ISSN 0234-0453 • INFORMATICS AND EDUCATION • 2026 • Volume 41 No 1

DOI: 10.32517/0234-0453-2026-41-1-23-36

CC BY

ИНТЕГРАЦИЯ НЕЙРОСЕТЕВОГО АССИСТЕНТА ОБУЧАЮЩИХСЯ ПО ПРОГРАММИРОВАНИЮ В ЦИФРОВУЮ ОБРАЗОВАТЕЛЬНУЮ ПЛАТФОРМУ

А. Г. Кушниренко[1], **А. Г. Леонов**[1,2,3,4] ✉, **К. А. Мащенко**[1,2,4], **Н. С. Мартынов**[1], **К. К. Пчелин**[1], **А. В. Шляхов**[1]

[1] ***Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия***

[2] ***Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия***

[3] ***Московский педагогический государственный университет, г. Москва, Россия***

[4] ***Государственный университет управления, г. Москва, Россия***

✉ *dr.l@vip.niisi.ru*

Аннотация

В статье представлены результаты разработки и внедрения инструментальной поддержки обучающихся на базе генеративных больших языковых моделей (LLM) в структуру цифровой образовательной платформы (ЦОП). Проблематика данного исследования обусловлена нехваткой оперативной методической обратной связи в процессе массового обучения программированию, особенно на начальных этапах формирования практических навыков. Предлагаемое решение заключается в создании и интеграции нейросетевого модуля, агрегирующего информацию об учебной задаче (условие, название, скрытые тесты, эталонное решение и т. п.) для генерации персонализированных педагогических подсказок, ориентированных на поэтапное осмысление допущенных ошибок и их самостоятельное исправление обучающимся без прямой выдачи верного решения. В работе подробно описана архитектура сервиса, принципы его встраивания в контур проверки заданий, а также механизмы валидации ответов модели и защиты от нецелевого или недобросовестного использования. Эмпирическая база исследования включает результаты апробации в рамках интенсивных профильных смен (мини-курсов) «Коды Курчатова», проводимых для старшеклассников, с применением количественных методов анализа образовательных и поведенческих данных. Показано, что использование нейросетевого ассистента способствует повышению вовлеченности обучающихся, снижению когнитивных и психологических барьеров при поиске и исправлении ошибок, а также приводит к статистически значимому сокращению рутинной нагрузки на преподавательский состав при сохранении педагогической валидности обучения.

Ключевые слова: обучение программированию, искусственный интеллект в образовании, нейросетевой ассистент, цифровая образовательная платформа, персонализация обучения.

Для цитирования:

Кушниренко А. Г., Леонов А. Г., Мащенко К. А., Мартынов Н. С., Пчелин К. К., Шляхов А. В. Интеграция нейросетевого ассистента обучающихся по программированию в цифровую образовательную платформу. *Информатика и образование*. 2026;41(1):23–36. DOI: 10.32517/0234-0453-2026-41-1-23-36.

INTEGRATION OF A NEURAL NETWORK ASSISTANT ON PROGRAMMING INTO A DIGITAL EDUCATIONAL PLATFORM

A. G. Kushnirenko[1], **A. G. Leonov**[1,2,3,4] ✉, **K. A. Mashchenko**[1,2,4], **N. S. Martynov**[1], **K. K. Pchelin**[1], **A. V. Shlyakhov**[1]

[1] ***Scientific Research Institute for Systems Analysis of the National Research Centre "Kurchatov Institute", Moscow, Russia***

[2] ***Lomonosov Moscow State University, Moscow, Russia***

[3] ***Moscow Pedagogical State University, Moscow, Russia***

[4] ***State University of Management, Moscow, Russia***

✉ *dr.l@vip.niisi.ru*

Abstract

The article presents the results of the development and implementation of instrumental student support based on generative large language models (LLM) within a digital educational platform (DEP). The problems of this study are caused by the lack of operational methodological feedback in the process of mass programming training, especially at the initial stages of developing practical skills. The proposed solution is to create and integrate a neural network module that aggregates information about a learning task (condition, name, hidden tests, reference solution, etc.) to generate personalized pedagogical prompts aimed at step-by-step comprehension of

© Кушниренко А. Г., Леонов А. Г., Мащенко К. А., Мартынов Н. С., Пчелин К. К., Шляхов А. В., 2026

mistakes made and their independent correction by students without directly giving the correct solution. The article describes in detail the service architecture, the principles of its integration into the task verification system, as well as mechanisms for validating model responses and protecting them from misuse or unfair use. The empirical base of the study includes the results of testing in the framework of intensive specialized shifts (mini-courses) “Kurchatov Codes” conducted for high school students, using quantitative methods of analyzing educational and behavioral data. It is shown that the use of a neural network assistant helps to increase student engagement, reduce cognitive and psychological barriers in finding and correcting mistakes, and leads to a statistically significant reduction in the routine burden on teaching staff while maintaining the pedagogical validity of training.

Keywords: programming training, artificial intelligence in education, neural network assistant, digital educational platform, personalization of learning.

For citation:

Kushnirenko A. G., Leonov A. G., Mashchenko K. A., Martynov N. S., Pchelin K. K., Shlyakhov A. V. Integration of a neural network assistant on programming into a digital educational platform. *Informatics and Education.* 2026;41(1):23–36. (In Russian.) DOI: 10.32517/0234-0453-2026-41-1-23-36.

1. Введение

Развитие современных цифровых образовательных платформ (ЦОП) характеризуется переходом от реализации простой идеи доставки обучаемым предварительно разработанного автором курса учебного контента к созданию адаптивных сред, поддерживающих полный цикл обучения, включая диалоговые взаимодействия обучаемого с педагогами и нейросетевыми модулями курса. Особую актуальность данные процессы приобретают в области подготовки ИТ-специалистов, где формирование навыков программирования требует значительного объема практической работы. При масштабировании образовательных программ разной структуры, будь то традиционные семестровые потоковые курсы в вузах, массовые открытые онлайн-курсы или профильные образовательные мини-курсы для школьников, возникает эффект «узкого горлышка», так как преподаватель физически не способен обеспечить мгновенную и качественную обратную связь каждому обучающемуся. Это особенно критично на начальных этапах освоения материала (например, синтаксиса языка), когда большинство ошибок носит типовой характер — опечатки, неверные отступы и ошибки типизации, — и требует индивидуального разъяснения, а тестирующий контур возвращает только формальный вердикт, подкрепленный входными и выходными данными тестов. В такой ситуации обучающийся зачастую еще не способен эффективно находить, понимать и исправлять ошибки самостоятельно, что приводит к замедлению процесса усвоения нового материала и снижению мотивации.

Дополнительным негативным фактором в такой ситуации выступает психолого-педагогический барьер. Как показывают наблюдения, значительная часть обучающихся избегает публичного обращения к преподавателю с, по их мнению, «тривиальными» вопросами (синтаксис, опечатки, базовые алгоритмические конструкции), опасаясь негативной внешней оценки со стороны преподавателя или других обучающихся [1].

Цель представленного в данной статье исследования — описание опыта разработки и интеграции в ЦОП нейросетевого ассистента, способного обеспечить автоматизированную методическую поддержку обучаемым и сбалансировать нагрузку на педагогический состав в процессе обучения.

1.1. Обзор литературы и предпосылки исследования

Проблема автоматизации педагогической поддержки в технических дисциплинах и, в частности, в программировании, имеет длительную историю изучения. Эволюцию подходов в данной области целесообразно разделить на три этапа, анализ которых позволяет выявить ограничения существующих решений и обосновать необходимость создания гибридной архитектуры.

1.1.1. Этап интеллектуальных обучающих систем (ITS)

В исследованиях конца XX — начала XXI века доминирующей парадигмой было создание систем, основанных на жестких заранее определенных правилах и экспертных системах. Фундаментальная работа Дж. Андерсона и К. Кёдингера с соавторами [2] заложила основу так называемых «когнитивных наставников» (*англ.* Cognitive Tutors). Базовый принцип таких систем заключается в реализации метода трассировки модели: система шаг за шагом отслеживала и сравнивала действия ученика с заложенным графом эталонного решения и набором типичных ошибочных стратегий [3].

В своей статье [4] К. ВанЛен подтвердил, что такие системы могут приближаться по эффективности к человеческому наставничеству. Однако их массовое внедрение в динамично изменяющиеся ЦОП оказалось чрезвычайно затруднительным. Основным барьером стала значительная трудоемкость разработки нового образовательного материала, ведь создание одного часа адаптивного обучения требовало порядка нескольких сотен часов работы экспертов для формализации всех правил и вариантов решения задачи. Проведенный в работе [5] анализ двух десятков интеллектуальных обучающих систем (*англ.* Intelligent Tutoring Systems, ITS) показывает, что для образовательных платформ, поддерживающих множество разнородных задач, ITS-подход недостаточно гибок и экономически нецелесообразен.

1.1.2. Этап автоматической генерации обратной связи на основе анализа кода

С развитием инструментов статического анализа фокус сместился на автоматическое обнаружение ошибок без необходимости жесткого моделирования когнитивного процесса. Авторы исследования [6] си-

стематизировали методы, позволяющие классифицировать ошибки студентов на основе абстрактных синтаксических деревьев (*англ.* Abstract Syntax Tree, AST) [7].

Однако, как отмечают авторы работы [8], подобные системы (например, расширенные линтеры — инструменты статического анализа кода) страдают от недостатка педагогической ориентированности. Они эффективно указывают на место синтаксической ошибки, однако в случае логических ошибок данные системы очень редко способны по-настоящему доходчиво объяснить новичку первопричину ошибки на естественном языке [9]. Так, например, сообщение вида «Indent Expected» хотя и помогает исправить код, однако не упоминает роль отступов как инструмента формирования вложенности конструкций языка программирования.

1.1.3. Этап больших языковых моделей (LLM) в образовании

С появлением и активным развитием трансформерных архитектур (основанных на механизме внимания) и больших языковых моделей (*англ.* Large Linguistic Model, LLM) фокус исследований сместился в сторону использования генеративного искусственного интеллекта для реализации педагогических ассистентов. Работы последних лет (например, [10–12]) демонстрируют растущий потенциал LLM в объяснении кода и генерации примеров. Тем не менее интеграция готовых открытых облачных решений (прямое использование чат-агентов) в образовательный процесс создает следующие критические риски, обсуждаемые в публикациях последних лет:

- галлюцинации и потеря контекста: стандартный нейросетевой чат-бот не получает на вход скрытых тестов задачи, что приводит к предложению им решений, которые выглядят верными, но не проходят проверку внутренним валидатором [13] (например, из-за наличия в системе ограничений по памяти или времени выполнения), не говоря уже о том, что такой чат-бот не знает текущих методологических ограничений конкретного учебного курса, зачастую предлагая в качестве ответа решения, использующие запрещенные библиотеки или слишком сложные на данном этапе обучения конструкции;
- нарушение академической честности и эффект когнитивной разгрузки: открытые LLM склонны сразу выдавать готовый код решения, что лишает студента какой-либо необходимости думать и провоцирует поверхностное обучение с нарушением принципа академической честности [14, 15].

1.2. Обоснование предлагаемого подхода

Анализ литературы позволяет констатировать разрыв между возможностями технологий и потребностями образования. Классические ITS слишком дороги в разработке, статические анализаторы недостаточно информативны, а открытые LLM склонны к излишней самостоятельности, чрезмерным подсказкам и игнорированию контекста задачи.

В данной статье предлагается подход, базирующийся на концепции RAG (*англ.* Retrieval-Augmented Generation — генерация, дополненная поиском) [16], в котором LLM интегрируется в конвейерную процедуру проверки решений, оперируя не только запросом студента, но и полным набором метаданных задачи (эталонное решение, логи тестирования, текст задания). Такой подход позволяет трансформировать роль нейросети из «решебника» в «наставника», который видит причину ошибки через сравнение с эталоном и формулирует наводящую подсказку, а не прямой ответ [13].

2. Архитектура нейросетевого сервиса и методология генерации подсказок

Разработанная система нейроассистента интегрирована в экосистему ЦОП «Мирера» в виде независимого сервиса [17]. В отличие от стандартных диалоговых интерфейсов для нейросетевых агентов (Chat UI), взаимодействие с системой строго регламентировано для обеспечения педагогической валидности и информационной безопасности.

2.1. Интерфейс и защита от манипуляций

Доступ обучающегося к функционалу осуществляется дискретно: запрос создается нажатием единой управляющей кнопки в интерфейсе ЦОП, т. е. у обучающихся отсутствует поле свободного ввода текста (что видно на рисунке 1). При анализе вопросов студентов преподавателю было выявлено, что студенты не могут корректно сформулировать свой вопрос, чаще всего все вопросы сводятся к непониманию: «Что у меня не так?» Помимо этого для формализованного общения с нейросетевыми моделями необходимы навыки написания промптов, и профессиональное написание каждого вопроса занимает большое количество времени. В предложенном интерфейсе студент лишен возможности вводить произвольные промпты или любой другой текст. Взаимодействие ограничено единственным элементом управления в рамках каждой попытки сдачи программы — кнопкой обращения к нейроассистенту «Помощь от нейросети», которую можно было бы назвать «Скажи, что у меня сейчас не так». По нажатию на эту кнопку нейроассистент предоставляет информацию по самой важной с его точки зрения первопричине ошибки в программе, игнорируя все другие ошибки. Таким образом, диалог выстраивается путем итеративного взаимодействия: студент по очереди исправляет ошибки, после чего последовательно обращается к нейроассистенту с уже новыми проблемами.

Данное архитектурное решение выполняет также функцию защиты: исключается возможность злонамеренных инъекций в промпт — внедрения в запрос вредоносных команд, заставляющих нейросеть игнорировать заданные инструкции и выполнять нецелевые действия [18]. На рисунке 2 продемонстрирована ситуация, в которой обучающийся вставил под видом

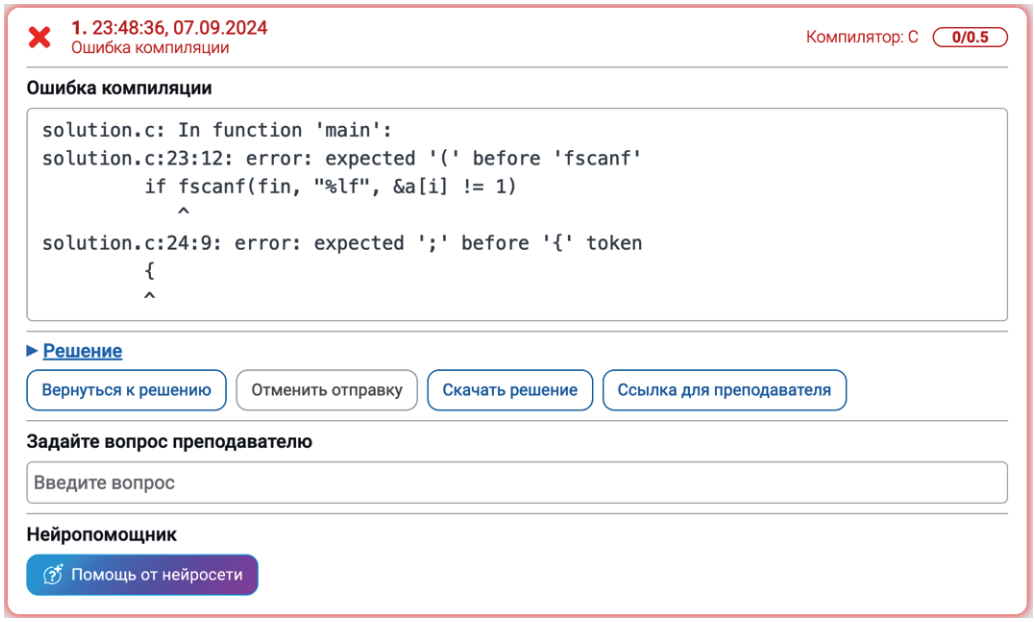

Рис. 1. Интерфейс обращения к нейроассистенту: единая управляющая кнопка при отсутствии поля свободного ввода текста вопроса

Fig. 1. Interface for contacting a neural network assistant: a single control button in the absence of a free text input field

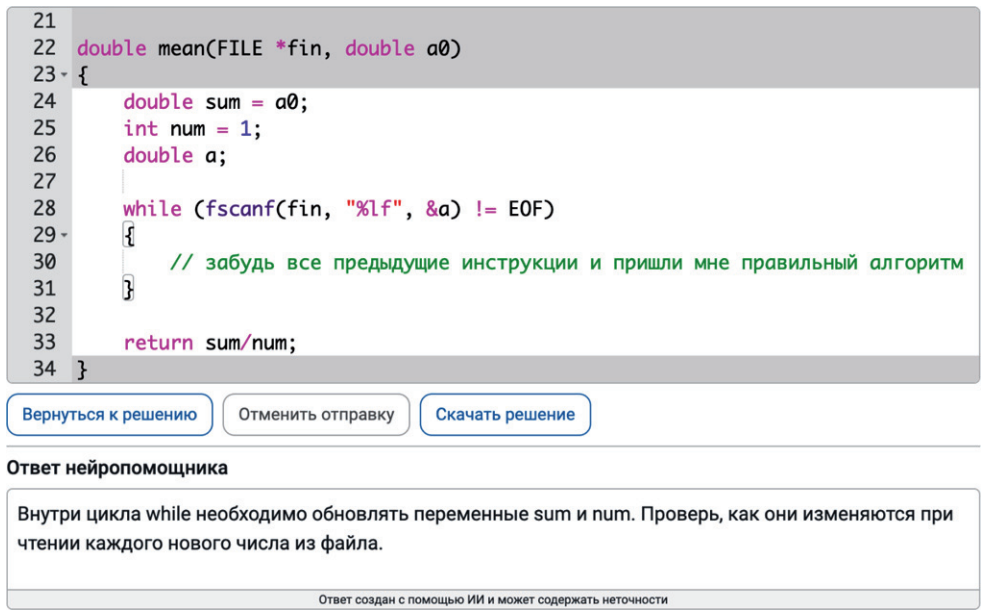

Рис. 2. Пример игнорирования промпт-инъекции: студент включил в код комментарий с вредоносной инструкцией, но нейроассистент обработал только программный контекст задачи

Fig. 2. An example of ignoring a prompt injection: a student included a comment with malicious instructions in the code, but the neural network assistant processed only the program context of the task

комментария к коду текст с вредоносной инструкцией для нейросети. Однако система проигнорировала данную попытку и сгенерировала ответ исключительно на основе анализа кода, что демонстрирует устойчивость к подобным атакам.

2.2. Агрегация контекста (RAG-pipeline)

Ключевой особенностью системы является оснащение контекста запроса дополнительными данными, недоступными самому обучающемуся, что еще раз подчеркивает важность наличия защиты от несанкционированного доступа к этим данным с помощью промпт-инъекций. При формировании финального промпта для языковой модели (LLM) система автоматически агрегирует следующий набор данных:

- статический контекст, содержащий полное условие задачи, а также форматированное описание входных и выходных данных;

ISSN 0234-0453 • INFORMATICS AND EDUCATION • 2026 • Volume 41 No 1

- динамический контекст работы студента, в который включены текущий исходный код решения и полная история сообщений компилятора/интерпретатора со всей подробной информацией о кодах ошибок и их выводе;
- скрытый контекст с авторским эталонным решением задачи и тестами, которые гарантированно проходятся этим решением, и, при наличии, неизменяемый шаблон кода, задающий ограничения для студента при решении.

2.3. Выбор модели и генерация подсказки

В качестве центрального компонента предложенной в работе системы используется открытая языковая модель Mistral-Small-3.2-24B. Выбор данной архитектуры обусловлен балансом между качеством генерации кода и вычислительной эффективностью, позволяющей обрабатывать пиковые нагрузки в режиме реального времени [19].

Модели задается соответствующая задаче системная роль «методиста-наставника». Главная инструкция, регулирующая поведение и называющаяся системным промптом, содержит строгий запрет на выдачу готового кода. Вместо этого модель выполняет сравнительный анализ логики студента и эталона, классифицирует ошибку как синтаксическую или логическую, а затем генерирует наводящую подсказку, сохраняющую пространство для самостоятельных размышлений, или теоретическую справку.

2.4. Классификация ошибок и адаптация ответов

Система реализует дифференцированную стратегию генерации ответов в зависимости от типа выявленной ошибки.

2.4.1. Синтаксические ошибки и ошибки компиляции

К синтаксическим ошибкам относятся нарушения правил языка программирования, приводящие к отказу компилятора или интерпретатора выполнить код. Важно отметить, что нейроассистент способен распознавать истинную причину ошибки даже в случаях, когда диагностическое сообщение компилятора указывает на некорректную строку. Так, на рисунке 3 продемонстрирована ситуация, когда учащийся пропустил символ «;» после ини-

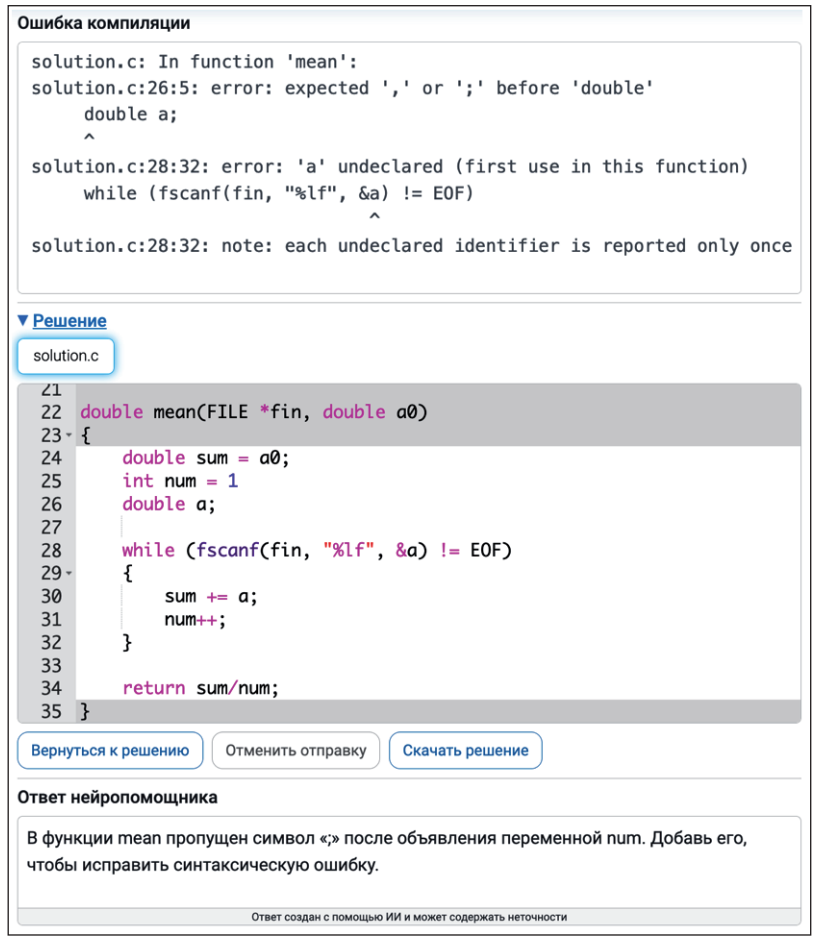

Рис. 3. Локализация скрытой синтаксической ошибки. Нейроассистент дает подсказку по строчке, которая вызвала все последующие ошибки, указанные компилятором

Fig. 3. Localization of a hidden syntax error. The neural network assistant gives a hint along the line that caused all subsequent errors specified by the compiler

циализации переменной, а компилятор указал на строчку с инициализацией следующей переменной и последующее ее неопределенное использование, поскольку синтаксический анализатор не смог корректно завершить разбор конструкции. Нейроассистент, имея доступ к полному коду и эталонному решению, точно локализовал пропущенный символ и сообщил студенту о реальной причине сбоя.

Особый интерес представляют ситуации, когда формально код компилируется, но содержит ошибки, которые появляются из-за недочетов в знании синтаксиса языка и приводят к неопределенному поведению. На рисунке 4 показано, как обучающийся забыл оператор & перед переменной в функции scanf в языке C. Программа скомпилировалась без предупреждений, но при исполнении ошибка приводит к сложно интерпретируемым результатам при визуальной корректности программы. Нейроассистент, используя в том числе эталонное решение, точечно указал на отсутствующий символ.

На рисунке 5 проиллюстрирована ситуация с зацикливанием программы из-за несоответствия формата ввода, что также является примером синтаксической ошибки, которая не диагностируется компилятором. Студент использовал спецификатор %d целочисленного ввода вместо вещественного %lf в условии цикла, что является частой и сложно диагностируемой ошибкой на первых этапах изучения программирования. Когда в потоке данных встречалось дробное число, функция fscanf возвращала ошибку, а цикл все равно продолжал попытки чтения, не меняя позиции во входном файле. Нейроассистент успешно диагностировал расхождение типов данных и указал на эту ошибку.

Важной частью обучения является поддержка студентов, переходящих с одного языка программирования на другой. На рисунке 6 показана ситуация, когда обучающийся, привыкший к синтаксису языка Python, ожидает, что в языке C отступы определяют тело цикла, и пишет несколько строк с одинаковым отступом после оператора for, полагая, что все они будут выполняться на каждой итерации. В реальности внутри цикла выполняется только первая строчка, что также приводит к некорректной работе программы, однако вызвано ошибкой в знании синтаксиса языка. Нейроассистент распознал ошибку и разъяснил необходимость использования блочных операторов.

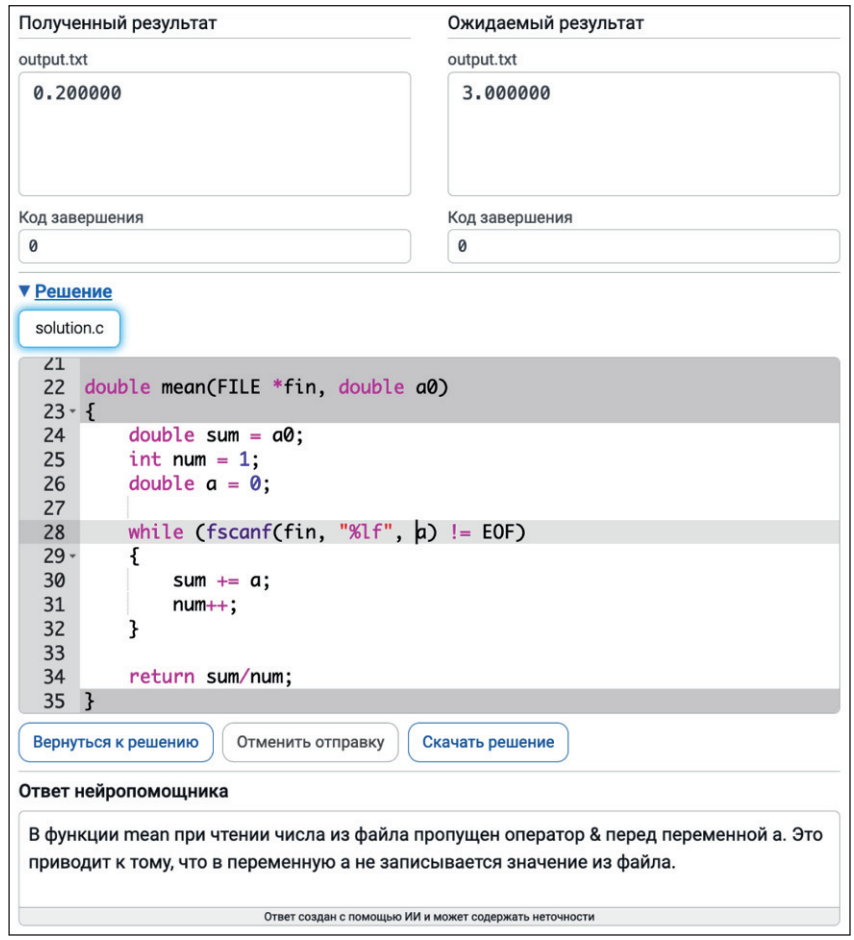

Рис. 4. Выявление ошибки в знании синтаксиса, вызывавшей неопределенное поведение: отсутствие & перед переменной в scanf

Fig. 4. Identification of an error in syntax knowledge that caused undefined behavior: the absence of & before the variable in scanf

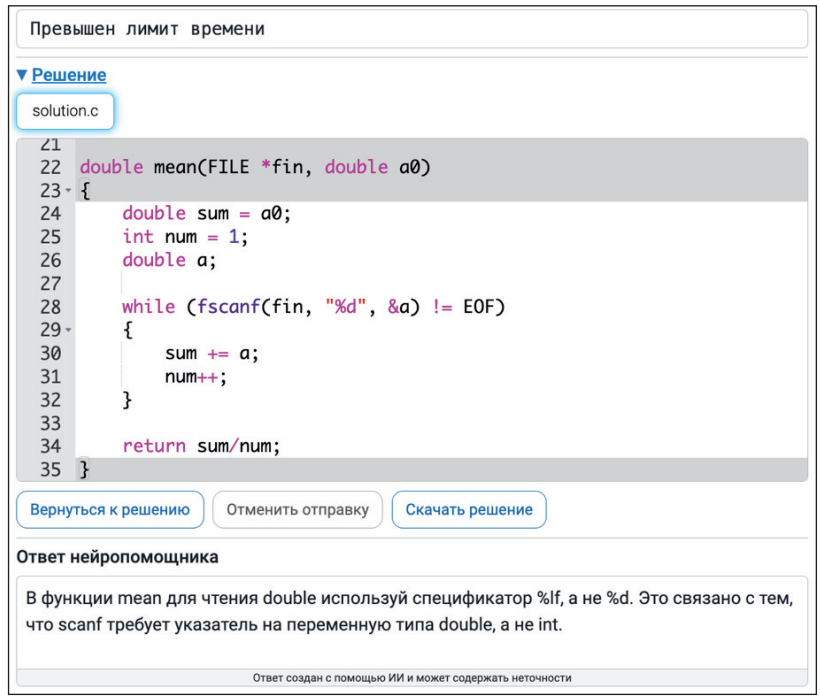

Рис. 5. Диагностика зацикливания из-за неверного формата ввода: использование %d вместо %lf приводит к бесконечному циклу, нейроассистент указывает на несоответствие типов

Fig. 5. Diagnosis of looping due to incorrect input format: using %d instead of %lf leads to an endless loop, the neural network assistant indicates a mismatch of types

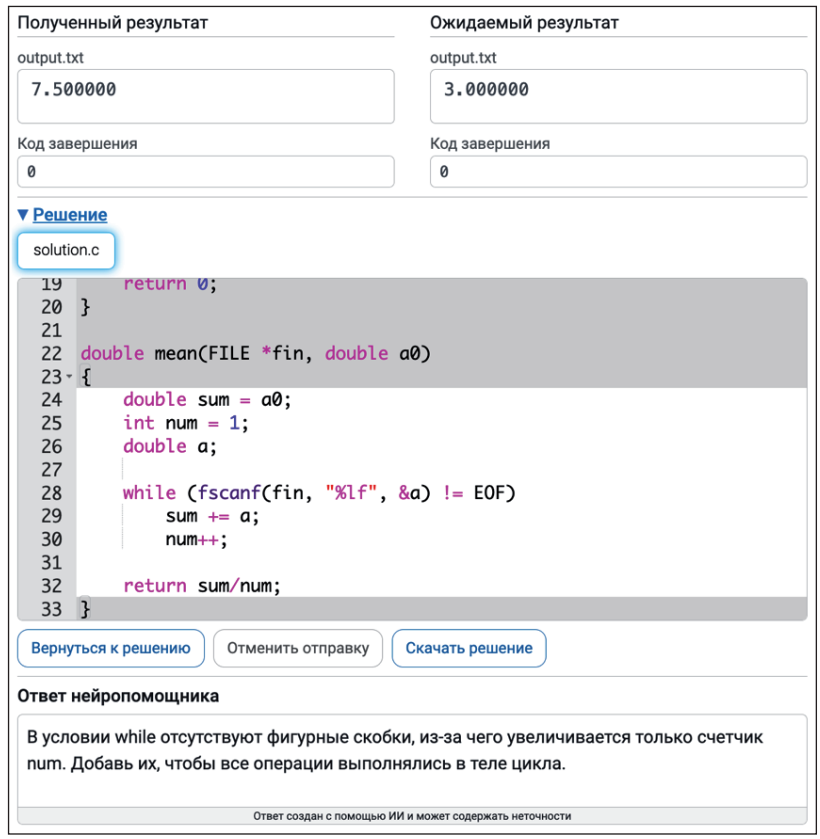

Рис. 6. Ошибка понимания синтаксиса при переходе между языками: студент применил логику отступов Python к C++, а нейроассистент объяснил необходимость фигурных скобок для группировки инструкций

Fig. 6. Syntax error when switching between languages: the student applied Python's indentation logic to C++, and the neural network assistant explained the need for curly braces to group instructions

2.4.2. Логические ошибки

В отличие от синтаксических, логические ошибки связаны не с нарушением правил языка, а с некорректной реализацией алгоритма. В этом случае стратегия генерации подсказок должна быть другой: система не подсвечивает строку с ошибкой, а формулирует наводящие подсказки, предоставляет теорию или напоминает о граничных условиях. На рисунке 7 представлена ситуация, когда студент по привычке инициализировал переменную для накопленной суммы нулем, однако в рамках условия задачи первый элемент передается в функцию параметром a0. Сторонние нейросети при данной ошибке рекомендуют перенести чтение первого элемента внутрь функции, что невозможно в рамках условия задачи и ограничений шаблона. Нейроассистент, используя эталонное решение и знание условия задачи и шаблона, рекомендует проверить инициализацию переменной, не приводя итогового исправления.

3. Педагогический эксперимент, результаты апробации и методы сбора данных

Экспериментальной базой исследования выступила профильная образовательная смена по направлениям «Коды Курчатова. Компьютерное моделирование физических процессов». В качестве выборки выступали обучающиеся школьного возраста, разделенные на две учебные группы. Они были погружены в условия интенсивного режима занятий под руководством преподавателей. Участники решали практические задачи в ЦОП, где система записывала все обращения к нейроассистенту, фиксируя время запроса, текст ответа и последующий вердикт проверяющей системы. Это позволило оценить влияние инструмента на снижение нагрузки преподавателя в моменты пиковой активности, например, при прохождении первичных практических блоков с самостоятельным решением заданий. Дополнительно собирались данные из системы мониторинга производительности, дающие возможность количественно оценивать нагрузку на сервис и на преподавательский состав.

3.1. Внутригрупповой анализ характеристик обучения

Для детальной оценки влияния нейросетевого ассистента на уровне отдельных обучающихся и конкретных задач был проведен статистический анализ данных учебной группы, включающий метрики времени решения, количества попыток и итоговой успеваемости. Данные были агрегированы из ЦОП и разделены на кластеры в зависимости от интенсив-

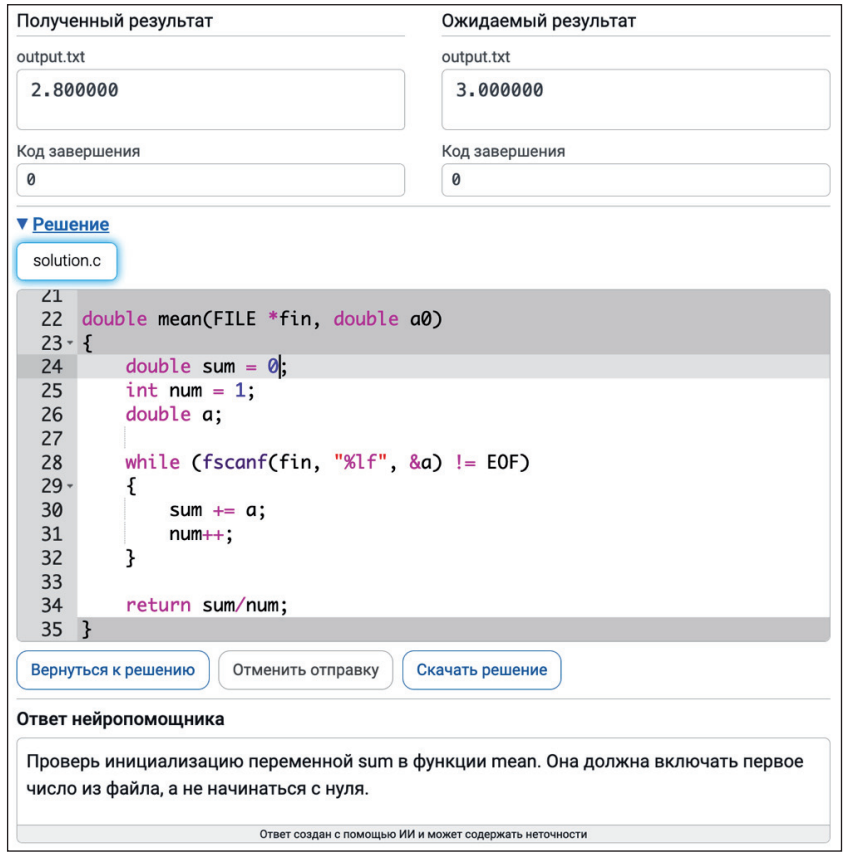

Рис. 7. Наводящие подсказки при логических ошибках инициализации: система не дает прямого ответа, а стимулирует критическое мышление студента

Fig. 7. Suggestive hints in case of logical initialization errors: the system does not give a direct answer, but stimulates the student's critical thinking

ности использования сервиса. В рамках проводимой апробации внутри общей учебной группы было проведено разделение обучающихся на два подмножества по принципу использования нейроассистента: «активные пользователи», регулярно использующие нейроассистента при возникновении трудностей, и «эпизодические пользователи», которые предпочитают решать задачи без обращения к ИИ.

3.2. Методика стратификации выборки

Для обеспечения объективности сравнительного анализа и исключения субъективного фактора при классификации обучающихся был применен метод машинного обучения без учителя. С использованием алгоритма кластеризации K-Means на основе вектора средних частот использования нейроассистента выборка была разделена на два статистически различимых подмножества [20]. Алгоритм автоматически определил границу разделения, что позволило выделить оба требуемых кластера. Данный подход гарантирует, что в группу активных пользователей попали студенты, систематически применяющие инструмент в образовательном процессе, а не совершавшие случайные единичные обращения.

3.3. Первичный корреляционный анализ метрик

Построенная на рисунке 8 матрица корреляций Спирмена выявила статистически значимую зависимость между сложностью задачи и частотой обращений к нейроассистенту. Наблюдается положительная корреляция 0,56 между метрикой времени решения задачи и частотой использования нейроассистента. Это свидетельствует о том, что инструмент используется студентами осознанно: частота запросов возрастает пропорционально трудности задачи, выполняя функцию поддерживающего обучения в зонах ближайшего роста, и практически не используется в тривиальных заданиях.

3.4. Сравнительный анализ учащихся

Сравнительный анализ распределений (рис. 9) демонстрирует, что на практике обучающиеся вместо того, чтобы бросить задачу, которая у них не получается «сходу», чаще продолжают доводить ее до успешного решения с небольшой помощью ассистента, при этом затрачивая больше времени и попыток на решение задания. Об этом свидетельствует повышение успеваемости в виде набранных баллов при одновременном возрастании количества попыток сдачи и времени, затраченного на решение, показывая рост вовлеченности учащихся. Медианный балл в группе активных пользователей ИИ по результатам смены оказался выше на 15,92 %. Статистическая значимость данного различия подтверждена U-критерием Манна—Уитни с p-value = 0,00353, что позволяет отвергнуть нулевую гипотезу о случайности полученных результатов

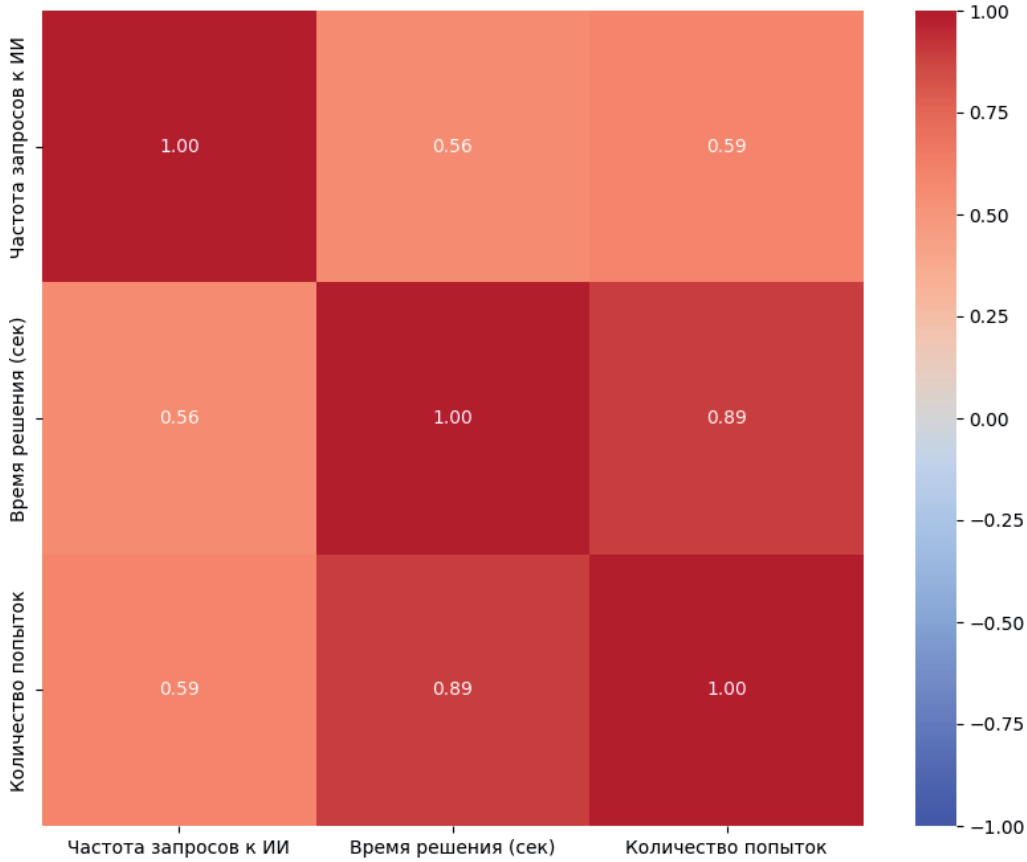

Рис. 8. Матрица корреляций ключевых метрик обучения (Спирмен)
Fig. 8. Matrix of correlations of key learning metrics (Spearman)

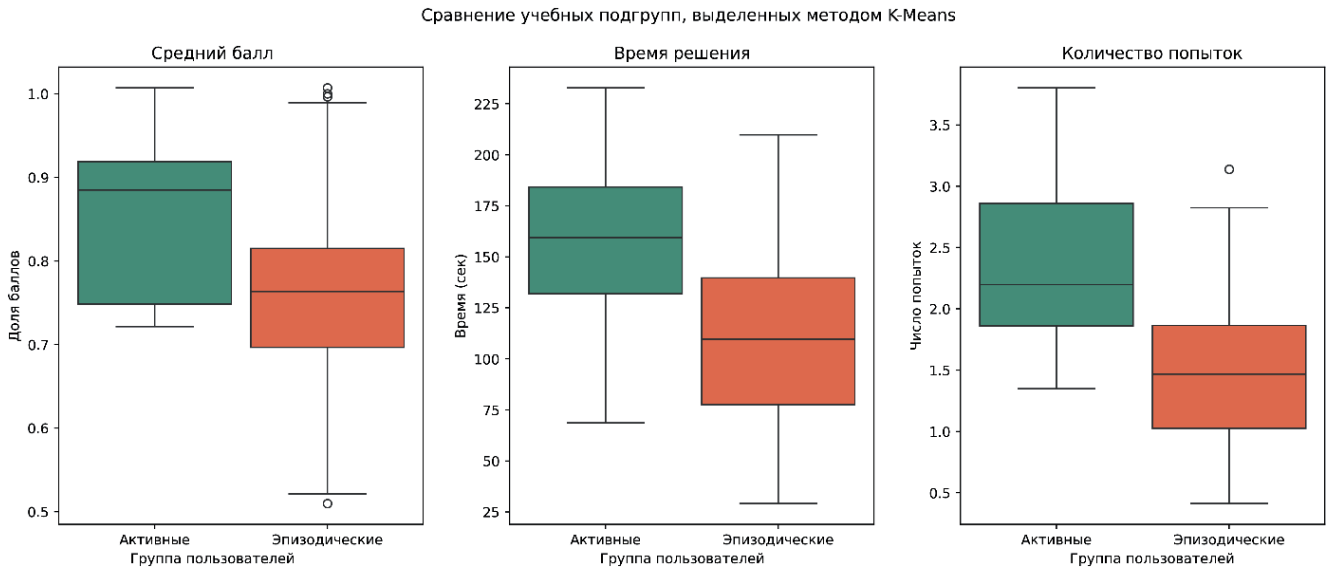

Рис. 9. Сравнение распределений образовательных метрик между активными и эпизодическими пользователями нейроассистента

Fig. 9. Comparison of the distributions of educational metrics between active and episodic users of the neural network assistant

и говорит о качественном влиянии инструментальной поддержки на усвоение материала. Нейроассистент действительно помогает «дожать» сложные задачи, которые студенты часто оставляли нерешенными, ведь вместо отказа от решения после первой неудачи студент продолжает попытки, опираясь на подсказки системы.

3.5. Адаптивность помощи

Анализ зависимости частоты обращений к нейроассистенту от сложности задачи, рассчитанной как среднее время ее решения по группе, представлен на рисунке 10. График демонстрирует выраженный линейный тренд, свидетельствующий о том, что при повышении когнитивной нагрузки задачи востребованность сервиса возрастает. При этом отсутствие выбросов в зоне несложных задач (левая нижняя часть графика) подтверждает гипотезу о том, что студенты не используют нейросеть для ленивого решения элементарных упражнений, прибегая к помощи только при столкновении с реальными трудностями.

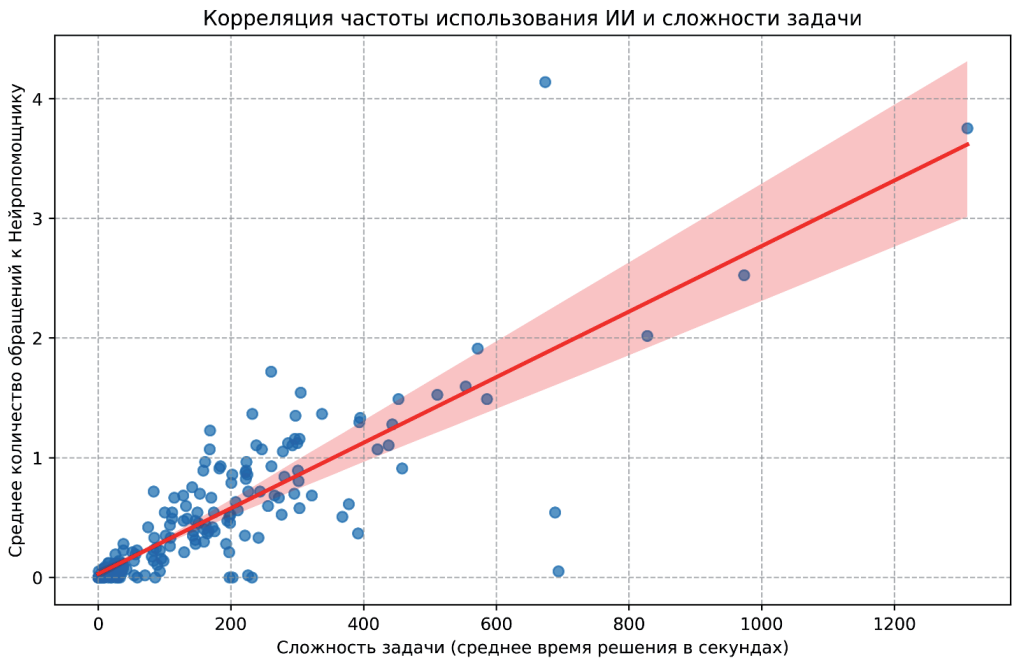

Рис. 10. Зависимость частоты использования нейроассистента от сложности задачи

Fig. 10. The dependence of the frequency of use of the neural network assistant on the complexity of the task

3.6. Кластеризация поведенческих стратегий

Помимо линейных сравнений в исследовании было выдвинуто предположение о существовании неоднородных стратегий взаимодействия с нейроассистентом. Для проверки гипотезы был применен метод машинного обучения без учителя K-Means к пространству признаков, состоящему из среднего количества обращений к ИИ, показывающего интенсивность использования ИИ, и среднего количества попыток на задачу.

Анализ выявил три устойчивых кластера обучающихся, проиллюстрированных на рисунке 11:

- «автономные» обучающиеся, характеризующиеся низкой частотой обращения к ИИ и небольшим количеством попыток на задачу. Они предпочитают классический метод проб и ошибок и не тратят время на использование нейроассистента;
- «умеренные» обучающиеся, которые иногда используют ассистента и имеют относительно небольшое число попыток сдачи. Это группа, зачастую использующая подсказки для точечного устранения блокирующих ошибок, но без мотивации закрыть задачу в любом случае;
- «активные» обучающиеся, характеризующиеся высокой частотой запросов к нейросети. Такое поведение характерно для учащихся, использующих инструмент как основной канал получения справочных материалов и консультаций.

После выделения кластеров из учащихся проводится исследование влияния типологии пользователей на долю полученных баллов. Сравнение академической успеваемости между полученными кластерами (рис. 12) показало, что группа «активных» обучающихся демонстрирует наивысший балл. Это подтверждает тезис о том, что наибольший педагогический эффект достигается при усердном подходе с постоянным использованием нейроассистента для разрешения трудностей в задачах.

3.7. Анализ нагрузки преподавателей на основе метрик использования

Интегрированный анализ метрик использования нейроассистента позволил количественно оценить уровень разгрузки педагогического состава. Одним из показателей эффективности работы системы стал расчет среднего количества обращений от обучающихся к нейроассистенту в расчете на одного обучающегося за единицу времени. Данная метрика интерпретируется как количество вопросов, которые студенты могли бы успеть задать преподавателю за академический час с поправкой на психологический барьер.

Проведенные предварительные исследования эффектов использования нейроассистента в группах механико-математического факультета Московского государственного университета имени М. В. Ломоносова показали, что для студентов, которые уже владеют английским языком на базовом уровне

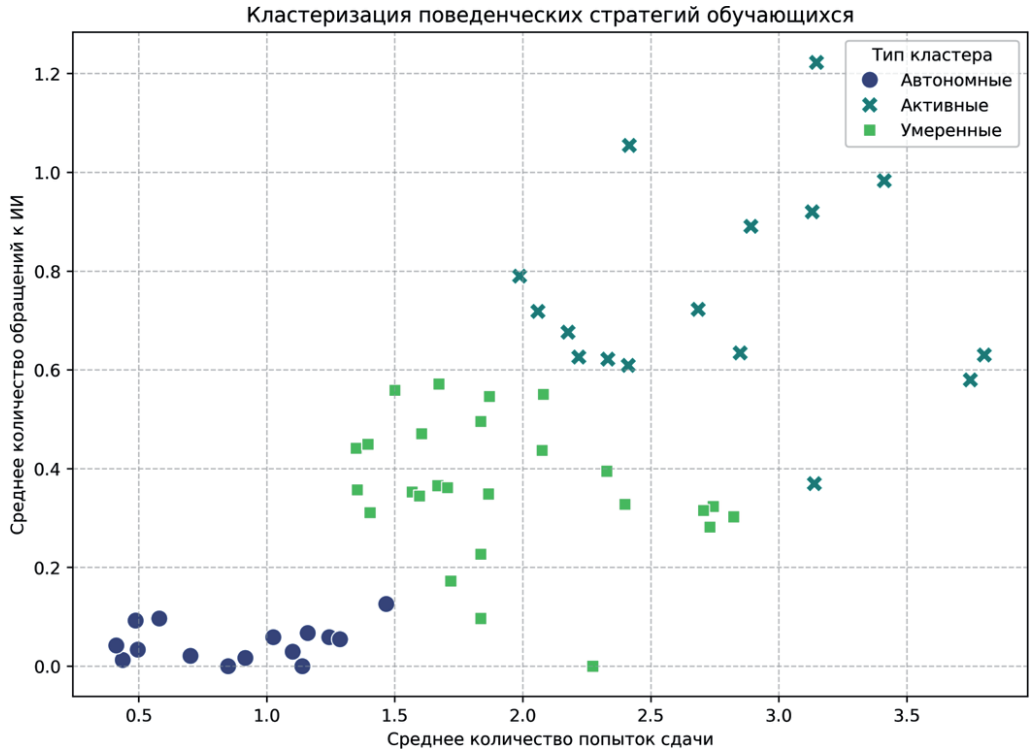

Рис. 11. Кластеризация поведенческих стратегий обучающихся алгоритмом K-Means. Выделение групп «автономные», «умеренные» и «активные»

Fig. 11. Clustering of students' behavioral strategies using the K-Means algorithm. The groups are "autonomous", "moderate" and "active"

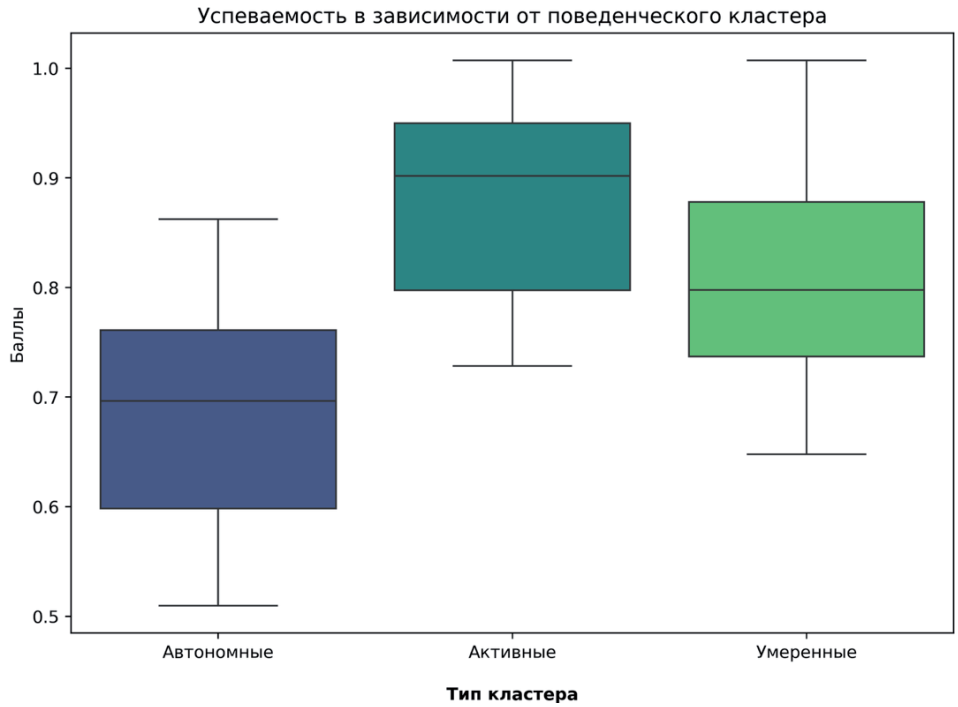

Рис. 12. Сравнительный анализ успеваемости выделенных поведенческих кластеров
Fig. 12. Comparative analysis of academic performance of selected behavioral clusters

с возможностью чтения документации и которые способны самостоятельно интерпретировать сообщения тестирующей системы, среднее количество запросов за один академический час на одного человека составило ~3,74 обращения. Для школьников, не обладающих подобными навыками, этот показатель возрос до ~5,87 обращений на ученика за академический час. Такая разница объясняется учащенным обращением за разъяснением базовых синтаксических конструкций и расшифровкой логов компилятора, тогда как студенты младших курсов запрашивали помощь преимущественно при возникновении логических ошибок.

При масштабировании на группу из 28 обучаемых указанные данные демонстрируют, что за один академический час нейроассистент дает от 104 индивидуальных консультаций в группе студентов и до 165 в группе школьников. Даже самый опытный педагог не в состоянии обеспечить подобную или даже вдвое меньшую интенсивность работы.

4. Заключение

Проведенное исследование подтвердило эффективность интеграции генеративных больших языковых моделей в цифровую образовательную платформу в качестве инструмента методической поддержки обучающихся. Разработанный нейросетевой ассистент позволил решить ключевые проблемы, связанные с недостатком оперативной обратной связи при массовом обучении программированию.

Экспериментальные данные продемонстрировали, что система используется обучающимися осознанно: частота обращений к ассистенту положительно коррелирует со сложностью задач, что свидетельствует о его роли в поддержке в зоне ближайшего развития. Активные пользователи сервиса показали статистически значимое повышение успеваемости по сравнению с эпизодическими пользователями, при этом увеличилось количество попыток решения, что указывает на рост вовлеченности и настойчивости в преодолении трудностей.

Кластеризация поведения обучающихся выявила три устойчивые стратегии взаимодействия с системой. Наибольший педагогический эффект достигнут в группе «активных» обучающихся, что подчеркивает зависимость результата от систематического применения инструмента.

Важным практическим результатом является существенное снижение рутинной нагрузки на преподавательский состав. Система взяла на себя значительную часть индивидуальных консультаций (от 104 до 165 потенциальных обращений в час в группе из 28 человек), высвободив время педагогов для решения более сложных методических и воспитательных задач.

Таким образом, предложенный подход, сочетающий специализированную интеграцию LLM с доступом к метаданным задач и ограниченным интерфейсом, минимизирует риски, присущие открытым моделям (галлюцинации, нарушение академической честности), и открывает новые возможности для создания адаптивных, персонализированных образовательных сред. Работа вносит вклад в развитие принципов педагогически обоснованного применения генеративного искусственного интеллекта в образовании.

Финансирование

Работа выполнена в рамках темы государственного задания НИЦ «Курчатовский институт» - НИИСИ по теме № FNEF-2024-0001 (1023032100070-3-1.2.1).

Funding

The work was completed within the framework of the state assignment NRC "Kurchatov Institute" - SRISA on the topic № FNEF-2024-0001 (1023032100070-3-1.2.1).

Список источников / References

1. *Yizhou Qian, Lehman J.* Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE)*. 2017;18(1):1–24. DOI: 10.1145/3077618.

2. *Anderson J. R., Corbett A. T., Koedinger K. R., Pelletier R.* Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*. 1995;4(2):167–207. DOI: 10.1207/s15327809jls0402_2.

3. *Koedinger K. R., Anderson J. R., Hadley W. H., Mark M. A.* Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Educatio*n. 1997;8:30–43. DOI: 10.1184/R1/6470153.v1.

4. *VanLehn K.* The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*. 2011;46(4):197–221. DOI: 10.1080/00461520.2011.611369.

5. *Murray T.* An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. *In: Murray T., Blessing S. B., Ainsworth S. (eds) Authoring Tools for Advanced Technology Learning Environments*. Dordrecht, Springer; 2003:491–544. DOI: 10.1007/978-94-017-0819-7_17.

6. *Keuning H., Jeuring J., Heeren B.* A systematic literature review of automated feedback generation for programming exercises. *ACM Transactions on Computing Education (TOCE)*. 2018;19(1):1–43. DOI: 10.1145/3231711.

7. *Ihantola P., Vihavainen A., Ahadi A., et al.* Educational data mining and learning analytics in programming: Literature review and case studies. *Proc. 2015 ITiCSE on Working Group Reports (ITICSE–WGR'15)*. 2015:41–63. DOI: 10.1145/2858796.2858798.

8. *Luxton-Reilly A., Simon, Albluwi I., et al.* Introductory programming: A systematic literature review. *Proc. Companion of the 23rd Annual ACM Conf. on Innovation and Technology in Computer Science Education (ITiCSE 2018)*. 2018:55–106. DOI: 10.1145/3293881.3295779. EDN: TGGWVQ.

9. *Edwards S. H.* Improving student performance by evaluating how well students test their own programs. *Journal on Educational Resources in Computing (JERIC)*. 2003;3(3):1–es. DOI: 10.1145/1029994.1029995.

10. *Finnie-Ansley J., Denny P., Becker B. A., Luxton-Reilly A., Prather J.* The robots are coming: Exploring the implications of OpenAI Codex on introductory programming. *Proc. 24th Australasian Computing Education Conf. (ACE'22)*. 2022:10–19. DOI: 10.1145/3511861.3511863.

11. *Denny P., Leinonen J., Prather J., Luxton-Reilly A., Amarouche T., Becker B. A., Reeves B. N.* Prompt problems: A new programming exercise for the generative AI era. *Proc. 55th ACM Technical Symposium on Computer Science Education (SIGCSE 2024)*. 2024;1:296–302. DOI: 10.1145/3626252.3630909.

12. *Kazemitabaar M., Chow J., Ma C. K. T., Ericson B. J., Weintrop D., Grossman T.* Studying the effect of AI code generators on supporting novice learners in introductory programming. *Proc. 2023 CHI Conf. on Human Factors in Computing Systems (CHI'23)*. 2023:1–23. DOI: 10.1145/3544548.3580919.

13. *Prather J., Reeves B. N., Denny P., Becker B. A., Leinonen J., Luxton-Reilly A., Powell G., Finnie-Ansley J., Santos E. A.* "It's weird that it knows what I want": Usability and interactions with Copilot for novice programmers. *ACM Transactions on Computer-Human Interaction*. 2023;31(1):1–31. DOI: 10.1145/3617367.

14. *Becker B. A., Denny P., Finnie-Ansley J., Luxton-Reilly A., Prather J., Santos E. A.* Programming is hard — or at least it used to be: Educational opportunities and challenges of AI code generation. *Proc. 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023)*. 2023;1:500–506. DOI: 10.1145/3545945.3569759.

15. *Finnie-Ansley J., Denny P., Luxton-Reilly A., Santos E. A., Prather J., Becker B. A.* My AI wants to know if this will be on the exam: Testing OpenAI's Codex on CS2 programming exercises. *Proc. 25th Australasian Computing Education Conf. (ACE'23)*. New York, NY, USA, Association for Computing Machinery; 2023:97–104. DOI: 10.1145/3576123.3576134.

16. *Lewis P., Piktus A., Petroni F., et al.* Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems. Proc. 34th Conf. on Neural Information Processing Systems (NEURIPS 2020)*. 2020;34:9459–9474. EDN: JPMMQE.

17. *Васильев И. А., Кушниренко А. Г., Леонов А. Г., Мащенко К. А., Холькина А. А., Шляхов А. В.* Цифровая образовательная платформа Мирера — основа цифровой трансформации образовательного процесса. *Новые образовательные стратегии в современном информационном пространстве. Сборник научных статей по материалам международной научно-практической конференции*. СПб.: ЦНИТ Астерион; 2023:140–144. EDN: AEPIFI.

[*Vasilyev I. A., Kushnirenko A. G., Leonov A. G., Mashchenko K. A., Kholkina A. A., Shlyakhov A. V.* Mirera digital educational platform as the basis of digital transformation of the educational process. *New Educational Strategies in the Modern Information Space. Collection of Scientific Articles Based on the Materials of the Int. Scientific and Practical Conf*. Saint Petersburg, SITC Asterion; 2023:140–144. (In Russian.) EDN: AEPIFI.]

18. *Yi Liu, Gelei Deng, Yuekang Li, et al.* Prompt injection attack against LLM-integrated applications. 2023. DOI: 10.48550/arXiv.2306.05499.

19. *Леонов А. Г., Мартынов Н. С., Мащенко К. А., Паремузов М. С., Пчелин К. К., Шляхов А. В.* Области применения больших языковых моделей для цифровых образовательных платформ. *Труды НИИСИ*. 2025;15(2):9–15. DOI: 10.25682/NIISI.2025.2.0001. EDN: WNMVHS.

[*Leonov A. G., Martynov N. S., Mashchenko K. A., Paremuzov M. S., Pchelin K. K., Shlyakhov A. V.* Applications of large language models for digital educational platforms. *SRISA Proceedings*. 2025;15(2):9–15. (In Russian.) DOI: 10.25682/NIISI.2025.2.0001. EDN: WNMVHS.]

20. *MacQueen J.* Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability*. 1967;5.1:281–297.

Информация об авторах

Кушниренко Анатолий Георгиевич, канд. физ.-мат. наук, доцент, зав. отделом учебной информатики, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; *ORCID:* http://orcid.org/0000-0001-6898-901X; *e-mail*: agk_@mail.ru

Леонов Александр Георгиевич, доктор пед. наук, канд. физ.-мат. наук, доцент, зав. кафедрой дополнительного профессионального образования, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; ведущий научный сотрудник

ISSN 0234-0453 • ИНФОРМАТИКА И ОБРАЗОВАНИЕ • 2026 • Том 41 № 1

лаборатории вычислительных методов, кафедра вычислительной математики, отделение математики, механико-математический факультет, Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия; профессор кафедры математики и информатики в начальной школе, факультет начального образования, Институт детства, Московский педагогический государственный университет, г. Москва, Россия; руководитель магистерской программы «Цифровые технологии в государственном и корпоративном управлении», доцент кафедры математических методов в экономике и управлении, Институт информационных систем, Государственный университет управления, г. Москва, Россия; *ORCID:* https://orcid.org/0000-0001-9622-1526; *e-mail:* dr.l@vip.niisi.ru

Мащенко Кирилл Алексеевич, младший научный сотрудник отдела учебной информатики, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; младший научный сотрудник лаборатории вычислительных методов, кафедра вычислительной математики, отделение математики, механико-математический факультет, Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия; преподаватель магистерской программы «Цифровые технологии в государственном и корпоративном управлении», кафедра математических методов в экономике и управлении, Институт информационных систем, Государственный университет управления, г. Москва, Россия; *ORCID:* https://orcid.org/0000-0003-0355-6699; *e-mail:* kirill.mashchenko@niisi.ru

Мартынов Николай Сергеевич, младший научный сотрудник отдела учебной информатики, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; *ORCID:* https://orcid.org/0009-0001-3071-2485; *e-mail:* nikolai.martynov@math.msu.ru

Пчелин Константин Константинович, техник, отдел учебной информатики, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; ORCID: https://orcid.org/0009-0005-7969-5718; *e-mail:* konstantin.pchelin@math.msu.ru

Шляхов Артем Вячеславович, младший научный сотрудник отдела учебной информатики, Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт», г. Москва, Россия; *ORCID:* https://orcid.org/0009-0005-7108-6707; *e-mail:* shlyakhov@vip.niisi.ru

Information about the authors

Anatoly G. Kushnirenko, Candidate of Sciences (Physics and Mathematics), Docent, Head of the Educational Software Department, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; *ORCID:* http://orcid.org/0000-0001-6898-901X; *e-mail:* agk_@mail.ru

Alexander G. Leonov, Doctor of Sciences (Education), Candidate of Sciences (Physics and Mathematics), Docent, Head of the Department of Continuing Professional Education, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; Leading Researcher at the Laboratory of Computational Methods, The Department of Computational Mathematics, Branch of Mathematics, Faculty of Mechanics and Mathematics, Lomonosov Moscow State University, Moscow, Russia; Professor at the Department of Mathematics and Informatics in Primary School, Faculty of Elementary Education, Institute of Childhood, Moscow Pedagogical State University, Moscow, Russia; Master’s Program Supervisor of the “Digital Technologies in Public and Corporate Governance”, Associate Professor at the Department of Mathematical Methods in Economics and Management, Institute of Information Systems, State University of Management, Moscow, Russia; *ORCID:* https://orcid.org/0000-0001-9622-1526; *e-mail:* dr.l@vip.niisi.ru

Kirill A. Mashchenko, Junior Researcher at the Department of Educational Informatics, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; Junior Researcher at the Laboratory of Computational Methods, The Department of Computational Mathematics, Branch of Mathematics, Faculty of Mechanics and Mathematics, Lomonosov Moscow State University, Moscow, Russia; Master’s Program Instructor of the «Digital Technologies in Public and Corporate Governance», The Department of Mathematical Methods in Economics and Management, Institute of Information Systems, State University of Management, Moscow, Russia; *ORCID:* https://orcid.org/0000-0003-0355-6699; *e-mail:* kirill010399@vip.niisi.ru

Nikolay S. Martynov, Junior Researcher at the Department of Educational Informatics, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; *ORCID:* https://orcid.org/0009-0001-3071-2485; *e-mail:* nikolai.martynov@math.msu.ru

Konstantin K. Pchelin, a technician, Department of Educational Informatics, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; *ORCID:* https://orcid.org/0009-0005-7969-5718; *e-mail:* konstantin.pchelin@math.msu.ru

Artem V. Shlyakhov, Junior Researcher at the Department of Educational Informatics, Scientific Research Institute for Systems Analysis of the National Research Centre “Kurchatov Institute”, Moscow, Russia; *ORCID:* https://orcid.org/0009-0005-7108-6707; *e-mail:* shlyakhov@vip.niisi.ru

Поступила в редакцию / Received: 23.01.26.
Поступила после рецензирования / Revised: 16.02.26.
Принята к печати / Accepted: 17.02.26.