11 ИИ-инструментов, которые ускоряют создание IT-продуктов на всех этапах разработки

Искусственный интеллект становится неотъемлемой частью разработки IT-продуктов. ИИ не только ускоряет процессы, но и повышает их качество и эффективность. Мы рассмотрим 11 инструментов ИИ, которые активно используются в индустрии разработки программного обеспечения.

Согласно исследованиям, глобальный рынок ИИ в IT-индустрии вырастет до $271,9 млрд к 2028 году, демонстрируя среднегодовой темп роста 27,1%. Это связано с растущим спросом на автоматизацию бизнес-процессов, повышение эффективности и инновации.

ИИ внедряется для оптимизации бизнес-процессов и улучшения производительности. Он способен выполнять монотонные задачи быстрее и точнее, чем человек. Анализ больших объемов данных и выявление закономерностей помогает компаниям принимать более обоснованные решения

С каждым годом использование искусственного интеллекта в IT-индустрии становится все более широким и глубоким, проникая во все сферы бизнеса и повседневной жизни. Мы собрали 11 инструментов, которые помогают на основных этапах при разработке продуктов:

Генерация идей и концептуализация

Эти процессы играют ключевую роль в разработке продукта, а при вовлечении искусственного интеллекта могут стать более эффективными.

1. **ChatGPT**

Модели GPT генерируют идеи и решения, помогают в мозговом штурме и уточняет концепции. Обрабатывают большие объемы данных и предоставляют разнообразные перспективы при правильном использовании подсказок.

Пример использования:

Компания работала над новым сервисом для управления проектами, который требовал текстового контента для пользовательского интерфейса и обучающих материалов. UX-писателя в команде не было, а другие специалисты были загружены работой, и процесс написания текстов затянулся.

Тогда аналитик проекта предложил использовать ChatGPT для автоматизации этого процесса. Он создал прототип интеграции ChatGPT с внутренней системой управления контентом, которая позволяла генерировать тексты на основе введенных ключевых слов и требований.

Команда начала использовать этот инструмент для быстрого создания текстов. Например, когда дизайнеры разрабатывали новые экраны сервиса, они вводили описание функциональности в ChatGPT, и инструмент генерировал подходящий текст для этих экранов (который оставалось только проверить). Это позволило команде сократить время на написание текстов с нескольких дней до нескольких часов.

Пример прототипаПроект был завершен на месяц раньше запланированного срока, и продукт получил высокие оценки за информативность и удобство использования.

2. **Gluecharm**

Этот инструмент использует искусственный интеллект для анализа идей продуктов и функций и мгновенно воплощает их в спецификациях разработки, варианты использования, диаграммы и пользовательские истории (user story). Помогает Agile-командам создавать понятные и лаконичные пользовательские истории, критерии приемки и схемы процессов. Облегчает обмен знаниями, адаптацию, повышает производительность, обеспечивает контроль качества и эффективную реализацию. Помогает провести мозговой штурм и создать техническую документацию для быстрого перехода к разработке.

Пример использования:

Компания занимается разработкой новой функциональности для мобильного приложения, которое помогает путешественникам организовывать свои поездки. Заказчик поставил задачу добавить функцию «Рекомендации местных достопримечательностей» на основе местоположения пользователя.

Аналитик использовал Gluecharm для формулирования пользовательской истории: «Я путешественник и хочу видеть рекомендации местных достопримечательностей, чтобы исследовать новые места во время моего путешествия». Инструмент помог ему структурировать историю, добавить атрибуты (например, приоритет рекомендаций, транспортную доступность, уровень сложности). Также Gluecharm связал эту историю пользователя с другими в проекте.

3. **Frase**

Использует искусственный интеллект для генерации идей контента, помощи в SEO и оптимизации контент-стратегий. Помогает создавать высококачественный контент, соответствующий намерениям пользователя. Среди его функций анализ исследований SERP, составление контента и брифов с помощью ИИ.

Пример использования:

Автор из команды IT-разработки готовит посты в блоге компании об использовании новых технологий в создании программного обеспечения. В создании качественного контента для целевой аудитории и видимости в поисковых результатах помогает аналитик. С помощью Frase он исследует страницы с результатами поиска и анализа контента, чтобы определить наиболее популярные запросы и темы в этой нише. И с помощью инструмента создает бриф для автора, включающий ключевые слова, структуру контента и рекомендации по оптимизации для лучшего ранжирования в поисковых системах.

Проектирование и прототипирование

ПО для проектирования на базе ИИ может помочь в создании более эффективных продуктов. Этот процесс включает использование алгоритмов генеративного проектирования, которые предлагают оптимальные проектные решения на основе конкретных критериев, таких как требования к материалу, стоимости и производительности. При создании прототипов ИИ может моделировать работу продукта, позволяя дизайнерам вносить коррективы до разработки физической модели.

1. Miro Assist

Инструмент, который помогает командам на ранних этапах разработки продукта, когда заинтересованные стороны обсуждают идеи. Он улучшает встречи по мозговому штурму и заполняет пробелы с помощью ИИ. Среди его возможностей — создание заметок для подведения итогов обсуждения, преобразование текста в изображение, сопоставление пользовательских историй с помощью персонажей

пользователей, составление диаграмм последовательности для обзора основной идеи, генерация блоков кода с возможностями обработки естественного языка.

Пример использования:

Представим, что компания «Ромашка» создает продукт для клиента — усовершенствованная система управления контентом (CMS) для крупных корпораций. В команде есть аналитик, менеджер по продукту и 2 разработчика.

На начальном этапе разработки, во время сбора идей и требований менеджер по продукту организует встречу в Miro, создавая доску с различными секциями для функциональности, дизайна, технических ограничений и требований заказчика.

Во время встречи и мозгового штурма все участники команды используют интерактивные маркеры и стикеры для представления своих идей. Аналитик применяет функцию Miro Assist для быстрого форматирования и структурирования информации, чтобы упорядочить идеи и сделать их более понятными.

Когда обсуждается интеграция с другими системами, инструмент предлагает список популярных API и инструментов, которые могут быть использованы. Также команда использует интерактивные инструменты для голосования и оценки идей, визуализируя результаты опроса.

После встречи автоматически создается резюме и плана действий на основе обсуждаемых идей и решений, которое рассылается всем участникам.

Так ИИ-инструмент помог улучшить коммуникацию и эффективность работы команды на ранних этапах разработки IT-продукта.

Разработка

Генерация кода искусственным интеллектом работает на основе ML-алгоритмов, обученных с использованием существующего исходного кода, часто полученного из open-source-проектов.

Существует три основных метода ИИ-генерации кода:

- **Функция автозаполнения.** Разработчики инициируют написание кода, а инструмент искусственного интеллекта пытается автоматически завершить код на основе шаблонов, полученных из набора обучающих данных.
- **Ввод на естественном языке.** Разработчики формулируют намерения посредством ввода естественного языка, побуждая инструмент искусственного интеллекта генерировать предложения по коду, соответствующие их целям.
- **Прямое взаимодействие.** Разработчики напрямую общаются с ИИ, используя интерфейс чата, отправляя конкретные запросы или команды для исправления ошибок, демонстрируя диалоговые возможности технологии. Пользователь вводит текстовые подсказки, описывающие желаемую функциональность кода. Генеративные инструменты искусственного интеллекта реагируют на них, предлагая фрагменты кода или генерируя полные функции.

Рассмотрим инструменты:

1. **Copilot**

Разработан GitHub в сотрудничестве с OpenAI, инструмент завершения кода на базе искусственного интеллекта, который может легко интегрироваться в

популярные интегрированные среды разработки (IDE), такие как Visual Studio Code, предлагая контекстно-зависимые предложения и дополнения кода по мере ввода.

Он использует Codex OpenAI, языковую модель, обученную на различных репозиториях кода, для генерации предложений по коду по мере ввода разработчиками. OpenAI Codex является наиболее мощным в Python, но он также поддерживает другие языки, включая JavaScript, Go, Perl, PHP, Ruby и TypeScript.

Пример использования:

Представим, что компания XYZ, специализирующаяся на разработке программного обеспечения, решила использовать GitHub Copilot для повышения производительности своих специалистов и улучшения качества кода. Вот как команда может использовать Copilot в своей повседневной работе:

- **Написание основного кода.** Разработчик начинает набирать код, определяя основные функции. По мере ввода кода, GitHub Copilot предлагает шаблоны функций, которые можно быстро принять, нажав Tab.
- **Оптимизация кода.** После того как основные функции написаны, разработчик решает оптимизировать их, добавив валидацию данных. Когда разработчик начинает набирать код для валидации, Copilot предлагает стандартные методы валидации, которые можно легко интегрировать в существующий код.
- **Тестирование и отладка.** После написания кода разработчик запускает тесты, и при обнаружении ошибок Copilot помогает быстро исправлять их, предлагая возможные решения на основе контекста ошибки.
- **Рефакторинг.** Когда код уже работает, разработчик решает улучшить его структуру. Copilot предлагает различные подходы к рефакторингу, такие как использование классов для управления юзерами, что упрощает дальнейшее расширение функциональности.
- **Code Review.** Перед слиянием изменений в основную ветку другой разработчик проводит ревью кода. Copilot помогает в этом процессе, предлагая вопросы и предложения по улучшению кода, что повышает его качество и ускоряет процесс ревью.

2. **Модели OpenAI GPT**

О них мы упоминали в разделе о генерации идей и концептуализации, можно точно настроить и для задач генерации кода. Хотя ChatGPT не предназначен специально для нее, все же это возможно. Разработчики могут взаимодействовать с моделями, используя подсказки на естественном языке для получения фрагментов кода. В отличие от GitHub Copilot, ChatGPT не интегрирован с IDE и имеет собственный интерфейс, что позволяет ему работать с множеством популярных языков программирования, включая Python, JavaScript, C++, Java, Ruby, C#, PHP и Go. При желании можно поэкспериментировать даже с Dart, R или Lua.

3. **Инструменты Google**

У Google есть несколько инструментов для генерации кода искусственного интеллекта, каждый из которых имеет свои сильные стороны и направленность.

- **Google Gemini (ранее Bard)**

Обучена работе с обширным набором данных, что позволяет ей генерировать изображения, текст и код. Поддерживает C++, Go, Java, JavaScript, Python и TypeScript.

- **Vertex AI от Google Cloud**

Использует языковую модель Pathways 2 (PaLM 2) для генерации текста и кода в ответ на диалоговые подсказки.

- **Gemini Code Assist (ранее Duet AI for Developers)**

Второй пилотный проект с искусственным интеллектом, основанный на моделях Google, который работает в IDE (таких как VS Code или PyCharm), предлагая помощь в кодировании в реальном времени, аналогичную GitHub Copilot.

4. **Code Llama от Meta**

Модель искусственного интеллекта с открытым исходным кодом, основанная на Llama 2, предназначена для генерации и обсуждения кода. Он справляется с задачами кодирования среди общедоступных программ LLM. Модель призвана оптимизировать рабочие процессы разработчиков, облегчить их обучение, а также повысить надежность программного обеспечения и документацию. Поддерживает Python, C++, Java, PHP, Typescript (Javascript), C# и другие.

5. **TabNine**

Инструмент автоматического завершения на базе искусственного интеллекта, который интегрируется с различными редакторами кода (IDE), такими как VS Code, IntelliJ и Eclipse. Инструмент использует LLM, который обрабатывает последовательные данные и выдает ответы на основе знаний, полученных из обучающих данных. Поддерживает JavaScript, Java, Python, TypeScript, PHP и C++.

В чем подвох?

Внедрение генерации кода ИИ сопряжено с некоторыми проблемами. Исследование, опубликованное в IEEE Transactions on Software Engineering, показывает, что ChatGPT не всегда справляется с поставленной задачей — и, более того, не всегда предлагает качественное решение: успешность варьируется от 89% для самых простых промтов до 0,66% в случае объемной постановки вопроса. Подобные данные вызывают серьезную обеспокоенность по поводу надежности и качества кода, сгенерированного ChatGPT, подчеркивая потенциальные риски, связанные с его широким использованием.

Как и в случае с любой зарождающейся технологией, к ней есть вопросы:

1. **Надежность.** Можно ли доверять коду, созданному ИИ? Исследования показали, что, хотя в целом ИИ-помощники надежны, они иногда могут создавать ошибочный или небезопасный код, что подчеркивает необходимость тщательной проверки кода. Исследование, изучавшее точность кода, сгенерированного Copilot, показало, что качество генерации кода этим продуктом постепенно снижается после релиза — кроме того, инструмент зачастую выдает результаты, противоречащие базовым принципам программирования на выбранном языке.
2. **Поддержка.** ИИ может создавать избыточно сложный код, использовать нераспространенные подходы и функции — и работать у каждого разработчика по своему. Синхронизировать выполнение задач по какому-то лекалу не получится, рефракторинг кода от ИИ тоже бывает очень непрост. Так что для совместной работы это не лучшее решение
3. **Бесконечные доработки.** ИИ обучают на синтаксически правильных элементах. Поэтому, хотя сгенерированный код практически всегда соответствует стандартам, он не обязательно гарантирует оптимальное быстродействие или удобство поддержки.
4. **Потеря контроля.** Некоторые разработчики обеспокоены тем, что слишком сильная зависимость от помощников ИИ может снизить их навыки и опыт

программирования. Однако ключ заключается в том, чтобы рассматривать ИИ как ценный инструмент, а не замену человеческого суждения и критического мышления.

Заключение

Интеграция ИИ в создание IT-продуктов — это значительный шаг на этапе прототипирования и разработки. ИИ-инструменты имеют способность понимать потребности потребителей, рыночные тенденции, постоянно развивающийся технологический ландшафт и адаптироваться к нему.

Предоставлено: Проф. Олегом Фиговским