Обзорная статья
УДК 004.89
DOI: 10.14529/ctcr250402

ЭВОЛЮЦИЯ ИСПОЛЬЗОВАНИЯ И ПРИМЕНЕНИЯ МАШИННОГО ОБУЧЕНИЯ И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В РАЗРАБОТКЕ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ И В СИСТЕМАХ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ

***А.А. Шинкарев**, sania.kill@mail.ru*
***М.В. Ядрышникова**, reeyardma@gmail.com*
***О.В. Логиновский**, loginovskiiov@susu.ru, https://orcid.org/0000-0003-3582-2795*
***С.А. Лазарева**, lazarevas124@gmail.com*
***В.М. Губин**, teacfoou@gmail.com*
Южно-Уральский государственный университет, Челябинск, Россия

Аннотация. В настоящее время применение больших языковых моделей (large language model, далее – LLM) в корпоративных информационных системах и системах поддержки принятия решений становится все более распространенной практикой. **Цель исследования:** рассмотреть последовательное развитие технологий, использующихся в корпоративных информационных системах, от математических моделей до нейронных сетей, включая большие языковые модели, а также выдвинуть рекомендации по применению LLM в корпоративных системах и предположить, каковы будущие тенденции и риски применения алгоритмов искусственного интеллекта в таких системах. **Методы и материалы.** Исследование основано на ретроспективном анализе алгоритмов искусственного интеллекта. Рассмотрено несколько вариантов использования больших языковых моделей в корпоративных системах и системах поддержки принятия решений и проведен их сравнительный анализ по ряду критериев, выявленных на основе изученных существующих исследований. **Результаты.** Приведено несколько рекомендаций лучших практик по применению рассмотренных подходов к применению LLM в корпоративных системах. Также описаны преимущества и недостатки нового подхода к разработке программного обеспечения с применением LLM, где человек выступает в роли архитектора и планировщика задач, а LLM – в роли их исполнителя. Рассмотрены передовые технологии, ядром которых являются большие языковые модели – агентный искусственный интеллект, позволяющий моделям взаимодействовать со внешним миром посредством инструментов и выполнять разноплановые задачи подобно человеку. Приведены предположения о будущих тенденциях в применении искусственного интеллекта в целом и LLM в частности в корпоративных информационных системах и системах поддержки принятия решений и рисках, связанных с дальнейшим развитием этих технологий. **Заключение.** Полученные результаты могут быть использованы в качестве основы для принятия решений управленцами в отношении целесообразности применения рассмотренных в статье методов и связанных с ними рисков при построении корпоративных информационных систем.

Ключевые слова: поддержка принятия решений, корпоративные информационные системы, машинное обучение, искусственный интеллект, большие языковые модели, агентные системы, вайбкодинг

Для цитирования: Эволюция использования и применения машинного обучения и искусственного интеллекта в разработке корпоративных информационных систем и в системах поддержки принятия решений / А.А. Шинкарев, М.В. Ядрышникова, О.В. Логиновский и др. // Вестник ЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». 2025. Т. 25, № 4. С. 17–42. DOI: 10.14529/ctcr250402

Review article
DOI: 10.14529/ctcr250402

EVOLUTION OF USAGE AND APPLICATION OF MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE IN THE DEVELOPMENT OF ENTERPRISE INFORMATION SYSTEMS AND IN DECISION SUPPORT SYSTEMS

***A.A. Shinkarev**, sania.kill@mail.ru*
***M.V. Yadryshnikova**, reeyardma@gmail.com*
***O.V. Loginovskiy**, loginovskiiov@susu.ru, https://orcid.org/0000-0003-3582-2795*
***S.A. Lazareva**, lazarevas124@gmail.com*
***V.M. Gubin**, teacfoou@gmail.com*
South Ural State University, Chelyabinsk, Russia

Abstract. Usage of large language models (LLMs) in enterprise information systems has been growing more common in recent years. **The purpose of the study** is to consider the evolution of technologies used in enterprise information systems from mathematical models to neural networks, including large language models, as well as suggest recommendations on the application of LLMs in enterprise systems and assume future tendencies and risks of using artificial intelligence (AI) in such systems. **Methods and materials.** A retrospective method was used to analyze the development of artificial intelligence algorithms. Several approaches to using LLMs in enterprise systems and decision support systems were considered and compared by several criteria identified based on the existing studies. **Results.** The conducted study includes several recommendations on the best practices of applying LLM-based approaches to enterprise systems. It also covers the advantages and disadvantages of a new LLM-based programming approach, where a person acts as a system architect, while a model performs the technical tasks. The study describes the most recent advanced technology, agentic AI, which allows large language models to interact with their external environments and perform diverse tasks using various tools. The study also includes assumptions about future tendencies of AI usage in enterprise information systems and the corresponding risks. **Conclusion.** The results of this study can be used as a base for managers' decision making regarding the feasibility of using LLM-based methods considered in the study and their corresponding risks when building enterprise information systems.

Keywords: decision support, enterprise information systems, machine learning, artificial intelligence, large language models, agent-based systems, vibe coding

For citation: Shinkarev A.A., Iadryshnikova M.V., Loginovskiy O.V., Lazareva S.A., Gubin V.M. Evolution of usage and application of machine learning and artificial intelligence in the development of enterprise information systems and in decision support systems. *Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics*. 2025;25(4):17–42. (In Russ.) DOI: 10.14529/ctcr250402

Введение

В современном мире востребованным как никогда становится умение перестроиться под радикально меняющийся информационный ландшафт. Еще буквально вчера индустрия корпоративных информационных систем была одержима цифровыми двойниками, большими данными и цифровой трансформацией, а уже сегодня мир втягивается в новую гонку – гонку искусственного интеллекта. И этот искусственный интеллект пронзает все сферы человеческой жизни, становясь незаменимым помощником в принятии повседневных и рабочих решений.

При этом он же создает риски для человечества, становясь опасным конкурентом на рынке труда. А разработчики программного обеспечения в общем и инженеры машинного обучения в частности лишь приближают увлекательное, но для многих и пугающее будущее. Будущее, где человек должен освободиться от рутины монотонной работы и заняться более интересными, креативными и высокоуровневыми задачами. Однако не ясно, насколько радужными окажутся эти перспективы и насколько глубокой – выкопанная самим себе яма. Ведь креативной работы на всех может и не хватить, а искусственный интеллект со временем вполне сможет взять на себя и креатив.

Креативной и творческой деятельностью можно считать и сам процесс разработки программного обеспечения, а не только верхнеуровневое принятие управленческих решений. Споры о том, является программирование творчеством или ремеслом, не утихают, а теперь разгораются вновь с приходом агентного искусственного интеллекта. Но сегодня вопрос заключается уже не в том, творчество ли это, а в том, когда и какую часть труда разработчиков, тестировщиков, дизайнеров и других направлений в ИТ заменит искусственный интеллект. Результаты, которые показывают такие экстремальные подходы к разработке программных систем, как вайб-кодинг, подчас ошеломляют, создание прототипов систем и пилотных версий приложений становится доступно каждому, у кого есть продуктовая идея для стартапа, и занимает часы труда этого человека, у которого может и не быть специализированного инженерного образования. Эта пьянящая простота таит в себе большую опасность взрывного, неконтролируемого роста сложности поддержки системы, разработанной искусственным интеллектом, а также проблем с безопасностью и сложностью поиска и исправления ошибок в бизнес-логике. Как и в диалектике, где присутствует тезис – переход к быстрому программированию роботом без человека, антитезис – риски низкого качества, проблем с безопасностью, быстрого роста сложности неприемлемы и синтезис – поиск баланса между тем, к чему мы привыкли и тем, к чему стремится вся индустрия информационных технологий.

Прежде чем рассматривать изменения в процессе разработки программного обеспечения, начнем с более детального изучения развития информационных систем поддержки принятия решений в разрезе развития их технологического базиса, начиная от первых систем и заканчивая агентным искусственным интеллектом.

1. Системы поддержки принятия решений

Принятие решений – процесс идентификации альтернатив и выбора среди них подходящего исходя из ценностей и предпочтений принимающего решение, причем зачастую необходимо делать выбор в условиях неопределенности. Современному управленцу приходится регулярно сталкиваться с необходимостью анализа сложных ситуаций, в которых когнитивные искажения лица, принимающего решения, могут снижать качество выбора, особенно в условиях ограниченности времени, стрессовых ситуаций и большого объема разнородной информации.

В такой распространенной ситуации кстати приходятся системы поддержки принятия решения (Decision Support Systems, далее – DSS). DSS – это особый класс компьютеризированных информационных систем, направленный на содействие в принятии решений, чаще всего в рамках организаций. DSS предназначена для того, чтобы помочь лицам, принимающим решения, агрегировать полезную информацию из сырых необработанных данных, бизнес-моделей компании и даже из личного опыта руководителя, что в совокупности позволяет эффективнее выявлять и решать проблемы и принимать правильные решения [1].

Использование DSS в корпоративной среде, где ошибочное решение может привести к серьезным финансовым и репутационным потерям для целой организации, позволяет снизить риски, связанные с неопределенностью, за счет моделирования ситуаций, а также благодаря выявлению скрытых, но существенных для бизнеса связей, существующих между событиями и ключевыми метриками предприятия. DSS повышают адаптивность организаций к изменяющимся условиям ведения бизнеса и усиливают стратегическую устойчивость за счет дополнения (Augmentation) возможностей человека впечатляющим потенциалом современных информационных технологий и вычислительных мощностей. Для российских реалий ведения бизнеса в целом и принятия управленческих решений в частности характерен стиль ручного единоличного управления, при котором часто недостает подкрепления интуитивных решений руководителя и их обоснования. Внедрение современных и перспективных систем принятия решений позволит подступиться к решению этой проблемы, при этом не обременяя руководителя неудобным и неполноценным инструментом, а предоставляя ему мощный, нетребовательный к однородности данных инструмент анализа и поддержки принятия решений.

Основной фокус исследований в этой сфере – повышение эффективности как процесса принятия решений руководителем, так и самих решений. Изначально главная роль DSS заключалась в предоставлении пользователю сводной информации на основе статистики, собираемой из внутренней корпоративной базы данных. Алгоритмы, использовавшиеся в данных системах, не были

гибкими и, как правило, ограничивались четко заданными правилами и условиями. Такие DSS можно было использовать для формирования отчетности, статического анализа и простых решений с четко заданными условиями «если-то». По мере увеличения гибкости применяемых алгоритмов и совершенствования аналитических возможностей систем расширялись их прогностические способности, что позволяло решать более сложные задачи по моделированию, например, комплексное планирование производства и поставок. С добавлением интеграции внешних данных появилась возможность анализировать рынок, проводить онлайн-мониторинг и конкурентную разведку и учитывать полученные данные при принятии решений. С началом использования технологий машинного обучения объем задач, связанных с прогнозированием, аналитикой большого объема данных и их классификацией, значительно увеличился, а также стала возможной работа DSS в масштабе времени, близкого к реальному. Современные системы, использующие то, что сегодня называется искусственным интеллектом, способны рекомендовать стратегии дальнейшего развития на основе смоделированного поведения рынков, а в некоторых случаях – принимать решения автоматически.

Системы поддержки принятия решений стали неотъемлемым инструментом для многих компаний при построении стратегий, аналитики, планировании и в других управленческих процессах, поскольку успешное их внедрение позволяет организации минимизировать риски, своевременно находить новые точки роста в условиях быстро изменяющегося рынка. На текущий момент DSS продолжают активно развиваться в области машинного обучения и искусственного интеллекта. Однако такой рост обуславливает появление новых требований к компаниям: для сохранения конкурентоспособности необходимо оперативно, но при этом своевременно и разборчиво внедрять постоянно обновляющиеся технологии в архитектуру корпоративных информационных систем, корректировать организационные процессы и обучать персонал эффективному взаимодействию с информационными системами. При этом все чаще возникают вопросы о степени интеграции и применимости DSS, их самостоятельности и автономности в принятии решений, границах ответственности за принятые решения, глубине и характере влияния DSS на организационные структуры, корпоративную культуру и саму природу управленческих процессов. Для лучшего понимания места современных подходов к построению систем поддержки принятия решений рассмотрим в исторической ретроспективе развитие технологической и научной базы, на основе которой построены такие системы.

1.1. Первые системы поддержки принятия решений

Первые системы поддержки принятия решений появились в середине XX века и в основном относились к модельно-ориентированным системам поддержки принятия решений (Model-Oriented DSS) – системам, основанным на взаимодействии пользователя с различного рода моделями (финансовыми, статистическими, имитационными и пр.) [2]. Одним из ключевых подходов, применявшихся на этом этапе, было динамическое программирование (Dynamic Programming, далее – DP), предложенное Ричардом Беллманом в 1950-х [1]. Суть метода заключалась в разбиении крупной задачи на подзадачи и последовательном решении каждой из них с запоминанием промежуточных результатов – такая техника называлась мемоизацией (Memoization). На каждом этапе вычислялся наиболее оптимальный вариант следующего шага и запоминался. Динамическое программирование применялось в маршрутизации транспорта, управлении запасами, планировании производства и других подобных задачах оптимизации. Однако данный подход требовал четко формализованной логики и структуры задачи и построения заранее известной модели среды, в которой он будет применен. Система не выявляла скрытые закономерности: она лишь вычисляла оптимальную последовательность действий в рамках строго заданных правил и параметров на основе уже прописанных связей. Поэтому метод позволял распределять известное количество ресурсов по известным критериям, но был неспособен предсказать появление новых критериев или изменение существующих. Например, в задаче маршрутизации система на основе динамического программирования смогла бы учесть влияние пробок на стоимость маршрута только при наличии заранее прописанной функции, связывающей интенсивность пробок со стоимостью маршрута. Система не знает, с какой вероятностью пробка возникает на этом маршруте, и если из-за аварии дорожная ситуация изменится на пути, для которого зависимость от интенсивности пробок не задана, то система может выбрать в качестве оптимального маршрута тот, который

далеко не является таковым на данный момент. При этом критерии требуется настраивать вручную, то есть если управляющий системой не знает однозначно, какая конкретно взаимосвязь между пробками и стоимостью маршрута, система не сможет выявить эту зависимость самостоятельно.

После закрепления методов модельно-ориентированного программирования в 1970–1980-х годах в поддержке принятия решений начали применять экспертные системы (Knowledge-Based Systems), которые использовали знания и правила (например, в формате «если-то») для анализа данных и предоставления рекомендаций и выводов. В отличие от модельно-ориентированных систем, оперировавших количественными параметрами, экспертные системы могли работать с критериями качественного и описательного характера, например, «тяжелое состояние» вместо более конкретного «температура 38,9» Это позволило применять данный подход в более слабоструктурированных задачах, то есть тех, для которых построить математическую модель сложно или даже практически невозможно. В качестве ответа экспертные системы предоставляли логический вывод или рекомендацию, имитируя оценку эксперта-человека. Благодаря этому данный вид систем применялся в медицине и диагностике, например, на нем в 1972 году была основана MYCIN [3] – медицинская система, использовавшая логические правила для рекомендаций по антибиотикам. Главным недостатком экспертных систем была проблема масштабируемости и поддержки в актуальном состоянии базы данных и правил, на основе которых осуществлялся поиск решения.

Следующим этапом развития DSS в 1980-х годах стали системы, управляемые данными (Data-Driven DSS), в которых основой работы стало взаимодействие с данными, накопленными компанией (как правило, представленными в виде временных рядов), а также данных извне, то есть произведенных не в рамках деятельности компании, а, например, описывающих ситуацию в мире [2]. Если ранее основой систем были математические модели или правила, а данные подставлялись лишь как переменные на вход, то, применяя подход, основанный на управлении данными, главным элементом стали именно они, а аналитические методы, применяемые к данным для выявления закономерностей, стали второстепенными. Ведь если данные есть, то они могут быть использованы любыми алгоритмами и системами отчетов для того, чтобы ответить на вопросы бизнеса сегодня. Важно, что ориентация на сбор всевозможных данных предоставляет возможность в будущем ответить на те вопросы, которые сегодня не удается предсказать. Наличие новых инструментов при отсутствии исторических данных во многих случаях делает их бесполезными.

Смещение фокуса на данные стало возможным, во-первых, благодаря распространению систем планирования ресурсов предприятия (Enterprise Resource Planning, далее – ERP) и управления взаимоотношениями с клиентами (Customer Resource Management, далее – CRM) в корпоративной среде, что увеличило накапливаемый компаниями объем данных, которые можно было использовать для помощи в принятии решений. Во-вторых, толчком к этому качественному переходу послужило активное развитие реляционных СУБД (систем управления базами данных) в 1970-х годах [4] и особенно выход Oracle [5] в 1979 году – первой коммерческой реляционной СУБД. Такой переход, обусловленный технической революцией, будет нам встречаться и в дальнейшем, ведь принципиально новая технологическая база открывает новые, ранее невиданные перспективы.

Гибкость и применимость систем, управляемых данными, стала значительно выше использовавшихся ранее DSS, ориентированных на модели и правила, поскольку пользователь мог варьировать и критерии анализа, и срезы данных, к которым он их применял. В качестве аналитических инструментов использовались различные методы агрегирования и статистики, а также технология интерактивной аналитической обработки данных (Online Analytical Processing, далее – OLAP). OLAP – технология высокоскоростного многомерного анализа больших объемов данных [6]. В данном случае многомерность данных означает возможность формирования среза данных по различным категориям и, соответственно, рассмотрение их по разным показателям. OLAP позволяет выполнять практически любые запросы к данным и всем их комбинациям, при этом работая с историческими данными, объем которых исчисляется терабайтами и даже петабайтами, в режимах массового импорта и обработки потоковых событий [7]. OLAP, однако, не позволял автоматически выявлять закономерности в данных, поэтому с его помощью DSS не могли предсказы-

вать на основе анализируемых данных дальнейшие тенденции. Предиктивные возможности систем поддержки принятия решений начали расширяться с появлением и развитием методов глубинного анализа данных (Data Mining) [8] – поиска и извлечения закономерностей из большого объема данных. Впоследствии концепция глубинного анализа данных и его использование в информационных системах послужили мостом для интеграции технологий искусственного интеллекта в корпоративную среду.

1.2. Начало применения искусственного интеллекта в системах поддержки принятия решений

В 1990-е годы накопленные, отработанные и уже стандартизированные подходы хранения, обработки и анализа данных объединились в концепцию бизнес-интеллекта (Business Intelligence, далее – BI) – набора технологических процессов для сбора, управления и анализа данных организации с целью получения информации, способной помочь в принятии решений касательно стратегии и операционной деятельности компании [9]. Первоначально основными функциями BI были обработка, хранение и визуализация корпоративных данных, но он был ограничен ретроспективным взглядом на бизнес-процессы, что не позволяло предсказывать последующие тенденции. То есть можно было с разных точек зрения увидеть и проанализировать, как выстраивалась деятельность организации до текущего момента, но на основе этого анализа не выявлялись закономерности, на которые можно было бы опираться при планировании будущей деятельности. На этом этапе и возникла потребность в интеграции глубинного анализа данных, основная часть алгоритмов которого на тот момент основывалась на статистике и логике. В глубинном анализе впервые применялись алгоритмы машинного обучения (Machine Learning, далее – ML), такие как деревья решений, метод поиска ближайших соседей и нейронные сети, но пока что они использовались только в комбинации с традиционной аналитикой, причем преобладала именно последняя.

В середине 1990-х в связи с постоянным увеличением объемов цифровых данных потребность в новом поколении инструментов для извлечения ценной информации из этих данных становилась все острее [10]. При этом обработка таких объемов наконец становилась возможной в связи с увеличением вычислительных мощностей устройств. В то же время активно развивалось машинное обучение: появились метод опорных векторов (Support Vector Machine, SVM) и ансамблевые методы – бэггинг (Bagging), бустинг (Boosting), метод случайных лесов (Random Forest), – применение которых значительно повлияло на рост точности используемых ML-решений. Машинное обучение показывало все большую эффективность в бизнес-задачах и постепенно его алгоритмы стали основой глубинного анализа данных, вытесняя статистические методы и реализуя идею предиктивной аналитики внутри систем бизнес-интеллекта [11].

Теперь BI-системы могли решать задачи прогнозирования спроса, сегментации клиентов (т. е. предсказания принадлежности клиента к категории на основе некоторых критериев), оценивать риски и пр. – помощь бизнес-интеллекта в принятии решений в корпоративной среде становилась все эффективнее. Если на этапе зарождения концепции BI бизнес-интеллект и системы поддержки принятия решений разделяли, относя первое в большей степени к информационным системам, то на этом этапе, к началу 2000-х годов, BI-платформы уже помогали выявлять скрытые закономерности и предлагать варианты действий и стратегии на будущее на основе имеющихся данных, а не только представлять и визуализировать их для отчетности, что сделало границу между этими двумя видами систем практически незаметной [12].

2. Развитие алгоритмов искусственного интеллекта

Машинное обучение, ставшее ядром глубинного анализа данных и, как следствие, двигателем предиктивных возможностей бизнес-интеллекта, закрепилось как инструмент аналитики в корпоративном мире, что обозначило последующее внедрение различных отраслей искусственного интеллекта как ключевую точку роста для дальнейшего развития систем принятия решений, управляемых данными, которые на тот момент уже показали и доказали свою ценность для бизнеса [13]. Рассмотрим детальнее развитие одного из видов алгоритмов искусственного интеллекта – нейронных сетей, которые во многом определяют машинное обучение сегодня.

2.1. Появление и развитие нейронных сетей

Нейронные сети, являющиеся видом алгоритмов машинного обучения, способны выучивать нелинейные закономерности во входных данных. Первые попытки применения нейронных сетей в корпоративной среде относятся к 90-м годам XX века, когда алгоритмы машинного обучения начали использоваться в BI-системах. Появление метода обратного распространения ошибки в районе 1986 года послужило стартом активного использования нейронных сетей в коммерческих задачах, поскольку данный метод позволял кодировать во внутренних слоях нейронной сети признаки, релевантные доменной области решаемой задачи, и описывать закономерности во входных данных в связях между слоями, что было недоступно при использовании более ранних методов обучения, таких как метод коррекции ошибки [14]. На тот момент в сравнении с настоящим временем возможности вычислительной техники были ограничены [15], а также присутствовал дефицит цифровых данных для обучения, ограничивающий применение нейронных сетей [16]. Ввиду этих факторов первые попытки применения нейронных сетей имели экспериментальный характер. Одной из первых сфер, нашедшей применение нейронных сетей, стал финансовый сектор, где использовались многослойные перцептроны, представляющие собой полносвязные нейронные сети, для задачи оценки кредитных рисков [17]. Появление к концу XX века сверточных нейронных сетей позволило автоматизировать распознавание рукописных символов в банковских чеках, однако потенциал внедрения подобных систем по-прежнему был ограничен дефицитом как данных для обучения, так и вычислительных мощностей [16].

В XXI веке применение нейронных сетей в корпоративной среде стало шире. Одной из основных причин их распространения помимо роста объема данных, доступных для обучения, стало увеличение возможностей вычислительных устройств, в первую очередь за счет применения графических процессоров [18]. В финансовом секторе нейронные сети стали основой систем мониторинга мошеннических транзакций [19], в розничной торговле и электронной коммерции появились нейросетевые системы прогнозирования спроса на товары [20], оптимизации ценообразования [21] и создания персонализированных рекомендаций товаров и услуг [22].

Кроме того, в XXI веке шло активное развитие моделей для обработки естественного языка (Natural Language Processing, далее – NLP) [23, 24]. Основными задачами, решаемыми при помощи нейронных сетей, были анализ эмоционального тона отзывов клиентов, классификация и маршрутизация обращений в службах поддержки, и зачастую лучшее решение каждой из задач предполагало уникальную, специфическую для типа задачи архитектуру нейронной сети. Эти архитектуры в основном представляли собой совмещение сверточных и рекуррентных слоев. Однако, несмотря на активное применение, у всех использовавшихся моделей присутствовало архитектурное ограничение, заключающееся в том, что они слабо улавливали семантические зависимости в тексте и не обладали способностью к глубокому пониманию контекста, не имели когнитивных способностей. Поиск новых архитектур, устраняющих эти недостатки, привел к появлению архитектуры трансформера в 2017 году [25]. Это событие обозначило начало новой эры в развитии искусственного интеллекта.

2.2. Появление архитектуры трансформера

Архитектура трансформера, представленная для решения задачи машинного перевода в 2017 году [25], имела два основных преимущества перед предшественниками. Первое и основное заключается в способности строить взаимосвязи между каждой парой элементов во всей входной последовательности. Значение, определяющее взаимосвязь между двумя элементами входной последовательности, называется оценкой внимания.

Вторым преимуществом трансформера является возможность параллелизации его обучения, так как оценки внимания можно вычислять для каждой пары элементов входной последовательности независимо. Параллелизации обучения нельзя было достичь, используя рекуррентные архитектуры, поскольку в них обработка входной последовательности происходит пошагово. Это позволяет обучать трансформер со схожим числом параметров быстрее рекуррентной сети, задействуя при этом больше вычислительных мощностей.

В 2019 году был представлен BERT (Bidirectional Encoder Representations from Transformers) [26] – модель, унаследовавшая архитектуру трансформера, предобученная для глубокого семантического представления текста и показавшая лучшие на тот момент метрики качества на широ-

ком круге задач. Преимущество модели в том, что она может быть адаптирована под задачи обработки естественного языка путем добавления всего одного дополнительного слоя.

Идеи BERT развивались научным сообществом, появлялись более совершенные варианты архитектуры [27, 28], имеющие повышенные способности к глубокому семантическому пониманию языка. В корпоративных системах модели, имеющие архитектуру, основанную на BERT, активно применялись для задачи распознавания намерений пользователя в диалогах чат-ботов, коррекции ошибок правописания [29], предсказания банкротства [30]. Стоит отметить, что в силу квадратичной сложности механизма внимания и ограничений по памяти графических ускорителей число параметров в данных моделях чаще всего составляло до одного миллиарда, в то время как число параметров в современных крупнейших языковых моделях превышает триллион.

Модели семейства BERT решают задачи маскированного языкового моделирования (Masked Language Modeling, далее – MLM) и задачи оценки взаимосвязи между двумя предложениями (Next Sentence Prediction, далее – NSP). Задача MLM заключается в предсказании корректного токена (Token) для заданной позиции во входной последовательности. В качестве примера задачи в такой постановке, приравнивая для наглядности токен к целому слову, можно привести предложение «Катится колобок по дороге, а навстречу (пропуск) заяц» – модель учится предсказывать, какое слово должно стоять на месте пропуска. Задача NSP заключается в предсказании по двум подряд идущим последовательностям, являются ли они семантически связанными, либо не являются. Иными словами, могли ли в исходном тексте две последовательности идти друг за другом. Допустим, предложения «Катится колобок по дороге» и «А навстречу ему заяц» могут быть связаны по смыслу и быть написанными друг за другом, в то время как предложения «Я тебя съем» и «А навстречу ему заяц» – нет.

При решении задачи MLM модели семейства BERT вычисляют оценки внимания для всех элементов входной последовательности, находящихся до и после целевого элемента, это обуславливает двунаправленность данных моделей. Параллельно с таким подходом к решению задач MLM развивался альтернативный, характеризующийся вычислением оценок внимания только для элементов, находящихся перед целевым. Данный подход реализован в архитектуре генеративного предобученного трансформера (Generative Pre-trained Transformer) и называется задачей предсказания следующего токена (Next Token Prediction, далее – NTP) [31]. Модели, решающие задачу NTP, оказались более способными к генерации текста. Масштабирование вычислительных мощностей, объема обучающих данных и числа параметров в таких моделях привело к появлению в 2019 году моделей, способных генерировать текст, неотличимый от человеческого [32, 33]. На сегодняшний день данный класс моделей называется большими языковыми моделями (Large Language Models, далее – LLM).

3. Современные варианты реализации систем поддержки принятия решений в корпоративной среде

Появление и развитие архитектуры трансформеров, ставшее ключевой точкой в эволюции качества языковых моделей, повлекло за собой широкое внедрение LLM в корпоративную среду. Стоит упомянуть, что в данной статье не рассматривается реализация программных систем с применением LLM, доступных как сервис, то есть так называемых SaaS-решений (Software as a Service), к которым, например, относится ChatGPT. Везде в дальнейшем по умолчанию речь идет о LLM, развернутых самостоятельно в закрытом или контролируемом сетевом контуре, только такой вариант может гарантировать отсутствие утечки конфиденциальных сведений. Однако при возможности и желании все подходы, описанные в статье, применимы и с SaaS-решениями.

На сегодняшний день существует несколько вариантов того, как именно организация будет интегрировать LLM в свои процессы принятия решений и разнообразные более низкоуровневые рабочие процессы, которые в общем виде тоже можно считать процессами принятия решений.

3.1. Большие языковые модели

Первый подход – использование базовых, предобученных LLM, то есть моделей, обученных на огромных наборах данных, включающих разные сферы, языки, лексику различной специфики и прочую вариативность. Такой подход начали использовать в бизнесе в 2020 году с выходом GPT-3

от OpenAI. Модели в этом случае применяются для генерации контента, а также в работе с текстом, в основном оказывая значимое влияние на процессы взаимодействия компаний с клиентами.

LLM способны анализировать большие объемы данных, например, отзывы клиентов, выявляя их отношение к продукту и определяя потенциальные точки роста, а также подводить итоги по отчетам существующих проектов компании и предлагать на этой основе варианты стратегий последующего развития [34]. Результаты такого анализа помогают руководству принимать решения по дальнейшей работе: какие проекты стоит приоритизировать, какие – остановить, кто составляет основную аудиторию продукта, соответствует ли она целевой, и это только примеры с поверхности, на практике их куда больше.

Генеративные возможности LLM используются в планировании: модели способны создавать индивидуальные расписания проектов, используя условия по времени, бюджету, количеству участников и прочие инструкции, которые пользователи моделей могут задавать в промтах – текстовых запросах, в которых ставятся задачи для моделей [34]. Генерация также применяется бизнесами в написании информационных и рекламных публикаций для социальных сетей, составлении описаний продуктов и других задачах, имеющих отношение к копирайтингу.

Аналитика и генерация часто работают в комбинации. Сначала через промты модель получает информацию о деятельности компании, ее текущих проектах, о том, какую аудиторию она стремится сформировать и какую привлекает на данный момент. Все эти данные модель подытоживает, извлекая ключевые моменты, и учитывая этот контекст, может более качественно работать над планированием проектов и написанием текстов, придерживаясь текущих приоритетов и сохраняя фирменный стиль организации либо корректируя их в соответствии с заданными в промте целями компании.

LLM достигли впечатляющей связности и естественности речи в своих ответах, однако в своем базовом варианте их способность к работе, адаптированной под нужды конкретного пользователя или организации, ограничена [35]. Широкий спектр данных, на которых обучаются эти модели, ведет к их склонности давать слишком обобщенные ответы.

Для персонализации ответов моделей требуется уделять большее внимание промтам и тем корпоративным данным, которые отправляются модели на вход. Даже небольшое изменение промта может значительно повлиять на генерируемый моделью ответ [35]. Это повышает порог входа для сотрудников организации, которые хотели бы воспользоваться системой для взаимодействия с корпоративной базой знаний: им нужно правильно сформулировать запрос при неполной уверенности в том, что именно они ищут. А в контексте систем CRM-формата чата поддержки с ответами на вопросы необходимость точного промта создает еще больше неудобств: вопрос клиента вряд ли будет представлять собой достаточно точный для модели контекст, содержащий всю нужную информацию. По мере развития LLM и укрепления их в индустрии как инструмента корпоративной среды вопрос улучшения качества персонализации ответов стал одним из наиболее актуальных.

Кроме того, базовые LLM наиболее склонны к «галлюцинациям» среди всех основанных на LLM подходах. Галлюцинациями называется феномен генерации правдоподобного и связного текста, по существу являющегося фактически неверным, бессмысленным или не соответствующим контексту [36, 37]. Это явление считается одним из главных недостатков LLM, и борьба с галлюцинациями моделей является одним из самых актуальных направлений исследования. Ведь при применении моделей в бизнесе и ежедневной работе неверный или даже заведомо ложный, но связный на вид, совет может серьезно навредить результатам и тем самым поставить под сомнение целесообразность использования таких «советчиков». Сегодня, чтобы отловить ложные сведения от LLM, необходимо быть экспертом в области, в которой советуемся с ней, а это не всегда возможно. Например, отдав на суммаризацию документ с техническим заданием на разработку сайта объемом 150 страниц, можно получить совершенно бредовые тезисы, про требования к языку программирования C++, о котором в исходном документе нет ни слова. До сих пор нет надежных инструментов верификации подлинности ответов LLM, эти риски сейчас каждый берет на себя и должен их как минимум осознавать и учитывать.

Варианты реализации систем поддержки принятия решений, рассматриваемые далее, построены на основе LLM и разработаны с целью исправления недостатков базовых больших языковых моделей с основным фокусом на улучшение персонализации их ответов.

3.2. Дообученные большие языковые модели

Тонкая настройка, или дообучение (Fine-Tuning), – процесс изменения параметров модели с целью достижения большего соответствия ее ответов конкретной задаче или предметной области. Цель такого подхода в том, чтобы дать модели больше дополнительных данных, чем может поместиться в промт, и дать ей возможность не просто получать доступ к данным, а извлекать из них новые паттерны и запоминать их без потери ранее полученных знаний [34]. Тонкая настройка полагается на предобученные модели, предоставляющие основную базу знаний, которая в ходе дообучения корректируется под конкретные сценарии использования. В отличие от предобучения, тонкая настройка часто использует более конфиденциальные данные, поскольку проводится на данных определенной организации, а также в зависимости от целей дообучения и доступных ресурсов применяет специфические подходы для оптимизации скорости и снижения стоимости процесса: от полного дообучения (Full Fine-Tuning, FFT) до применения адаптеров – дополнительных слоев, обучаемых под конкретные задачи [38].

Дообучение LLM помогает в персонализации ответов модели. Оно делает модель менее требовательной к наличию длинных, детализированных промтов и повышает качество предиктивных способностей LLM в специфичных для конкретной области задачах. Дообученная модель обладает теми же возможностями и применяется для решения тех же задач, что и базовая LLM, но способна работать в рамках этих активностей более направленно, с меньшим участием человека за счет меньшей потребности в объемном промте. В контексте бизнеса дообучение позволяет моделям учитывать корпоративную терминологию, особенности фирменного стиля компании, ее историю и другие аспекты, влияющие на процесс принятия решений.

Однако, чтобы персонализировать ответы модели с помощью дообучения, компании необходимо собрать и подготовить для этого собственные данные. Причем, во-первых, имеет значение количество данных. Хотя больший объем данных обычно означает повышение качества ответов модели, особенно с точки зрения персонализации, сбор этих данных ведет к увеличению сроков и раздуванию бюджета проекта. Компании требуется заранее оценить необходимый объем данных для достижения некоторого порога качества ответов модели (способы оценки этого качества рассматриваются в дальнейших разделах), чтобы не собирать, допустим, 10 000 примеров взаимодействий в формате вопрос-ответ с клиентами, когда того же порога можно было бы достичь, дообучив модель на 5000 сообщений. Либо компания может решить, что требуемые инвестиции не соответствуют ее возможностям, и сделать выбор в пользу персонализации с помощью, например, промтов. Во-вторых, данные должны быть достаточно качественными, чтобы тонкая настройка действительно имела смысл и была эффективной. Данные с большим количеством шума, пустых или нулевых значений или слишком однообразные, то есть не отражающие разнообразие информации в домене дообучения, могут только ухудшить результаты работы модели. В-третьих, если дообучение проводится по методу дообучения с учителем (Supervised Fine-Tuning, SFT), то данные требуют разметки, то есть ручной классификации и маркировки. Формирование наборов данных для дообучения моделей – отдельное направление исследований [39], это процесс ресурсо- и трудоемкий, времязатратный и требующий экспертного подхода, что необходимо учитывать, выбирая подход тонкой настройки как альтернативу базовым LLM.

Кроме того, дообучение LLM на личных или корпоративных данных сопряжено с определенными угрозами безопасности [38]. К ним относятся, например, атаки на определение принадлежности (Membership Inference), когда злоумышленник хочет узнать о вхождении конкретного примера в тренировочный набор, или атаки извлечения данных, позволяющих получить конфиденциальные, частные сведения на основе их представления внутри модели или ее ответов. Также существуют бекдор-атаки (Backdoor), цель которых – внедрить уязвимости или вредоносные паттерны поведения в модель, манипулируя процессом дообучения. И это лишь несколько видов угроз, сопряженных с процессом дообучения. Разумеется, есть и методы защиты: анонимизация данных, дифференциальная приватность, федеративное обучение и другие – однако у них есть свои ограничения, во многом связанные с многообразием как способов дообучения [40], так и видов угроз и соответствующей невозможностью полностью унифицировать и стандартизировать процесс защиты конфиденциальности при работе с дообученными LLM.

На сегодняшний день не ясно, есть ли надежный способ безопасной работы с конфиденциальными данными при необходимости разделения прав доступа. Например, если модель дообу-

чалась на данных, которые содержат сведения о зарплатах сотрудников, то велика вероятность, что сотрудники, для которых доступ к этим сведениям закрыт, смогут взломать модель, чтобы получить эти сведения. У LLM архитектурно отсутствуют механизмы, позволяющие обеспечить абсолютное и надежное разграничение прав доступа к данным, на которых она обучалась.

Дообучение может требовать значительных вычислительных ресурсов [35]. Для снижения этих требований применяют методы параметрически эффективной тонкой настройки (Parameter-Efficient Fine-Tuning, далее – PEFT), суть которых заключается в уменьшении количества параметров, настраиваемых при дообучении. Использование PEFT делает дообучение намного доступнее, но, во-первых, требует неоднократных экспериментов, чтобы найти баланс между экономией ресурсов и сохранением качества модели, а во-вторых, в зависимости от базовой модели и параметров дообучения по-прежнему зачастую может быть проведено не на любом устройстве: даже с уменьшенным количеством обучаемых параметров. Дообучение чаще всего требует применения графических ускорителей. Требования к оборудованию делают тонкую настройку LLM потенциально дорогим процессом для компании.

Масштабирование системы на основе дообученной модели и поддержание ее в актуальном состоянии также влечет за собой некоторые трудности [35]: появление новых данных означает необходимость вновь проводить тонкую настройку. Учитывая темпы увеличения объема данных в современном мире, компании может быть нелегко адаптировать свою инфраструктуру так, чтобы динамически обновлять LLM, особенно при нехватке достаточных вычислительных мощностей.

Так, тонкая настройка дает возможность обойти ограничения базовых LLM, связанные с персонализацией и нехваткой узкоспециализированных знаний у модели, позволяя учесть особенности сферы деятельности и базу знаний конкретной компании. Однако преимущества этого подхода неразрывно связаны с потенциальными трудностями его внедрения, включая требования к качеству и количеству данных, проблему сохранения их безопасности, необходимость регулярного обновления системы и значительные вычислительные затраты.

3.3. Генерация с дополненной выборкой

Концепция генерации с дополненной выборкой (Retrieval-Augmented Generation, далее – RAG) впервые была представлена в исследовании, опубликованном в 2021 году [41], и была, как и тонкая настройка, ориентирована на решение проблемы недостатка знаний базовых LLM в узких сферах. Однако в отличие от дообучения в этом подходе данные, использующиеся для улучшения качества работы модели в конкретном домене, существуют отдельно от LLM, во внешней базе данных.

На рис. 1. представлена обобщенная архитектура RAG-системы. Модуль поиска, получая запрос, извлекает из векторной базы данных, содержащей данные по соответствующему запросу домену, данные, наиболее актуальные запросу, – извлеченные данные называют контекстом. Затем система объединяет полученный контекст со своим исходным промтом, и LLM-модель, находящаяся в сердце RAG-системы, генерирует на этой основе ответ. Считается, что исходная предобученная модель в этом случае – параметрическая память RAG-системы, а база данных (как правило, векторная) – непараметрическая память [41]. Данные в векторную базу помещаются после деления на кусочки или же чанки (Chunking) и последующего их преобразования в векторы с помощью модели векторизации (Embedding, далее – эмбеддинг). За извлечение контекста из базы в свою очередь отвечает модуль поиска, также называемый ретривером (Retriever).

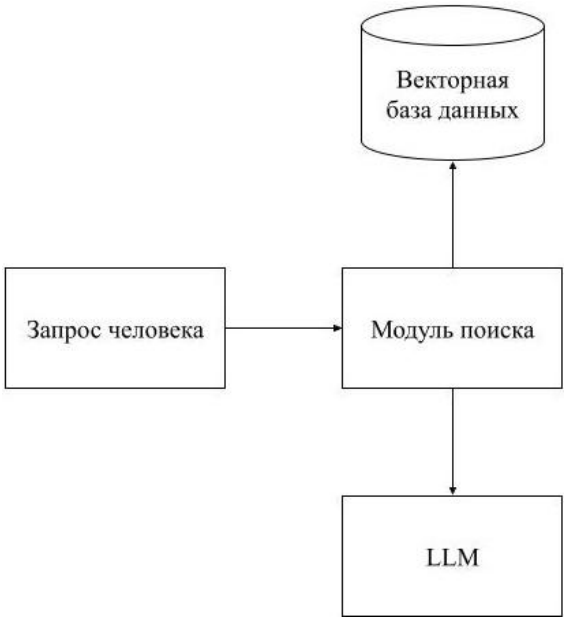

Рис. 1. Обобщенная архитектура RAG-системы
Fig. 1. Generalized architecture of the RAG system

Эта концепция решает проблему обновления и масштабирования данных, свойственную подходу с дообучением, поскольку при любых изменениях достаточно обновить только содержимое внешней векторной базы данных – нет необходимости заново проводить дообучение, что снижает затраты и ускоряет внедрение новых знаний. При этом достигается аналогичный результат – ответы модели больше соответствуют нуждам конкретного пользователя или компании, чем в случае с базовыми LLM, которые ничего не знают о специфике организации, ведь они не обучались на закрытых данных компании, только на публичных данных.

RAG также снижает склонность LLM к галлюцинациям за счет своей непараметрической памяти [35]. В зависимости от системного промта модели может быть позволено больше или меньше свободы действий: она может как самостоятельно рассуждать, чтобы принимать решения даже при наличии неоднозначности в извлеченном контексте или противоречий между найденными фактами, так и быть полностью ограниченной в предположениях на основе своей параметрической памяти и контекста, полученного на этапе извлечения данных из векторной базы.

В RAG-системе может быть реализовано разграничение доступа пользователей к данным. В случае тонкой настройки данные «вшиваются» в модель, то есть каждая дообученная LLM имеет доступ ко всей информации, на которой она дообучалась, что создает угрозу извлечения этих данных злоумышленниками. В случае RAG-системы модель может оставаться универсальной, а регулировать доступ к данным можно на уровне внешнего хранилища. Тогда в рамках одной организации сотрудники могут взаимодействовать с одной и той же последовательностью процессов (Pipeline, далее – пайплайн), но иметь доступ к разным сегментам базы данных, что повлияет на извлекаемые модулем поиска части релевантного контекста. Возможность строго и детерминированно ограничить доступ к информации является ключевым функциональным преимуществом RAG-систем в корпоративной среде, где ведется работа с конфиденциальной информацией.

В случае RAG увеличение затрат по сравнению с базовой LLM меньше, чем в случае тонкой настройки [35]. Конечно, использование и поддержание внешней базы данных, модели векторизации и механизма извлечения данных из базы также требует вычислительных ресурсов, экспертов и времени, однако эти процессы, как правило, менее требовательны, чем процесс дообучения LLM. Во многом реализация RAG-системы сводится к решению инженерной задачи проектирования и реализации информационной системы с использованием стандартных технологий и архитектурных паттернов, в то время как дообучение LLM является задачей из области машинного обучения или науки о данных, где результат требует исследований и экспериментов и сам результат не гарантирован и сложно предсказуем.

При использовании RAG подхода могут возникать трудности в ситуациях, когда объединенная длина информации из базы и исходного промта превышает допустимую длину контекста модели [35]. В таких случаях приходится прибегать к сокращению подаваемой модели информации, что может негативно сказываться на результатах ее работы, поскольку в контексте может не оказаться фактов, необходимых для правильного вывода. Подобный риск также возникает по причине того, что ретривер находит в базе данных множество подходящих чанков, но в контекст попадает только несколько из них с наибольшим коэффициентом сходства с запросом [42]. Соответственно, возможно, что найдено девять чанков, из них взято три с лучшим коэффициентом сходства, допустим, 0,93; 0,92; 0,91, а четвертый, с коэффициентом 0,89, уже не попал в контекст, хоть и содержал важную информацию. Одним из возможных способов избежать потери ценных чанков является добавление метаданных (например, имя документа и порядковый номер чанка) при сохранении чанков в векторную базу, чтобы при поиске, например, точно просмотреть все чанки из конкретного документа. Этот метод также поможет в случае неправильного нарезания чанков, из-за которого могут быть разорваны семантические связи в данных – наличие информации об имени исходного документа и порядке чанков в нем позволит восстановить целостность контекста.

Стоимость и срок внедрения RAG-системы ниже, чем у дообученных моделей, однако поскольку RAG в любом случае подразумевает последовательный процесс – извлечение данных и только после этого генерация, – это приводит к увеличенному времени задержки между запросом пользователя и ответом системы. Время обработки запроса пользователя также увеличивается по мере усложнения архитектуры пайплайна, добавления в него дополнительных этапов: предобра-

ботки данных, реранжирования чанков и др. [35]. Для снижения этой задержки применяют различные методы: предзаполнение семантического кеша часто повторяющимися вопросами, подбор техники индексации, которую использует векторная база данных, – однако универсального решения нет. Более того, по мере расширения базы данных зачастую практически невозможно не пожертвовать точностью поиска по базе, извлечения данных или генерации ответа в пользу снижения задержки. Снизить ее позволяет, например, поиск меньшего числа релевантных чанков, использование только поиска по векторам без учета метаданных, применение более простых моделей векторизации и LLM [43].

Многоступенчатость RAG-систем также затрудняет ее оценку и настройку, поскольку каждый из этапов пайплайна настраивается индивидуально, и различные конфигурации этапов могут выстраиваться в пайплайн в различных комбинациях и произвольной длины. Требуется проводить тестирование и оценку как всей системы целиком, так и каждого отдельного компонента, что делает процесс оценки качества RAG-системы многоплановым.

Стоит отметить, что тонкая настройка и RAG могут использоваться в комбинации. Наиболее распространенным случаем является использование дообученных LLM в качестве генеративного компонента пайплайна вместо их предобученных версий. Дообучение адаптирует «язык» модели под стилистическую специфику и терминологию конкретной области, а RAG позволяет поддерживать фактические данные в актуальном состоянии, динамически обновляя их. Кроме LLM дообучение в RAG-системах может применяться к моделям векторизации, использующимся и на этапе загрузки данных в векторную базу, и на этапе работы ретривера для преобразования запроса в вектор, на основе которого будет осуществляться поиск по сходству среди векторов в базе.

На текущий момент RAG является одним из самых активно развивающихся направлений как в научных исследованиях, так и в прикладной разработке. Сочетая в себе персонализацию ответов, возможности разграничения доступа к данным, своевременного обновления данных, возможности ссылаться на источники информации в ответах, снижение уровня галлюцинаций и затрат на процессы, связанные с обучением и дообучением моделей, этот подход становится стандартом внедрения LLM в корпоративной среде [44]. При этом внедрение RAG-систем порождает множество вопросов, подлежащих исследованию, таких как: сокращение времени задержек ответа, повышение эффективности поиска, наиболее оптимального хранения данных, методов оценки пайплайна, сохранения конфиденциальности, повторной индексации при смене механизма извлечения эмбеддингов и своевременного обновления базовых моделей.

4. Проблемы оценки систем, основанных на больших языковых моделях

Оценка систем, основанных на больших языковых моделях, в частности, оценка итогового ответа, сгенерированного языковой моделью, означает присвоение выражению, сформулированному на естественном языке, числового значения, отражающего соответствие ответа определенному, сформулированному заранее одному или нескольким критериям. Статистические методы оценки показывают низкую корреляцию с оценками человека, и это не позволяет использовать их на практике [45]. Одной из причин низкой корреляции с оценками человека можно назвать неспособность статистических моделей к построению семантических связей во входных последовательностях. У современных больших языковых моделей есть возможность эффективно имитировать процесс человеческой оценки, что открывает возможность использования данных моделей при оценке систем, результатом работы которых являются выражения, сформулированные на естественном языке в свободной форме.

Такой подход к оценке ответов систем, основанных на больших языковых моделях, называется LLM-как-судья (LLM-as-a-judge) [46]. У данного подхода есть существенные проблемы. Для применения подхода требуется корпус вопросов, итоговых ответов и оценок этих ответов. Также должны быть сформулированы правила оценивания ответа на вопрос, то есть шкала оценок с ее описанием на естественном языке – ровно в таком виде, чтобы человек, прочитавший ее и имеющий знания в доменной области, мог эффективно оценивать ответы. Со стороны LLM характерны искажения: смещение оценки из-за позиции варианта в списке, «подхалимство» – симпатия к стилю генерации моделей, имеющих схожую архитектуру и обученных на тех же данных, а также уязвимость к атакам в формулировке запроса [47, 48]. В этот список можно отнести и фактор случайности, мешающий воспроизводимости оценок. Случайность проявляется даже

при установке температуры модели на значение 0 – повторные запуски одного и того же запроса могут приводить к разным оценкам [49]. Температура – настраиваемый на уровне всей модели параметр, определяющий степень детерминизма в процессе генерации токенов. Чем выше температура, тем выше фактор случайности при выборе каждого последующего токена, это обусловлено сглаживанием распределения вероятностей токенов.

Решение указанных проблем в значительной степени связано с введением формализованных протоколов оценивания и требованием к структурированному формату вывода (Structured Output). Вместо итоговой оценки от модели, выраженной числом, структурированный вывод предполагает декомпозицию оценки, позволяющую модели шаг за шагом выполнить операции, которые при процессе оценки выполняет человек для формирования итогового значения (например, общее соответствие ответа теме вопроса). Данный подход позволяет снизить вариативность оценок, повысить корреляцию с человеческими оценками [50]. Для ответа на вопросы, получение которых происходит с использованием внешних источников, как, например, в RAG-системах, также можно требовать от модели явного указания цитат из источников, послуживших основанием для оценки ответа [51].

5. Лучшие практики применения больших языковых моделей в корпоративных информационных системах

Практика использования больших языковых моделей в корпоративных информационных системах появилась относительно недавно. Рассмотрим несколько удачных практик проектирования систем с применение больших языковых моделей, ориентируясь на которые можно снизить временные затраты на разработку и поддержку данных систем.

Рациональным принципом проектирования архитектуры системы с применением LLM может являться постепенное и своевременное ее усложнение. Начинать следует с простых архитектур, например, таких как цепочка последовательных вызовов LLM. Усложнять архитектуру системы, то есть добавлять в пайплайн новые шаги, задействующие LLM, требуется только в том случае, если уже реализованная архитектура не позволяет достичь целевого значения по метрике качества.

Замену уже использующейся в системе большой языковой модели на новую стоит рассматривать в случае улучшения показателей метрик системы при использовании новой модели по сравнению со значениями метрик старой и готовности специализированного программного обеспечения к запуску новой модели. Под готовностью специализированного программного обеспечения подразумевается прежде всего не только базовая поддержка новой модели для запуска, но и наличие соответствующих программных оптимизаций для повышения скорости работы новой модели. Не следует гнаться за обновлениями моделей, новейшие версии зачастую еще какое-то время не поддерживаются экосистемой, нужно ждать. Можно порекомендовать цикл обновления раз в квартал и брать только модели, которые уже протестированы, а инфарструктура их поддерживает нативно.

Для более прозрачного процесса поддержки и улучшения корпоративных систем с использованием больших языковых моделей необходимо иметь для каждого этапа получения итогового ответа набор тестовых данных. Это позволяет оценивать производительность текущей модели и изучать влияние изменений в одних этапах на другие, ведь они все связаны по цепочке и сбой в одном этапе будет вызывать каскадный сбой и искажения в последующих.

Значительного повышения качества ответов (производительности) системы с использованием LLM без усложнения самой архитектуры возможно достичь, тщательно улучшая промты, подаваемые на вход модели. Использование формата структурированного вывода позволяет направлять генерацию модели по заранее определенному пути, что тоже позволяет повысить итоговую производительность системы.

При оценке производительности системы, использующей LLM, также стоит опираться на обратную связь от конечных пользователей системы, если они доступны и могут ей поделиться. Человеческая оценка до сих пор остается самой достоверной в сравнении с LLM-as-a-judge и статистическими методами [52].

Обсудим также и применение больших языковых моделей в написании программного кода, поскольку они все чаще играют роль систем поддержки принятия решений в разработке ПО.

Эффективность системы, основанной на LLM и предназначенной для разработки программного обеспечения, во многом зависит от паттерна архитектуры проекта. Стоит рассматривать проработанные подходы, например, паттерны предметно-ориентированного программирования (Domain-Driven Design, DDD), программирования на основе функциональных срезов (Feature-Sliced Design, FSD) и подобных. Поскольку LLM показывают лучшие результаты при работе с небольшим контекстом и имеют свойство терять детали при работе с длинным контекстом [53], эффективней будет решаться задача, для постановки которой необходимо будет минимальное число токенов. Указанные ранее паттерны программирования позволяют снизить длину формулировки задачи для LLM, поскольку этому способствует проработанная структура программного проекта, а также высокий уровень абстракции и способ управления зависимостями. Допустим, текущий проект построен по DDD, тогда для внедрения нового функционала в одну из сущностей необходимо будет передать LLM ее описание и формат взаимодействия с остальными сущностями проекта, что в общем случае занимает небольшое число токенов в сравнении со случаем, где в силу архитектуры проекта пришлось бы передавать полный код нескольких классов. Вторым значимым фактором в эффективности такой системы является наличие автоматизированных тестов, позволяющих оперативно оценивать результат работы системы и предотвращать непредвиденные ошибки. Стоит отметить, что лучшие архитектурные паттерны еще не определены и нет достаточных исследований на этот счет. Здесь можно руководствоваться гипотезой о том, что чем лучше структурирован подход, пусть даже избыточно для простых случаев, тем лучшие результаты сможет предоставлять LLM.

Внедрение LLM в корпоративную систему стоит рассматривать при том условии, что более простые алгоритмы по выполнению задачи не дают ожидаемого результата либо для задачи нет возможности сформулировать алгоритм решения, но при этом ее может решить человек, а также при условии отсутствия необходимости в высокой интерпретируемости полученного решения, то есть можно руководствоваться принципом лишь бы работало.

Это обосновано рисками внедрения LLM, которые будут описаны далее. Внедрение LLM в корпоративную систему стоит рассматривать только если это единственное решение и его преимущества и возможности перевешивают риски внедрения.

6. Сравнение подходов к интеграции больших языковых моделей в корпоративные системы

В целях структурирования информации по основным подходам работы с LLM в ходе исследования было проведено сравнение вариантов интеграции LLM по ряду критериев, результаты которого представлены ниже. Сравнительный анализ подходов проводился с точки зрения использования каждой альтернативы в корпоративной системе, предназначенной для ответа на вопросы, возникающие у людей, включая как клиентов, так и сотрудников, в отношении компании. В сравнении участвуют три подхода: использование LLM (подразумевает собой направление запросов в LLM с фиксированным системным промтом), использование дообученной LLM (дополнительно обученная LLM на корпусе данных компании с фиксированным системным промтом), использование RAG (система с LLM с фиксированным промтом для генерации итогового ответа, модулем поиска для нахождения релевантных запросу данных и векторной базы данных для хранения информации).

В пунктах отражены критерии сравнения, в пояснениях к ним – оценка подхода по критерию.

1. Возможность адаптироваться к домену

LLM: присутствует только при задании соответствующего промта, понимание домена ограничено.

Дообученная LLM: присутствует, зависит от данных, использующихся для адаптации к домену.

RAG: аналогично дообученной LLM.

2. Требования к реализации решения

LLM: необходим провайдер модели, либо инфраструктура для локального запуска.

Дообученная LLM: более высокие требования к мощности локальной инфраструктуры для дообучения в сравнении с запуском.

RAG: помимо тех же требований, как к LLM, необходима дополнительная инфраструктура в

виде векторной базы данных, модели для векторизации информации. Дополнительные требования зависят от реализации RAG.

3. Возможность ссылаться на источники информации при ответе

LLM: возможно, если все источники помещаются в контекстное окно модели, однако результаты становятся нестабильными с ростом контекста [53].

Дообученная LLM: возможно в теории, однако на практике результаты оказываются нестабильными.

RAG: присутствует.

4. Время ожидания ответа

LLM: зависит от выделенных ресурсов, числа параметров LLM и оптимизации программного обеспечения.

Дообученная LLM: аналогично LLM.

RAG: из-за дополнительных этапов (модуль поиска, несколько запросов к LLM для оценки источников и для генерации ответа) чаще всего выше, чем у альтернатив.

5. Возможность оперативно обновлять контекстную информацию

LLM: затруднено, но возможно при оперативном обновлении промта модели. Однако изменение промта может повлечь снижение качества ответов.

Дообученная LLM: обновление устаревшей информации становится практически невозможным, так как модели не способны переучиваться.

RAG: обновить информацию легче, добавив ее векторное представление в базу данных.

6. Склонность к галлюцинациям

LLM: высокая, если в промте отсутствует релевантная информация. Если в промте отдельно прописаны инструкции, требующие отвечать, что информации по запросу нет, то вероятность галлюцинаций снижается.

Дообученная LLM: ниже, чем в LLM за счет того, что в силу дообучения модель запоминает больше информации, чем помещается в контекстное окно при стандартном проптинге.

RAG: Минимальная за счет присутствия первоисточника в контексте модели.

7. Возможность работы с большим объемом информации (10 тысяч токенов и более)

LLM: отсутствует в силу невозможности уместить ее в контекстное окно. Даже при условии того, что необходимая информация не занимает все контекстное окно, эффективность ответов значительно падает при объеме, сравнимом с размером контекстного окна.

Дообученная LLM: присутствует, но эффективность падает с ростом объема информации.

RAG: присутствует, эффективность не зависит от объема информации. Эффективность зависит от метода построения поискового индекса и модуля поиска релевантных данных в векторной базе данных.

8. Возможность генерировать ответы в зависимости от истории запросов и профиля пользователя

LLM: генерация ответов с учетом профиля и истории запросов пользователя невозможна на практике, поскольку требует заранее предопределенного списка пользователей и их предпочтений, а также их постоянности.

Помещать историю запросов каждый раз в промт не позволяет ограниченный контекст.

Дообученная LLM: аналогично LLM.

RAG: возможна при реализации механизма памяти, сохраняющего детали профиля пользователя в векторную базу данных.

Отдельно стоит рассмотреть агентов искусственного интеллекта, которые уже сейчас применяются в корпоративных информационных системах, поскольку возможности применения LLM – ограничены. ИИ-агенты (агенты искусственного интеллекта) – это следующая ступень развития использования нейросетей в бизнес-системах, которая требует расширенных критериев оценки.

7. Агенты искусственного интеллекта

На рис. 2 представлена обобщенная архитектура агентной системы на основе искусственного интеллекта. Генеративный ИИ-агент – система, которая управляет взаимодействием LLM с ее окружением для достижения цели, определенной пользователем, с помощью доступных модели инструментов [54]. Агенты способны действовать автономно, без прямых пошаговых указаний от

пользователя, на основе заданной задачи. Они способны самостоятельно планировать, какие шаги им необходимо предпринять, в каком порядке и с какими инструментами взаимодействовать.

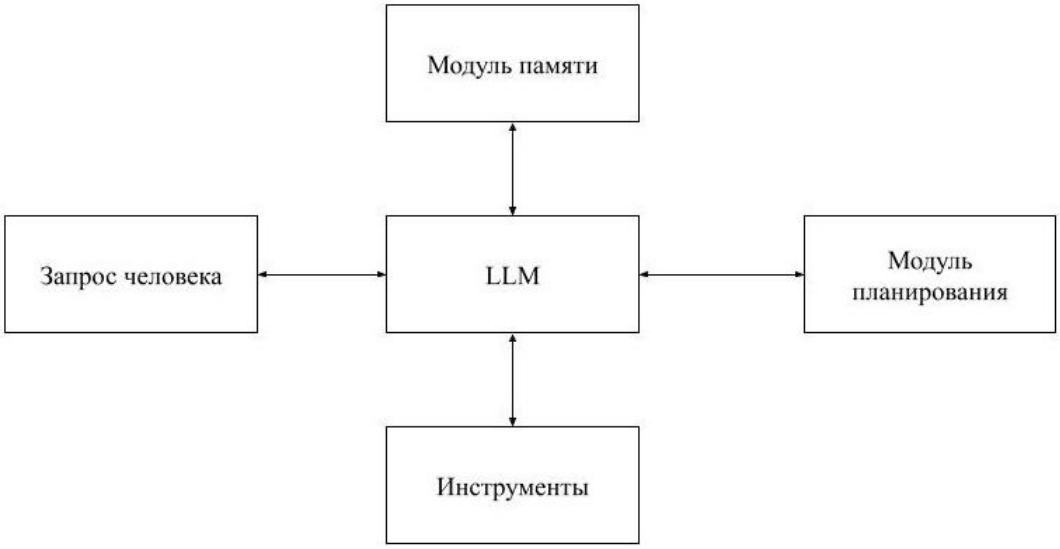

Рис. 2. Обобщенная архитектура агентной системы на основе искусственного интеллекта
Fig. 2. Generalized architecture of an agent system based on artificial intelligence

LLM является «мозгом» агента, выполняя в такой системе не просто функцию генерации итогового ответа: модель также отвечает за планирование, разбивая задачу на поэтапные шаги. Для этого применяются такие техники рассуждения моделей, как ReAct (рассуждение + действие), цепочка размышлений (Chain-of-Thought) или дерево размышлений (Tree-of-Thoughts) [54]. Планирование, или оркестрирование, – критически важный модуль (слой) в архитектуре агентов, поскольку управляет памятью, текущим состоянием агента, процессами рассуждения и принятия решений.

Память также является модулем архитектуры агента. Она позволяет сохранять как данные в рамках отдельной сессии пользователя, так и информацию о прошедших взаимодействиях, их частичный контекст и результаты, которые в будущем могут использоваться для задач со схожим контекстом.

Отдельным компонентом системы является модуль, отвечающий за использование агентами различных инструментов для взаимодействия с внешней средой. Как правило, инструменты интегрируются с помощью стандартных API (Application Programming Interface) операций [54], позволяя агентам, например, взаимодействовать напрямую с базами данных или делать запросы в Интернет. Это позволяет системам обрабатывать информацию, получаемую в масштабе, близком к реальному времени, и действовать на ее основе. Инструменты создают взаимосвязь между внутренней вычислительной и аналитической мощностью LLM и возможностями IT-систем, существующих во внешней среде. В настоящий момент зарождаются и формируются общепринятые стандарты взаимодействия агентов с внешней средой, такие как, например, MCP (Model Context Protocol) [55]. Кроме того, разрабатываются протоколы для взаимодействия самих агентов между собой, например, A2A (Agent to Agent Protocol) [56]. Индустрия видит в этих стандартных протоколах будущее для объединения инструментов в любых комбинациях для решения сложных, комплексных задач.

Текущий ход исследований и развития ИИ-агентов характеризуется быстрыми инновациями со стороны академических учреждений и лидеров индустрии. Такие организации, как IBM, Microsoft, AWS и Anthropic разрабатывают все более сложные архитектуры агентов, а команды исследователей в университетах предлагают новые бенчмарки (Benchmark) и системы оценки [57]. Развитие ИИ-агентов объединяет интересы академической и коммерческой сфер, и их совместный вклад значительно ускоряет прогресс этого направления.

ИИ-агенты уже активно применяются в корпоративной среде, изменяя бизнес-процессы, выполняя роль инструмента поддержки принятия решений и оптимизируя повседневный ход дея-

тельности предприятий. Они постепенно заменяют предыдущие технологии ML/AI, ранее применяющиеся в BI-системах. Доступ к инструментам и способность действовать автономно позволяют агентам объединять в себе функционалы различных инструментов аналитики, планирования и генерации контента. За счет взаимодействия с данными в реальном времени они обладают повышенной адаптивностью к сложным сценариям и изменяющимся обстоятельствам, из-за чего активно применяются в автоматизации бизнес-процессов и в DSS [57]. Использование агентов в сфере CRM позволяет снизить затраты на содержание соответствующего отдела, уменьшить среднее время ожидания ответа от службы поддержки и обеспечить ее круглосуточную работу [57]. Также одним из самых распространенных способов применения ИИ-агентов является использование их в качестве интеллектуальных помощников в бизнес-коммуникации и разработке ПО, где они задействуются в рутинных задачах написания сообщений или кода соответственно.

На самом деле сейчас сложно найти часть корпоративной деятельности, которую еще не затронули ИИ-агенты в той или иной степени. Кроме предприятий они используются и индивидуальными пользователями в личных, учебных и рабочих целях: от составления списка покупок или бюджета до формирования резюме и обучения. К помощи агентов также прибегают в медицине, урбанистике, сельском хозяйстве и других сферах [57].

При этом ИИ-агенты во многом имеют те же недостатки прошлых LLM и основанных на LLM альтернатив: вопрос ограниченности контекста, необходимость качественного промта, безопасность извлекаемых данных, прозрачность ответов, риск галлюцинаций и прочее. По мере развития самих LLM, использующихся в ядре агентов, и методов реализации других компонентов их архитектуры, многие из этих проблем становятся менее и менее заметными, однако совершенствование и распространение ИИ-агентов становится все более значимым поводом для обсуждения с точки зрения их непосредственного влияния на сферы, где они применяются, и на деятельность человека в них. Влияние ИИ на рынок труда, качество образование, творческую сферу, на способность человека действовать самостоятельно, рассуждать и мыслить критически рождает множество поводов для исследований и дискуссий.

8. Вайб-кодинг как новый подход к разработке программного обеспечения и его риски

Вайб-кодинг (Vibe Coding) – это довольно экстремальный подход к разработке программного обеспечения, характеризующийся делегированием написания программного кода, его отладки и проектирования архитектуры ПО с человека на LLM. Роль человека при этом заключается в формулировании требований к ПО и передаче их в LLM посредством промтинга. Зачастую для вайб-кодинга характерно неформальное выражение требований к ПО. Термин вайб-кодинг появился в начале 2025 года и получил распространение благодаря высказыванию Андрея Карпатого [58].

Можно выделить несколько характерных черт подхода, основанного на передаче работ по непосредственному исполнению задач от человека к LLM, причем он применяется как в разработке программного обеспечения, так и в других сферах, где можно делегировать выполнение работы посредством программных инструментов, например, закупки по требованиям, обработка заявок, то есть любая деятельность, где человек работает по скрипту и «скрепляет» собой инструменты. Отличительной чертой так называемого вайб-кодинга является тот факт, что процесс реализации идеи происходит при помощи диалога между человеком и LLM на естественном языке. Также для данного подхода характерно снижение или полное отсутствие необходимости владения техническими навыками в конкретной области для достижения желаемого результата, потому как современные LLM демонстрируют высокое качество в генерировании текста и программного кода на специализированных тестах качества [59]. Однако важно понимать, что рейтинги могут быть субъективны из-за утечек тестовых данных, на которых инструменты могут переобучаться, чтобы возглавить вершину рейтинга. Всегда необходимо проводить самостоятельное тестирование, включая модели, которые не находятся на первых позициях в списках лучших.

К возможностям, которые открывает вайб-кодинг, относятся снижение порога входа в разработку программного обеспечения, ускорение написания кода и прототипирования, освобождение части ресурсов человека и перенаправление их на планирование дальнейшего функционала разрабатываемого ПО. Способность LLM определять намерения человека из сформулированных на естественном языке предложений и реализовать их в программном коде позволяет

ускорять процесс создания программного обеспечения, и разработка при вайб-кодинге приобретает черты партнерства, где LLM отвечает за реализацию, а человек – за планирование и креативные задачи [58].

Применение вайб-кодинга может нести в себе как риски, связанные с низким качеством самой реализации ПО, то есть имеющие технический характер, так и риски, связанные с ухудшением навыков разработчиков, которое может стать следствием бесконтрольного использования ими такого подхода.

Код, сгенерированный моделью, может быть излишне усложнен, что в дальнейшем мешает специалистам поддерживать и развивать его. В силу возможной неверной интерпретации задачи со стороны LLM вайб-кодинг может повлечь непредвиденные изменения в кодовой базе программного обеспечения, как, например, добавление ненужного функционала, либо упрощение, удаление уже имеющегося. Кроме этого, процесс оценки работоспособности и качества кода становится менее прозрачным в сравнении с традиционными методами разработки, поскольку работа полностью делегируется LLM. Последствием приведенных рисков становится ускоренное накопление так называемого технического долга.

Вайб-кодинг вследствие делегирования исполнения технических задач LLM снижает у человека необходимость в критическом мышлении в процессе разработки, что со временем может быть причиной снижения уровня имеющихся у разработчика технических навыков, потому как при вайб-кодинге их использование больше не является необходимостью. Ослабление или потеря глубокого понимания технологий создают зависимость от LLM. В частности, из-за этого эффекта появляется уязвимость – при отсутствии доступа к LLM разработчики могут оказаться неспособными к разработке и поддержке программного обеспечения.

Можно выделить несколько принципов и условий, придерживаясь которых разработчик может применять подход вайб-кодинга без потери собственной эффективности и самостоятельности. Перечисленные далее условия релевантны в том случае, если вайб-кодинг используется программистом в качестве основного способа разработки. Для того чтобы предотвратить деградацию навыков, разработчику следует вручную реализовывать какую-либо часть поставленных задач, либо проводить оценку уже сгенерированного кода, решающего эти задачи. Определять соотношение задач с подробным разбором или ручной реализацией и задач, полностью делегируемых LLM, стоит исходя из личного опыта и уровня навыков разработчика – чем выше компетенции, тем меньшее число задач можно решать вручную без значительной деградации навыков.

Для обеспечения возможности дальнейшей поддержки и разработки ПО необходимо при его проектировании следовать архитектурным шаблонам, работая в рамках которых LLM должны показывать более стабильные, качественные и предсказуемые результаты. Список шаблонов и паттернов ни в коем случае не ограничивается лишь рассматриваемыми ранее DDD и FSD. Кажется, что хороший результат могут показывать любые хорошо документированные и структурированные подходы. Их применение позволяет контролировать зависимости проекта и формат кода, генерируемый в процессе вайб-кодинга, делая его предсказуемым, стандартизированным и понятным для разработчика. Также такой код проще тестировать и изменять без рисков затронуть другой функционал системы, что решает проблему непредвиденных изменений в программном обеспечении при вайб-кодинге.

В общем и целом если рассматривать применение вайб-кодинга или близкого к нему подхода, где человек согласовывает все изменения, для долгоживущих проектов требования к качеству кода должны быть строго регламентированы и в идеале их соблюдение должно верифицироваться не только человеком, но и автоматическими тестами. Не все такие тесты писать легко или даже возможно. Подход разработки программного обеспечения с глубокой интеграцией ИИ-агентов выглядит многообещающим, но все еще весьма нестабильным и высокорискованным сегодня.

Стоит отметить, что вайб-кодинг, применение которого согласовано с вышеизложенными принципами, может стать управляемым и эффективным процессом, полностью раскрывающим плюсы подхода и минимизирующим его отрицательные стороны. Однако для индустрии этот путь может оказаться ложным, ведущим к быстрому росту и дальнейшему откату и катастрофическому разочарованию от несбывшихся надежд о высокой скорости поставки изменений и снижении накладных расходов на разработку за счет сокращения количества вовлекаемых в процесс людей, которым необходимо платить заработную плату.

9. Риски внедрения больших языковых моделей

Внедрение LLM в корпоративные системы сопряжено с рисками, связанными со спецификой работы этих моделей.

Одним из главных рисков внедрения LLM в коммерческие продукты компании либо в ее внутренний контур, где цена ошибки высока и от точности и предсказуемости ответов зависит репутация компании, на данный момент являются галлюцинации. Они делятся на две основные категории: внутренние и внешние. В первом случае сгенерированный моделью ответ противоречит данным, на которых она обучалась, а во втором – ответ не может быть ни подтвержден, ни опровергнут тренировочной выборкой. В случае LLM чаще встречается именно второй тип, во многом потому, что эти модели обучаются на огромном объеме данных, зачастую содержащих противоречащие друг другу факты. По этой причине одним из ключевых методов нейтрализации этого риска считается улучшение качества данных, использующихся для обучения. Тонкая настройка и RAG также считаются более предпочтительными подходами для внедрения LLM с точки зрения галлюцинаций, чем базовые LLM, из-за более точной и узкоспециализированной выборки данных, на которые опираются модели при ответе. Хотя цена получения более качественных данных может быть слишком высокой для большинства предприятий.

Помимо галлюцинаций опасность представляют атаки на LLM, позволяющие злоумышленникам получить системный промт модели, заставить модель переписать доступную ей информацию или склонить к выполнению каких-либо действий в обход системных инструкций [60].

Отдельно стоит рассмотреть риски внедрения LLM в качестве разработчика программного обеспечения. Траектория роста компетенций нового программиста подразумевает сначала освоение базовых технических навыков, включающих синтаксис языка или языков программирования, затем постепенное накопление практического опыта при решении задач совместно с углублением в архитектурную составляющую. LLM, используемая в качестве универсального исполнителя, не позволяет неопытному программисту в должной степени овладеть необходимыми ему для профессионального роста базовым техническим навыкам. В такой ситуации начинающий специалист вынужден миновать освоение азов профессии и сразу управлять разработкой при помощи LLM. Это проблема, ведь для эффективной разработки при помощи LLM программисту необходимо обладать глубокой экспертизой в программировании и используемом стеке технологий, а также пониманием архитектурных паттернов и умением их применять, а эти навыки появляются только со временем за счет получения собственного опыта, путем решения задач вручную, а не с помощью LLM. Есть риск, что индустрия станет более закрытой для новичков, где позиции экспертов, учившихся десятилетиями за счет медленной постепенной практики, лишь усилятся, а приток начинающих специалистов снизится за счет реализации идеи экономии на найме и обучении новых сотрудников.

Такой сценарий может поставить отрасль разработки программного обеспечения вместе со сферой подготовки этих кадров в сложное положение, при котором лишь усугубится кризис предложения, которое будет значительно превышать спрос на рынке труда. Эта тенденция наблюдается уже сейчас. Вовлечение же специалистов с непрофильным образованием в сферу информационных технологий может оказаться под еще большим вопросом, ведь задачей, стоящей перед новыми специалистами, будет быстро стать архитекторами программного обеспечения, а это не позволит людям без глубокой фундаментальной базы эффективно продвигаться вперед по карьерной лестнице.

Развитие ИИ-агентов, в свою очередь, несет в себе уникальные риски. При использовании агентов в корпоративных системах важно контролировать среду, доступную агенту, тщательно оценив безопасность доступных ему инструментов, потому как при наличии потенциально опасного инструмента, допустим, права на выполнение команды в консоли от имени суперпользователя, существует вероятность исполнения вредоносного сценария как вследствие атаки на агента, так и вследствие галлюцинаций. Примером такого сценария в данном случае может служить команда удаления домашних каталогов пользователей, использованная агентом во время установки сторонних пакетов.

С точки зрения человеческих ценностей существует риск расхождения ценностей искусственного интеллекта и ценностей человека [61]. Высокий уровень развития искусственного интеллекта не означает, что его финальные цели соответствуют международным нормам морали и

права. Иными словами, интеллект сам по себе не гарантирует полное отсутствие вредоносного поведения. Процесс согласования ценностей искусственного интеллекта и ценностей человечества называется выравниванием (Alignment) и представляет собой активно исследуемую область исследований LLM [62]. В каком-то смысле искусственный интеллект можно рассматривать в качестве психопата – того, в ком нет эмпатии, нет эмоций.

10. Будущее применения искусственного интеллекта в бизнесе

Уже сейчас использование LLM становится частью повседневной практики в различных направлениях бизнеса. Компании внедряют вопросно-ответные системы на базе LLM для улучшения взаимодействий с клиентами, пробуют применять ИИ-агентов в создании контента, разработке ПО, BI-задачах и не только. На основе этого меняется природа рабочих процессов, зарождаются новые методики: например, вайб-кодинг в разработке. Умение взаимодействовать с LLM и с ИИ-агентами становится обязательным требованием многих работодателей по аналогии с тем, как десятилетия назад таковым становилось умение работать с компьютером и офисными программами. ИИ становится элементом технологического стека компаний, и по степени инновационного влияния этот процесс сравним с изобретением парового двигателя в XIX веке [63].

В ближайшее время можно ожидать усиления текущих тенденций. Вайб-кодинг будет развиваться как отдельная методика со своими лучшими практиками, сферами и антисферами применения и способами митигации рисков, связанных с ним. Упрощение процесса написания кода и, соответственно, снижение порога входа в сферу программирования приведет к большему количеству небольших, одноразовых программных решений. Теперь многие вопросы, с которыми сотрудники других направлений обращались в ИТ-отдел, можно будет решить самостоятельно, воспользовавшись агентом. Кода в целом станет больше – отчасти из-за того, что его легче написать, а отчасти потому, что он потребуется для построения корпоративных систем вокруг ИИ и создания «прослойки» для взаимодействия сотрудников с LLM или агентами.

Возможность делегировать задачи ИИ-агентам может стать определяющим фактором получения компанией преимущества перед конкурентами [64]. Организации будут стремиться при наименьших возможных затратах получить наиболее «умные» модели – не идеальные, но достаточно точные и стабильные для решения их задач. Умение грамотно делегировать задачи ИИ, включая написание эффективной инструкции, предоставление нужного контекста и проверку итогового ответа, станет необходимым навыком для сотрудников разных уровней и направлений.

Другой развивающейся тенденцией является интеграция рекламы в интерфейсы и ответы LLM на основе запросов пользователя. Некоторые компании, такие как OpenAI, Perplexity, Microsoft, уже рассматривают или используют эту концепцию в своих ИИ-продуктах. Со временем ИИ-агенты, вероятно, станут уникальным маркетинговым каналом, требующим отдельного внимания от специалистов по продвижению [65–67]. Вряд ли можно ожидать, что компании, владеющие сервисами на основе LLM, смогут полностью избежать соблазна примешивать к объективным данным продвижение выгодных им идей и услуг.

Рынок найма также может значительно измениться. Помимо уже упомянутых ранее требований к кандидатам обновится и процесс их отбора: с помощью LLM можно будет осуществлять «бесконечный» перебор в поисках сотрудника, наиболее точно подходящего под требования компании. Такая автоматизация рекрутинга реализуется уже сейчас, продолжая набирать все большую популярность [68].

С таким усилением внедрения ИИ в корпоративную среду лидерам компаний потребуется уделить отдельное внимание вопросам безопасности. Реальная вероятность катастрофического сценария развития искусственного интеллекта остается неопределенной, однако полностью ее отрицать на данный момент невозможно. Масштабы внедрения ИИ таковы, что подход к безопасности в этой сфере должен становиться таким же серьезным и институционализированным, как в сферах ядерной и биохимической безопасности [69], поскольку пока что не до конца известно, какие сценарии могут возникнуть и реализоваться и, соответственно, как действовать в каждом из них. Вероятно, в ближайшие годы появится практика создания внутри компаний специализированных подразделений, осуществляющих наблюдение за использующимися организациями ИИ-системами и их калибровку по технике обучения с закреплением на основе отзывов людей (Reinforcement Learning with Human Feedback) [69].

Скорость внедрения ИИ в бизнес в первую очередь означает необходимость успевать за всеми упомянутыми тенденциями. Если еще пару лет назад использование LLM и искусственного интеллекта было некой надстройкой над уже существующей инфраструктурой и информационной системой компании, то сейчас эти технологии встраиваются непосредственно в эти структуры, заменяя ранее применявшиеся подходы.

В ближайшем будущем в сфере ИИ требуется утверждение стандартов, методик, лучших практик, организационных единиц и прочих элементов, которые позволили бы регулировать темп, с которым происходит развитие в этой индустрии, причем как на уровне отдельных компаний, так и на мировом. Вопрос «Стоит ли внедрять ИИ?» уже не столь актуален: теперь рациональнее думать о том, как это сделать наиболее грамотно, эффективно, этично и безопасно.

Заключение

Рассмотрев во временном разрезе развитие систем с математическими алгоритмами и переход к использованию искусственного интеллекта, понимаем – мы совершили прорыв и органично вступили в новую гонку для человечества. Сейчас не стоит вопрос о том, использовать ли искусственный интеллект в повседневной жизни и корпоративных системах. Ответ очевиден – использовать. Необходимо проектировать системы поддержки принятия решений таким образом, чтобы они были готовы к следующему витку в спирали бесконечного совершенствования алгоритмов искусственного интеллекта.

Ускорение бизнес-процессов и оптимизация рутинной работы человека за счет внедрения ИИ способствует новым свершениям в разных значимых сферах, например, медицине или производстве, а способность искусственного интеллекта искать нетривиальные закономерности на пересечении областей знаний позволяет за счет широты обучающих данных дополнить глубину знаний отдельных людей. Если рассматривать ИИ как обобщенное представление всех знаний человечества – то в лице умных алгоритмов мы находим поддержку в принятии наших решений.

Но нужно учитывать риски, чтобы гонка не привела нас к катастрофе. В частности, мы рискуем попасть в ситуацию, где слепое доверие ИИ приведет нас к тому, что мир стал состоять из галлюцинаций, дыр в безопасности и решений логичных, но тем не менее антигуманных. Передача полного контроля управлением из рук человека в «руки» ИИ на текущий момент невозможна, поскольку не установлены четкие этические рамки и зона ответственности ИИ.

Необходимо также не допустить деградации у человека критически важных способностей, которые ранее считались принадлежащими только людям, что сегодня тем не менее ставится под сомнение: креатив, критическое мышление и профессиональные навыки уже не рассматриваются как прерогатива интеллекта естественного. Человечество и корпорации должны найти способ обогатить свою жизнь за счет внедрения ИИ, а не допустить собственной деградации и разрушения.

Искусственный интеллект – это партнер, который помогает развитию человечества и ускорению рутинных процессов. Поиск синергии между искусственным и естественным интеллектом – это новый вызов для человечества. Будем надеяться, что этот вызов нам по плечу.

Список литературы/References

1. Abu Naser S. Dynamic Programming as a Tool of Decision Supporting. *Journal of Applied Sciences Research.* 2009; 5(6):671–676.

2. Power D.J. A Brief History of Decision Support Systems. *DSSResources.com.* 2007.

3. Copeland B.J. MYCIN. *Encyclopaedia Britannica*. 2018.

4. Codd E.F. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*. 1970;13(6):11.

5. Oracle: website. Available at: https://www.oracle.com/ (accessed 15.08.2025).

6. What is OLAP (online analytical processing)? *IBM: website*. Available at: https://www.ibm.com/think/topics/olap (accessed 15.08.2025).

7. Клеппман М. Высоконагруженные приложения. Программирование, масштабирование, поддержка. Sprint Book, 2025. 640 с. (Бестселлеры O'Reilly). [Kleppman M. *High-Load Applications. Programming, Scaling Support*. Sprint Book, 2025. 640 p. (O'Reilly Bestsellers Series) (In Russ.)]. Available at: https://www.piter.com/collection/all/product/vysokonagruzhennye-prilozheniya-programmirovanie-masshtabirovanie-podderzhka-2 (accessed 15.08.2025).

8. Verma S., Rattan, P. Introduction to Data Mining Tools and Techniques Applications: A Review. In: *International Conference on Emerging New World (IECENW-2021).* 2021:21.

9. What Is Business Intelligence (BI)? *IBM: website*. Available at: https://www.ibm.com/think/topics/business-intelligence) (accessed 17.08.2025).

10. Fayyad U., Piatetsky-Shapiro G., Smyth, P. From Data Mining to Knowledge Discovery in Databases. *AI Magazine.* 1996;17(3):37. DOI: 10.1609/aimag.v17i3.1230

11. Shuvo S.A., Tabassum M., Tafannum N., Chadni S. Machine Learning in Business Intelligence: From Data Mining to Strategic Insights in MIS. *Review of Applied Science and Technology*. 2025;4:339–369. DOI: 10.63125/drb8py41

12. Arnott D., Shijia G., Lizama F., Meredith R.A., Sont Y. Are business intelligence systems different to decision support systems? In: *Australasian Conference on Information Systems (ACIS).* 2019:11.

13. Power D.J. Understanding Data-Driven Decision Support Systems. *Information Systems Management*. 2008; 25(2):149–154. DOI: 10.1080/10580530801941124

14. Rumelhart D.E., Hinton G.E., Williams R.J. Learning representations by back-propagating errors. *Nature*. 1986;323:533–536. DOI: 10.1038/323533a0

15. Moody J.E., Darken C.J. Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation*. 1989;1(2):281–294. DOI: 10.1162/neco.1989.1.2.281

16. LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W., Jackel L.D. Handwritten Digit Recognition with a Back-Propagation Network. *Advances in Neural Information Processing Systems.* 1989;2:396–404.

17. West D. Neural network credit scoring models. *Computers & Operations Research*. 2000;27(11):1131–1152. DOI: 10.1016/S0305-0548(99)00149-5

18. Zhou H., Lange K., Suchard M.A. Graphics Processing Units and High-Dimensional Optimization. *Statistical Science*. 2010; 25(3):311-324. DOI: 10.1214/10-STS336

19. Bhattacharyya S., Jha S., Tharakunnel K., Westland J.C. Data mining for credit card fraud: A comparative study. *Decision Support Systems*. 2011;50:602–613. DOI: 10.1016/j.dss.2010.08.008

20. Kozodoi N., Zinovyeva E., Valentin S., Pereira J., Agundez R. Probabilistic Demand Forecasting with Graph Neural Networks. *Arxiv preprint arXiv:2401.13096.* 2024:17.

21. Liu J., Zhang Y., Wang X., Deng Y., Wu X. Dynamic Pricing on E-commerce Platform with Deep Reinforcement Learning: A Field Experiment. *arXiv:1912.02572v3*. 2019:9.

22. Covington P., Adams J., Sargin E. Deep Neural Networks for YouTube Recommendations. In: *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. 2016;9.

23. Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781*. 2013:12.

24. Sutskever I., Vinyals O., Le Q.V. Sequence to Sequence Learning with Neural Networks. *Advances in neural information processing systems*. 2014;27:9.

25. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł., Polosukhin I. Attention Is All You Need. *Advances in Neural Information Processing Systems.* 2017;30:15.

26. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies.* 2019;1:16.

27. Liu Y., Ott M., Goyal N., Du J., Joshi M., Chen D., Levy O., Lewis M., Zettlemoyer L., Stoyanov V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*. 2019:10.

28. Warner B., Chaffin A., Clavié B., Weller O., Hallström O., Taghadouini S., Gallagher A., Biswas R., Ladhak F., Aarsen T., Cooper N., Adams G., Howard J., Poli I. Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference. *arXiv preprint arXiv:2412.13663*. 2024:20.

29. Yu S., Chen Y., Zaidi H. AVA: A Financial Service Chatbot based on Deep Bidirectional Transformers. *Frontiers in Applied Mathematics and Statistics*. 2021;7:604842. DOI: 10.3389/fams.2021.604842

30. Kim A.G., Yoon S. Corporate Bankruptcy Prediction with Domain – Adapted BERT. *arXiv preprint arXiv*:*2312.03194*. 2023:11.

31. Radford A., Narasimhan K., Salimans T. Sutskever I. Improving Language Understanding by Generative Pre-Training. *OpenAI Preprint*. 2018:12.

32. Radford A., Wu J., Child R., Luan D., Amodei D., Sutskever I. Language Models are Unsupervised Multitask Learners. *OpenAI blog*. 2019;1(8):9.

33. Brown T., Mann B., Ryder N., Subbiah M., Kaplan J.D., Dhariwal P., Neelakantan A., Shyam P., Sastry G., Askell A., Agarwal S. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*. 2020;33:1877–1901.

34. Cheung M. A Reality check of the benefits of LLM in business. *arXiv preprint arXiv*:*2406.10249*. 2024:20.

35. Rezkellah R., Temam A.M., Lhadj L.S., Dania A.A. Leveraging Large Language Models for business processes: A focus on Customer Service. *Final Year Project Report for École Nationale Supérieure d'Informatique.* 2025:149.

36. Ji Z., Lee N., Frieske R., Yu T., Su D., Xu Y., Ishii E., Bang Y.J., Madotto A., Fung P. Survey of Hallucination in Natural Language Generation. *ACM computing surveys*. 2022;55(12):1–38.

37. Ayyamperumal S.G., Ge L. Current state of LLM Risks and AIGuardrails. *arXiv preprint arXiv:2406.12934*. 2024:9.

38. Du H., Liu S., Zheng L., Cao Y., Nakamura A., Chen L. Privacy in Fine-tuning Large Language Models: Attacks, Defenses, and Future Directions. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2024:19.

39. Sun J., Mei C., Wei L., Zheng K., Liu N., Cui M., Li T. Dial-insight: Fine-tuning Large Language Models with High-Quality Domain-Specific Data Preventing Capability Collapse. *arXiv preprint arXiv:2403.09167*. 2024:10.

40. Lermen S., Rogers-Smith C., Ladish J. LoRA Fine tuning Efficiently Undoes Safety Training in Llama 2-Chat 70B, *arXiv preprint arXiv:2310.20624*. 2023:11.

41. Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., Küttler H., Lewis M., Yih W.T., Rocktäschel T., Riedel S. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems*. 2020;33:9459–9474.

42. Barnett S., Kurniawan S., Thudumu S., Brannelly Z., Abdelrazek M. Seven Failure Points When Engineering a Retrieval Augmented Generation System. In: *Proceedings of the IEEE. ACM 3rd International Conference on AI Engineering-Software Engineering for AI*. 2024:194–199.

43. Shen M., Umar M., Maeng K., Suh G.E., Gupta U. Towards Understanding Systems Trade-offs in Retrieval-Augmented Generation Model Inference. *arXiv preprint arXiv:2412.11854*. 2024:4.

44. Arslan M., Munawar S., Cruz C. Business insights using RAG–LLMs: a review and case study. *Journal of Decision Systems*. 2024:30. DOI: 10.1080/12460125.2024.2410040

45. Chan C.M., Chen W., Su Y., Yu J., Xue W., Zhang S., Fu J., Liu, Z. Chateval: Towards better llmbased evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*. 2023:16.

46. Zheng L., Chiang W.L., Sheng Y., Zhuang S., Wu Z., Zhuang Y., Lin Z., Li Z., Li D., Xing E., Zhang H. Judging llmasajudge with mtbench and chatbot arena. *Advances in neural information processing systems*. 2023;36.

47. Shi L., Ma C., Liang W., Diao X., Ma W., Vosoughi, S. Judging the Judges: A Systematic Study of Position Bias in LLM-as-a-Judge. *arXiv preprint arXiv:2406.07791*. 2025:22.

48. Wataoka K., Takahashi T., Ri R. Self-preference bias in llm-as-a-judge. *arXiv preprint arXiv:2410.21819*. 2025:11.

49. Atil B., Aykent S., Chittams A., Fu L., Passonneau R.J., Radcliffe E., Rajagopal G.R., Sloan A., TudrejT., Ture F., Wu Z. Non-Determinism of "Deterministic" LLM Settings. *arXiv preprint arXiv:2408.04667*. 2025:15.

50. Liu Y., Iter D., Xu Y., Wang S., Xu R., Zhu C. G-EVAL: NLG Evaluation using GPT-4 with Better Human Alignment. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP*). 2023:2511–2522.

51. Muller S., Loison A., Omrani B., Viaud G. Grouse: A benchmark to evaluate evaluators in grounded question answering. *arXiv preprint arXiv:2409.06595*. 2024:25.

52. Abeysinghe B., Circi R. The Challenges of Evaluating LLM Applications: An Analysis of Automated, Human, and LLM-Based Approaches. *arXiv preprint arXiv:2406.03339*. 2025:15.

53. Modarressi A., Deilamsalehy H., Dernoncourt F., Bui T., Rossi R. A., Yoon, S., Schütze H. NoLiMa: Long-Context Evaluation Beyond Literal Matching. *arXiv preprint arXiv:2502.05167*. 2025:17.

54. Viswanathan G., Samdani G., Dixit Y. AI Agents. *International Journal of Advanced Information Technology*. 2025;15(1/2):9–17.

55. Hou X., Zhao Y., Wang S., Wang H. Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. *arXiv preprint arXiv:2503.23278*.2025:37.

56. Pajo P. Comprehensive Analysis of Google's Agent2Agent (A2A) Protocol: Technical Architecture, Enterprise Use Cases, and Long-Term Implications for AI Collaboration. 2025:5. DOI: 10.13140/RG.2.2.32649.56164

57. Krishnan N. AI Agents: Evolution, Architecture, and Real-World Applications. *arXiv preprint arXiv: 2503.12687*. 2025:52.

58. Meske C., Hermanns T., von der Weiden E., Loser K.U., Berger, T. Vibe Coding as a Reconfiguration of Intent Mediation in Software Development: Definition, Implications, and Research Agenda. *arXiv preprint arXiv:2507.21928*. 2025:17.

59. OpenAI. GPT-5 System Card. 2025:60.

60. Liu X., Xu N., Chen M., Xiao, C. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. *arXiv preprint arXiv:2310.04451*. 2023:21.

61. Tallarita R. AI is Testing the Limits of Corporate Governance, Insights You Need from Harvard Business Review: A Year in Tech 2025. Harvard Business Review Press, 2025:53–71.

62. Wang Z., Bi B., Pentyala S.K., Ramnath K., Chaudhuri S., Mehrotra S., Asur S. A Comprehensive Survey of LLM Alignment Techniques: RLHF, RLAIF, PPO, DPO and More. *arXiv preprint arXiv: 2407.16216*. 2024:37.

63. Mayer H., Yee L., Chui M., Roberts R. Superagency in the workplace: Empowering people to unlock AI's full potential. McKinsey Company, 2025.

64. Bombalier J. The Competitive Advantage of Using AI in Business. FIU College of Business – Graduate Insights.

65. Tang B.J., Sun K., Curran N.T., Schaub F., Shin K.G. GenAI Advertising: Risks of Personalizing Ads with LLMs. *arXiv preprint arXiv: 2409.15436*. 2024:28.

66. Kumar S. Incorporating Ads into Large Language Models Outputs. *Sumit's Diary (Blog)*. 2024.

67. Fried I. OpenAI looks at chatbot ads. *Axios*. 2024.

68. Steinberg B. AI recruiting is all the rage. *New York Post*. 2025.

69. Webb A. *How to Prepare for a Gen AI Future You Can't Predict*. Insights You Need from Harvard Business Review: A Year in Tech 2025. Harvard Business Review Press, 2025:155–169.

Информация об авторах

Шинкарев Александр Андреевич, канд. техн. наук, доц. кафедры информационных систем и технологий, Южно-Уральский государственный университет, Челябинск, Россия; sania.kill@mail.ru.

Ядрышникова Мария Викторовна, аспирант кафедры информационных систем и технологий, Южно-Уральский государственный университет, Челябинск, Россия; reeyardma@gmail.com.

Логиновский Олег Витальевич, д-р техн. наук, проф., проф. кафедры информационных систем и технологий, Южно-Уральский государственный университет, Челябинск, Россия; loginovskiiov@susu.ru.

Лазарева Снежана Андреевна, выпускник кафедры системного программирования, Южно-Уральский государственный университет, Челябинск, Россия; lazarevas124@gmail.com.

Губин Владимир Михайлович, студент кафедры системного программирования, Южно-Уральский государственный университет, Челябинск, Россия; teacfoou@gmail.com.

Information about the authors

Alexander A. Shinkarev, Cand. Sci (Eng.), Ass. Prof. of the Department of Information Systems and Technologies, South Ural State University, Chelyabinsk, Russia; sania.kill@mail.ru.

Maria V. Yadryshnikova, Postgraduate Student of the Department of Information Systems and Technologies, South Ural State University, Chelyabinsk, Russia; reeyardma@gmail.com.

Oleg V. Loginovskiy, Dr. Sci. (Eng.), Prof., Prof. of the Department of Information Systems and Technologies, South Ural State University, Chelyabinsk, Russia; loginovskiiov@susu.ru.

Snezhana A. Lazareva, Graduate of the Department of Systems Programming, South Ural State University, Chelyabinsk, Russia; lazarevas124@gmail.com.

Vladimir M. Gubin, Student of the Department of Systems Programming, South Ural State University, Chelyabinsk, Russia; teacfoou@gmail.com.

Вклад авторов: все авторы сделали эквивалентный вклад в подготовку публикации.
Авторы заявляют об отсутствии конфликта интересов.
Contribution of the authors: the authors contributed equally to this article.
The authors declare no conflicts of interests.

Статья поступила в редакцию 18.08.2025
The article was submitted 18.08.2025