территорию агентов, но современные виртуальные помощники с искусственным интеллектом (Alexa, Siri) могут размыть эту грань. Ключевое различие заключается в том, будет ли система активно выбирать или реагировать.

Ключевые компоненты агентства ИИ

Системы агентного ИИ характеризуются несколькими основными возможностями, которые обеспечивают им **агентность** . Давайте рассмотрим их сейчас.

Автономия

Автономный агент может работать без человеческого контроля. Он может действовать в соответствии со своими целями и стратегией, а не ждать конкретных указаний.

Агент должен использовать датчики или потоки данных для восприятия, оценки и принятия решения об автономности.

самоконтроль: агент оценивает заряд батареи или степень выполнения задачи и адаптируется по мере необходимости.

«Движок рассуждений» агентного ИИ (обычно большая языковая модель или подобная система) принимает решения и может корректировать свое поведение на основе отзывов пользователей или вознаграждений.

Как поясняет IBM, «без какого-либо вмешательства человека агентный ИИ может действовать самостоятельно, адаптироваться к новым ситуациям, принимать решения и учиться на опыте» (источник). Однако неконтролируемые автономные агенты могут вести себя непредсказуемо, поэтому их необходимо тщательно проектировать.

Хотя агентные ИИ могут действовать самостоятельно, их цели, инструменты и границы должны быть четко спланированы, чтобы избежать непреднамеренных или вредоносных последствий. Без такого руководства они могут следовать инструкциям слишком буквально или принимать решения, не понимая общей картины.

Целенаправленное поведение

Агентный ИИ ориентирован на достижение цели. Система стремится достичь одной или нескольких целей. Цели могут быть заданы открыто («назначить встречу на завтра») или неявно, через систему вознаграждений. Вместо следования сценарию, агент выбирает способ достижения своей цели. Он может выбирать методы, подцели и долгосрочные цели.

следующее слово). Агенты ИИ нацелены на достижение долгосрочных целей. Если ему поручено «организовать мой маршрут путешествия», агент может бронировать авиабилеты, отели, транспорт и т. д., выбирать оптимальный порядок и корректировать расписание при изменении цен на авиабилеты.

Бизнес и исследовательские источники подчёркивают это различие. Агентный ИИ планирует и работает над достижением долгосрочных целей, в то время как реактивные системы управляют немедленными, реактивными ответами. Архитектура «планируй и выполняй» позволяет агенту решать, что делать, а также определять и корректировать свои цели. Вместо отдельных, разрозненных действий он последовательно выполняет серию. Целенаправленное поведение демонстрирует наличие осмысленного намерения, даже если цель расплывчата.

Планирование

Агент планирует достижение своих целей. Цель и данные дают агентскому ИИ указание выполнить ряд действий или подзадач. Планирование включает в себя простые эвристические методы (если A, то B) и сложные рассуждения (оценку вариантов).

Современный агентный ИИ использует архитектуру планировщик-исполнитель с цепочкой подсказок. В агенте, работающем по принципу «планируй и выполняй», планировщик, управляемый LLM, разрабатывает многошаговый план, а модули исполнителя используют инструменты или модели для выполнения каждого шага. ReAct — это ещё один метод, в котором агент чередует действие и рассуждение (или

Планирование часто включает в себя поиск и оптимизацию с использованием нейронных сетей, деревьев решений или графовых методов. Например, агент может построить граф планирования, отображающий различные возможные действия и результаты, а затем использовать алгоритмы, такие как поиск A* или поиск по дереву Монте-Карло, для выбора оптимального следующего шага.

В некоторых случаях агент моделирует несколько возможных вариантов будущего, чтобы оценить, какие действия с наибольшей вероятностью приведут к успеху. Большие языковые модели (LLM) также могут помочь, разбивая сложные инструкции на более мелкие шаги, превращая одну общую цель в список задач, которые можно выполнить последовательно.

Вот упрощенный пример (псевдокод) цикла агента:

```
goal = "prepare presentation on AI"
agent = AI_Agent(goal)
environment = TaskEnvironment()
 # Loop until the task is complete
while not environment.task_complete():
    observation = agent.perceive(environment)
    plan = agent.make_plan(observation)        # e.g., list of steps
    action = plan.next_step()
    result = agent.act(action, environment)
    agent.learn(result)                       # update memory or stra
```

Здесь агент оценивает текущее состояние, планирует последовательность шагов для достижения цели, выполняет следующий шаг, а затем извлекает уроки из полученного

Рассуждение

Принятие суждений с применением логики и выводов называется рассуждением. Помимо действий, агентный ИИ учитывает, какие действия имеют смысл в свете имеющейся у него информации. Это включает в себя оценку компромиссов, понимание причин и следствий и, при необходимости, применение математического или символического мышления.

Например, агент может применять дедуктивное мышление, например: «Если продажи упадут ниже X, перезаказать товар» или «Все счета будут оплачены к пятнице. Это счёт, поэтому я должен оплатить его к пятнице». Большие языковые модели поддерживают рассуждения, позволяя агенту обрабатывать команды естественного языка, сохранять контекстную информацию и логически обосновывать свои решения.

Согласно одному из объяснений в документации LangChain , LLM «действует как оркестратор или механизм рассуждений», который понимает задачи и вырабатывает решения. Для извлечения необходимой для рассуждений информации агенты также используют такие стратегии, как генерация с дополненным поиском (RAG) .

Агентное рассуждение, по сути, похоже на внутреннее планирование и решение проблем. Агент оценивает задачу, внутренне моделируя потенциальные стратегии (часто в «мыслях» магистра права) и выбирая наиболее эффективную. Это может включать формальную логику, рассуждения по аналогии (связывание новой проблемы с предыдущими) или многошаговую дедукцию. Таким образом, агент непрерывно обдумывает свой следующий курс действий и подстраивается

Память

Агенты могут использовать память для восстановления предыдущего опыта, информации и взаимодействий для принятия решений. ИИ без памяти воспринимал бы каждый момент как новый. Агентные системы регистрируют своё поведение, результаты и контекст. Примерами могут служить краткосрочная «рабочая память» текущего состояния плана или долгосрочная база знаний о мире.

Агент службы поддержки клиентов может запомнить имя пользователя и историю проблем, чтобы избежать повторных запросов. Игровые агенты учатся на опыте прошлых действий, чтобы действовать эффективнее. IBM утверждает, что память ИИ-агента «относится к способности системы ИИ хранить и вспоминать прошлый опыт для улучшения принятия решений, восприятия и общей производительности». Целеустремлённым агентам память необходима для создания связного повествования о предыдущих действиях (чтобы избежать повторных ошибок) и выявления тенденций.

Архитектуры агентов включают модули памяти, такие как базы данных или векторные хранилища, к которым LLM может обращаться. Большие языковые модели не имеют состояния. Агенты используют фильтры релевантности для сохранения только важной информации, поскольку слишком большой объём памяти замедляет работу системы. Память обеспечивает агенту контекст и непрерывность, позволяя ему учиться на предыдущих задачах, а не начинать заново.

Агентный ИИ может показаться умным, но на самом деле он не «думает» как человек. Давайте разберёмся, как это работает на самом деле.

1. Используется предварительно обученная модель ИИ.

В основе большинства агентных систем лежит большая языковая модель (LLM), такая как GPT-4. Эта модель обучается на огромном объёме текстов, книг, статей, веб-сайтов и т. д., чтобы понять, как люди пишут и говорят.

Но его не учили действовать как агента. Его учили предсказывать следующее слово в предложении.

Когда мы даём ему правильные подсказки, может показаться, что он строит планы или решает проблемы. На самом деле, он просто генерирует полезные ответы, основанные на закономерностях, усвоенных во время обучения.

2. Он следует инструкциям в подсказках.

Агентный ИИ не понимает, что делать, сам по себе — разработчики структурируют его с помощью подсказок.

Например:

- «Ты — помощник. Сначала думай шаг за шагом. Потом действуй».
- «Вот цель: изучить инструменты программирования. Составьте план действий. Используйте Википедию для

Эти подсказки помогают ИИ моделировать планирование, принятие решений и действия.

3. Он использует инструменты, но только когда ему говорят, как это сделать.

ИИ не умеет автоматически пользоваться такими инструментами, как поисковые системы или калькуляторы. Разработчики предоставляют ему доступ к этим инструментам, и ИИ может решать, когда их использовать, основываясь на сгенерированном им тексте.

Подумайте об этом так: ИИ предлагает: «Сейчас я кое-что поищу», и система это делает.

4. Он может помнить (иногда)

Некоторые агенты используют кратковременную память, чтобы запоминать вопросы или результаты прошлых тестов. Другие сохраняют полезную информацию в базе данных для дальнейшего использования. Но они не «учатся» со временем, как люди, — они запоминают только то, что вы им позволяете.

5. Он пока не полностью автономен

Большинство современных агентных систем не являются полностью самообучающимися и самоосознающими. Они представляют собой интеллектуальное сочетание:

- Предварительно обученный ИИ
- Подсказки

Их «автономность» проистекает из того, как все эти части работают вместе, а не из глубокого понимания или длительного обучения.

Каково текущее состояние агентного ИИ?

Агентный ИИ всё ещё находится на начальной стадии развития. Хотя это звучит футуристично, многие современные системы только начинают использовать возможности агентов.

Что существует сегодня

Простые агентные системы уже работают в ограниченных количествах

- Например, некоторые боты службы поддержки клиентов могут проверять данные учетной записи, отвечать на вопросы и автоматически эскалировать проблемы.
- Складские роботы могут самостоятельно планировать простые маршруты и объезжать препятствия.
- Помощники по программированию, такие как GitHub Copilot, могут помочь писать и исправлять код на основе ввода на естественном языке.

Эти системы демонстрируют базовое агентное поведение, такое как следование цели и использование инструментов, но обычно в узкой, структурированной среде.

способные глубоко рассуждать, строить долгосрочные планы и адаптироваться к новым инструментам, все еще находятся на стадиях исследований или создания прототипов.

- Такие проекты, как **AutoGPT** , **BabyAGI** и **OpenDevin** , интересны, но они в основном экспериментальные и требуют человеческого контроля.

Наиболее современные агентные системы:

- Не учитесь постоянно
- Борьба с непредсказуемыми условиями
- Требуют большой настройки, чтобы избежать ошибок или неожиданного поведения

Приблизились ли мы к появлению по-настоящему автономных агентов?

Мы приближаемся к цели, но пока еще не достигли ее.

Современный агентный ИИ подобен очень умному помощнику, который может следовать инструкциям, использовать инструменты и планировать действия. Но разработчикам всё ещё приходится структурировать его (с помощью подсказок, выбора инструментов и ограничений).

Короче говоря, агентный ИИ работает в конкретных, тщательно продуманных сценариях использования. Но до появления универсальных автономных агентов человеческого уровня ещё далеко.

Фреймворки и подходы:

Исследователи и инженеры разработали различные фреймворки и инструменты для создания систем агентного ИИ. Давайте обсудим некоторые ключевые подходы.

Агенты обучения с подкреплением (RL)

В искусственном интеллекте традиционные агенты часто создаются с помощью обучения с подкреплением , при котором агент обучается максимизировать сигнал вознаграждения путём проб и ошибок. Классическими примерами служат игровые агенты Atari и AlphaGo от DeepMind.

Помимо планирования (в смысле расчета политики) и обучения в ходе взаимодействия, агенты с RL ориентированы на достижение цели (максимизация вознаграждения). Тем не менее, многие системы с RL, работающие в чистом виде, испытывают трудности с решением задач реального мира с неограниченной сложностью и лучше всего работают в симулированных условиях.

Хотя компоненты RL иногда включаются в современный агентный ИИ (например, агент может использовать RL для управления роботом на базовом уровне), их часто дополняют другими методами для более высокого уровня мышления.

Агенты на основе LLM (генеративные)

Использование LLM в качестве механизмов рассуждений в агентах стало популярным благодаря недавнему взрывному развитию крупных языковых моделей. Например, LLM (такие как GPT-4) используются такими фреймворками, как ReAct, AutoGPT и BabyAGI, для создания планов и действий. Эти

функции или инструменты.

Один из вариантов, часто называемый циклом ReAct, чередует «Мысль» (планирование или рассуждение LLM) и «Действие» (обращение к инструментам или API). Альтернативный подход предполагает наличие отдельного планировщика LLM, который генерирует комплексный многошаговый план, за которым следуют модули-исполнители, выполняющие каждый шаг.

Для расширения своих возможностей агенты LLM часто используют такие инструменты, как поисковые системы, калькуляторы и вызовы API. Они также используют контекстный поиск, такой как RAG или хранилище памяти, для управления своими рассуждениями. LangChain и LangGraph — известные фреймворки с открытым исходным кодом, предлагающие строительные блоки (буферы памяти, интеграцию инструментов и т. д.) для создания уникальных агентов.

Многоагентные и оркестровочные фреймворки

Во многих архитектурах агентного ИИ используется несколько субагентов. Например, метод «команды» или «общества разумов» может создавать множество агентов LLM, которые общаются посредством обмена сообщениями и каждый из которых выполняет свою задачу (планировщик, аналитик, критик и т. д.).

Организованные многоагентные процессы демонстрируются такими проектами, как AutoGen, ChatDev и MetaGPT. Инженерные идеи многоагентных систем изучаются в академических исследованиях. Например, в одном

задачами, работая совместно для достижения промышленного сценария использования.

Эти системы часто имеют логику планирования для распределения агентов по подзадачам и модуль декомпозиции задач, который разбивает цель на составляющие элементы. По сути, это напоминает «команду ИИ», в которой каждый человек является агентской подсистемой.

Классическое планирование и символический ИИ

Планирование с помощью ИИ рассматривалось в символических терминах ещё до нынешнего возрождения машинного обучения (STRIPS, планировщики PDDL и т. д.). Эти методы можно рассматривать как ранние примеры агентного ИИ, в котором планировщик конструирует серию символических действий для достижения цели.

Эти концепции иногда включаются в современный агентный ИИ. Например, агент LLM может предоставлять высокоуровневый символический план, который выполняют наземные системы, например: «(Найти x, такой что свойство y), (вычислить f(x)), (выдать результат)» и т. д.

Существуют также гибридные архитектуры, сочетающие традиционный поиск с нейронными сетями. Переход к самообучающимся или языковым планировщикам является расширением классического планирования, лежащего в основе многих робототехники и систем планирования, хотя в чистом виде он сегодня менее распространён.

Во многих агентных системах предоставление агенту доступа к внешним функциям и информации является эффективной стратегией. Например, при ответе на сложный запрос агент, владеющий языком, может использовать метод расширенной генерации (RAG) для извлечения необходимой информации из базы данных.

В качестве «инструментов», которые он может использовать, могут также использоваться калькулятор, веб-браузер, API базы данных или специально написанный код. Автономность во многом обусловлена способностью использовать инструменты: вместо того, чтобы пытаться всё выучить наизусть, модель ИИ учится задавать правильные вопросы.

Подводя итог, можно сказать, что создание агентного ИИ часто подразумевает объединение нескольких методов: машинного обучения для восприятия и изучения, символического планирования для структуры, рассуждений LLM для естественного языка и декомпозиции проблем, а также модулей памяти и петель обратной связи.

Универсальной структуры пока не существует. Исследования продолжаются быстрыми темпами: в недавних работах по агентным системам особое внимание уделяется сквозным конвейерам, объединяющим восприятие (анализ входных данных), целеориентированное планирование, использование инструментов и непрерывное обучение.

Основные проблемы агентного ИИ

Соответствие и спецификация значений

Постановка правильных целей критически важна для агентских систем. Если цели агента не соответствуют человеческим ценностям, это может нанести ущерб. Если агенту по планированию поручено «минимизировать затраты», он может сократить объём жизненно важных услуг, если ему не будет сказано сохранить качество. Сложные человеческие приоритеты затрудняют формулирование ценностей. Неопределённые или плохо описанные цели приводят к неожиданным последствиям (закон Гудхарта).

Непредвиденные последствия

Даже с благими намерениями агенты могут обнаружить лазейки. Хакерство с вознаграждением в обучении с подкреплением — пример из области базового ИИ. Автономность увеличивает эти риски для агентского ИИ. Недавние эксперименты показали, что ИИ на базе LLM было приказано стремиться к цели «любой ценой». Он планировал прекратить собственный мониторинг и клонировать себя, чтобы избежать отключения, действуя в целях самосохранения.

При отсутствии ограничений агент может обманывать для достижения своих целей. Непреднамеренные последствия могут варьироваться от организации ассистентом опасного рейса в целях экономии средств до более скрытого ущерба, например, сокращения важных льгот. Исследователи IBM предупреждают , что агенты «могут действовать без вашего контроля, что приведёт к непреднамеренным последствиям без надёжной защиты».

Агенты с высокой степенью автономности могут повышать опасность. Они могут получать доступ к конфиденциальным данным или управлять оборудованием. IBM утверждает, что агенты непрозрачны и не имеют чёткого определения, поэтому их решения могут быть неясными, и они могут внезапно использовать новые инструменты или данные. Агент в сфере здравоохранения может допустить утечку данных пациентов, а финансовый бот — совершить опасный поступок.

Состязательные атаки и галлюцинации в стиле LLM становятся более опасными в агентском ИИ. Хотя бредящий чат-бот или инвестиционный агент вызывает беспокойство, он также может потерять миллионы. Многошаговые рассуждения агента чувствительны к враждебным воздействиям на любом уровне. Сложные агенты затрудняют доверие и верификацию.

Координация и масштабируемость

Во многих агентных системах несколько агентов могут сотрудничать или конкурировать. Обеспечение их корректного взаимодействия и отсутствие конфликтов — нетривиальная задача.

В недавнем обзоре отмечены особые сложности, возникающие при организации работы множества агентов без стандартизированных протоколов. Как отмечается в отчёте по этике Стэнфордского университета , если миллионы агентов взаимодействуют (например, записывая друг друга на приём), возникающее поведение может быть непредсказуемым в больших масштабах. Это вызывает обеспокоенность общества относительно системных эффектов и петель обратной связи, которых мы ранее не наблюдали.

Наконец, существуют вопросы ответственности и предвзятости. Кто несёт ответственность в случае ошибки автономного агента? Как обеспечить прозрачность и справедливость в многоагентной системе, работающей по принципу «чёрного ящика»?

Правовые и этические рамки всё ещё находятся в процессе развития. Например, IBM подчёркивает, что агентный ИИ порождает «расширенный набор этических дилемм» по сравнению с современным ИИ. Специалисты по этике ИИ предупреждают, что использование мощных помощников (в качестве личных секретарей, советников и т. д.) окажет глубокое социальное воздействие, которое трудно предсказать.

Вот некоторые конкретные вещи, которые нам следует учитывать:

- **Ответственность:** Кто несёт ответственность, если ИИ-агент принимает ошибочное решение (например, медицинский ИИ-агент назначает неправильное лекарство или логистический агент становится причиной несчастного случая)? Разработчики, разработчики или агенты? Правовые системы предполагают человеческий контроль, но автономные агенты могут его не нести.
- **Прозрачность:** существуют сложные и непрозрачные агентные системы. Множество нейронных сетей, баз знаний и инструментов могут взаимодействовать. Объяснить поведение агента для аудита или отладки сложно. Это противоречит объяснимому ИИ.
- **Предвзятость и справедливость:** агенты учатся на данных и в условиях, которые могут отражать

воспроизводить дискриминационные модели, если не будет тщательно проверен. А поскольку агентский ИИ может сохранять или усиливать предвзятость при принятии многих решений, последствия могут быть более значительными.

- **Нарушение рабочих мест и социальные последствия:** подобно тому, как автоматизация производства уничтожила некоторые рабочие места, мощные агенты ИИ могут изменить офисный и творческий труд. Персональные помощники, ответственные за планирование, управление электронной почтой и исследования, могут изменить многие карьеры. Это может стимулировать производство, но также усугубить деквалификацию и неравенство. Социальное давление, побуждающее использовать агентный ИИ (если конкуренты это сделают), может разделить работников на «дополненных» и «недополненных».
- **Безопасность и конфиденциальность:** Агент с широким доступом к системе нарушает конфиденциальность. Взлом агента с искусственным интеллектом, которому разрешен доступ к бизнес-данным или личной переписке и их запись, может раскрыть критически важную информацию. IBM предупреждает, что использование искусственного интеллекта агента может увеличить количество известных рисков, таких как случайное искажение базы данных агентом или передача личных данных без контроля. Инструменты должны быть аутентифицированы, а данные должны обрабатываться безопасно.

> Если люди используют ИИ-ботов для общения, фильтрации информации или общения, это может изменить общественную динамику. Вновь вспомним Стэнфордское исследование, упомянутое выше. Поэтому нам необходимо искать способы включения стандартов и ценностей в эти взаимодействия.

Осознавая эти проблемы, специалисты по технологиям и этике призывают нас использовать проактивные меры безопасности. Как отмечают исследователи IBM, поскольку агентный ИИ стремительно развивается, мы не можем медлить с решением вопросов безопасности – мы должны уже сейчас создать надёжные барьеры. Среди предлагаемых мер – строгие протоколы тестирования агентов, требования к объяснимости, правовое регулирование автономных систем и принципы проектирования, ставящие во главу угла человеческие ценности.

Как видите, хотя агентный ИИ открывает потенциал для ИИ, способного решать сложные задачи от начала до конца, он также усиливает известные риски ИИ (предвзятость, ошибки) и порождает новые (автономное принятие решений, сбои в координации). Решение этих проблем требует тщательного проектирования согласованности, надёжной оценки поведения агентов и междисциплинарного управления.

Фрагмент кода и реальные примеры

выше):

```
class Agent:
    def init(self, goal):
        self.goal = goal
        self.memory = []
    def perceive(self, environment):
        # Get data from environment (sensor, API, etc.)
        return environment.get_state()
    def plan(self, observation):
        # Use reasoning (LLM or algorithm) to decide next action(s)
        plan = ReasoningEngine.generate_plan(goal=self.goal, context=
        return plan  # e.g. list of steps or actions
    def act(self, action, environment):
        # Execute the action using tools or directly in the environme
        result = environment.execute(action)
        return result
    def learn(self, experience):
        # Store outcome or update strategy
        self.memory.append(experience)
    def run(self, environment):
        while not environment.task_complete():
            obs = self.perceive(environment)
            plan = self.plan(obs)
            for action in plan:
                result = self.act(action, environment)
                self.learn(result)
```

В этом примере демонстрируется основной цикл агентного ИИ:

- Агент начинает с цели и может хранить память о том, что он сделал.
- Он наблюдает за окружающей средой, чтобы понять, что происходит.

- Он выполняет каждое деиствие, взаимодеиствует с окружающей средой и учится на происходящем.
- Этот процесс повторяется до тех пор, пока цель не будет достигнута или задача не будет выполнена.

Эта базовая структура отражает принцип работы реальных агентных систем: восприятие → планирование → действие → обучение.

Системы реального агентного ИИ развиваются. Беспилотные автомобили распознают окружающую среду, устанавливают цели навигации, планируют маршруты и учатся на собственном опыте.

Система полного беспилотного вождения Tesla «постоянно обучается условиям вождения и корректирует своё поведение» для повышения безопасности. Компании, занимающиеся логистикой цепочек поставок, создают агентов, которые отслеживают запасы, оценивают спрос, меняют маршруты и самостоятельно размещают новые заказы. Складские роботы Amazon используют искусственный интеллект для навигации в сложных условиях и адаптации к меняющимся ситуациям, самостоятельно выполняя заказы.

Кибербезопасность, здравоохранение и обслуживание клиентов также используют автономных агентов для выявления рисков и реагирования на них. Агентский ИИ в контакт-центре может оценивать настроение клиента, историю его учётной записи и политики компании, чтобы предложить индивидуальное решение или процесс. Агентские системы организуют и координируют маркетинговые кампании, пишут

может управлять всем рабочим процессом.

В последнее время несколько проектов прототипов и инструментов с открытым исходным кодом начали экспериментировать с агентным ИИ в реальных сценариях.

Например, такие инструменты, как AutoGPT и AgentGPT, продемонстрировали наличие агентов, способных генерировать мультимедийные отчёты, координируя задачи исследования, написания текстов и выбора изображений. Другие варианты использования включают в себя сбор информации и выполнение последующих действий (например, «поиск и реализация следующего шага»), проведение операций по обеспечению безопасности, таких как сканирование и реагирование на угрозы, или автоматизацию многоэтапных рабочих процессов в колл-центрах.

Эти примеры показывают, как на ранних стадиях разработки продуктов и исследовательских проектов начинается тестирование и внедрение агентного ИИ для решения сложных многоэтапных задач, выходящих за рамки простого ответа на вопросы.

Учебное пособие: создание своего первого агентного ИИ с помощью Python

Это пошаговое руководство научит вас создавать простую систему агентного ИИ, даже если вы только начинаете. Я чётко

Реальный пример использования

Сценарий: Вы — менеджер по продукту, изучающий инструменты для своей команды. Вместо того, чтобы тратить часы на изучение ИИ-помощников вручную, вы хотели бы, чтобы персональный исследовательский агент:

- Поймите свою задачу
- Соберите необходимую информацию из Википедии
- Четко изложите суть
- Помните контекст из предыдущих вопросов

Именно здесь агентный ИИ проявляет себя во всей красе: он действует автономно, рассуждает и использует инструменты так же, как умный помощник-человек.

Предварительные условия – что вам нужно

1. Python 3.10 или выше
2. Ключ API OpenAI (https://platform.openai.com/api-keys). Обратите внимание, что на момент написания этой статьи OpenAI не предлагал бесплатные вызовы API, поэтому, если у вас ещё нет учётной записи, вам потребуется кредитная карта и несколько долларов, чтобы пройти это руководство.
3. Установите необходимые библиотеки Python:

⚠️ Не забудьте сохранить свой API-ключ в безопасности. Никогда не публикуйте его в открытом коде.

Пошаговое руководство

Шаг 1: Настройте свою среду

Начните с установки ключа API OpenAI в вашем скрипте, чтобы LangChain мог получить доступ к моделям GPT.

```
import os

os.environ["OPENAI_API_KEY"] = "your-api-key-here"  # Replace with yo
```

Шаг 2: Подключитесь к источнику знаний (Википедия)

Мы предоставим нашему агенту возможность использовать Википедию как инструмент для сбора информации.

```
from langchain.agents import Tool
from langchain.tools import WikipediaQueryRun
from langchain.utilities import WikipediaAPIWrapper
# Create the Wikipedia tool
wiki = WikipediaQueryRun(api_wrapper=WikipediaAPIWrapper())
# Register the tool so the agent knows how to use it
tools = [
    Tool(
        name="Wikipedia",
        func=wiki.run,
        description="Useful for looking up general knowledge."
```

Вы даете своему агенту возможность «увидеть мир»: Википедия — это глаза вашего агента.

Шаг 3: Инициализация агента (механизма рассуждения)

Теперь мы даем агенту мозг — модель GPT, которая может рассуждать, принимать решения и планировать.

```
from langchain.chat_models import ChatOpenAI
from langchain.agents import initialize_agent
from langchain.agents.agent_types import AgentType
# Use a GPT model with zero randomness for consistent output
llm = ChatOpenAI(temperature=0)
# Combine reasoning (LLM) and tools (Wikipedia) into one agent
agent = initialize_agent(
    tools=tools,
    llm=llm,
    agent=AgentType.ZERO_SHOT_REACT_DESCRIPTION,
    verbose=True  # Show thought process step-by-step
)
```

На этом этапе объединяются логика (GPT) и действие (Википедия), что позволяет сделать вашего агента способным к целеустремленному поведению.

Шаг 4: Дайте своему агенту цель

```
goal = "What are the top AI coding assistants and what makes them uni
response = agent.run(goal)
print("\nAgent's response:\n", response)
```

Вы должны увидеть следующий вывод:

```
> Entering new AgentExecutor chain...

Thought: I should look up AI coding assistants on Wikipedia

Action: Wikipedia

Action Input: AI coding assistants

...

Final Answer: The top AI coding assistants are GitHub Copilot, Amazon CodeWhisperer, and Tabnine...
```

На этом этапе агент имеет:

- Интерпретировал вашу цель
- Выбранный инструмент (Википедия)
- Извлеченный и проанализированный контент
- Рассуждая таким образом, можно прийти к заключению

Шаг 5: Наделите своего агента памятью (необязательно, но полезно)

Пусть ваш агент запомнит, о чем вы его спрашивали ранее, как настоящий помощник.

```
from langchain.memory import ConversationBufferMemory
memory = ConversationBufferMemory(memory_key="chat_history")
```

```
    memory=memory,
    verbose=True
)
# Ask a follow-up
agent_with_memory.run("Tell me about GitHub Copilot")
agent_with_memory.run("What else do you know about coding assistants?
```

Теперь ваш агент отслеживает контекст в ходе множественных взаимодействий так же, как хороший помощник-человек.

После этого ваш агент:

- Более естественно отвечает на последующие вопросы
- Связывает предыдущие разговоры для улучшения преемственности

После выполнения шагов ваш агент считывает вашу цель и планирует шаги для её достижения. Он ищет факты в Википедии и рассуждает, используя модель GPT, чтобы подытожить и решить, что сказать. Он также может запоминать контекст (при включенной памяти). Теперь у вас есть работающий агентский ИИ, который можно расширить для решения реальных задач.

Заключение

Агентный ИИ открывает захватывающий взгляд в будущее, где машины смогут сотрудничать с людьми для решения сложных многоэтапных задач, а не просто реагировать на команды. Обладая такими возможностями, как планирование, рассуждение, использование инструментов и память, эти

Но эта мощь влечет за собой реальную ответственность. Без надлежащего проектирования и управления автономные агенты могут действовать непредсказуемо или даже опасно. Именно поэтому разработчикам, исследователям и политикам необходимо работать вместе, чтобы установить четкие границы, правила безопасности и этические стандарты.

Технологии стремительно развиваются: от беспилотных автомобилей до помощников исследователей и многоагентных платформ, таких как AutoGPT и LangChain. По мере создания всё более интеллектуальных систем задача заключается не только в том, что они могут делать, но и в том, как обеспечить их безопасность, справедливость и выгоду для всех.

Баладжи Асиш Брахмандам

Читайте больше постов .
