

DOI: 10.15514/ISPRAS-2026-38(6)-7



Исследование семантического представления табличных данных для верификации фактов

А.И. Поддубный, ORCID: 0000-0002-2431-8692 <poddubnyyiv@yandex.ru>

Н.О. Дородных, ORCID: 0000-0001-7794-4462 <nikidorny@icc.ru>

А.С. Армаев, ORCID: 0009-0007-6764-9292 <armaevaldar7@gmail.com>

Институт динамики систем и теории управления имени В.М. Матросова СО РАН,
Россия, 664033, г. Иркутск, ул. Лермонтова, д. 134.

Аннотация. Таблицы широко используются в различных областях от аналитики до управления, однако их автоматическая интерпретация затруднена структурной неоднородностью и отсутствием явной семантики. В данной статье предложен подход к задаче рассуждений над табличными данными (Table Reasoning), основанный на компактном семантическом представлении таблиц и генерации исполняемых Pandas-запросов к ним. Представление является схемой данных таблицы в формате XML, обогащенной типами данных, семантическими ролями столбцов и контекстуальным описанием, что сохраняет структурные зависимости при значительном сокращении объема данных исходной таблицы. Эксперименты на наборе данных TabFact показали, что модель с LoRA-адаптером достигает наилучшего баланса по точности (80.15%) и исполнимости кода (95.38%). При сравнении модели с базовым решением без учета специальной коррекции ошибок, модель с LoRA-адаптером достигает лучшей точности. В работе также приводится детальный анализ ошибок, не позволяющих модели продемонстрировать аналогичные базовому решению итоговые результаты с учетом исправления ошибок. Для подтверждения переносимости предложенного подхода на другие данные, были проведены дополнительные эксперименты на наборе данных WikiTableQuestions в контексте задачи ответов на вопросы по табличным данным.

Ключевые слова: табличные данные; рассуждения по таблицам; верификация фактов в таблице; вопросно-ответная система; табличное представление; семантическое представление таблиц; выполнение запроса.

Для цитирования: Поддубный А.И., Дородных Н.О., Армаев А.С. Исследование семантического представления табличных данных для верификации фактов. Труды ИСП РАН, том 38, вып. 6, часть 1, 2026 г., стр. 97–122. DOI: 10.15514/ISPRAS-2026-38(6)-7.

Благодарности: Работа выполнена в рамках государственного задания Министерства науки и высшего образования Российской Федерации "Методы и технологии обучения представлений данных (representation learning) и самообучения (self-supervised learning) для задач обработки изображений и естественного языка. Тема № 124051500067-7".

Study of Semantic Representation of Tabular Data for Fact Verification

I.A. Poddubny, ORCID: 0000-0002-2431-8692 <poddubnyyiv@yandex.ru>

N.O. Dorodnykh, ORCID: 0000-0001-7794-4462 <nikidorny@icc.ru>

A.S. Armaev, ORCID: 0009-0007-6764-9292 <armaevaldar7@gmail.com>

Matrosov Institute for System Dynamics and Control Theory of the Russian Academy of Sciences,
134, Lermontov st., Irkutsk, 664033, Russia.

Abstract. Tables are widely used in various fields, from analytics to management, but their automatic interpretation is hampered by structural heterogeneity and the lack of explicit semantics. This paper proposes an approach to table reasoning based on a compact semantic representation of tables and the generation of executable Pandas queries for them. The representation is a table data schema in XML format, enriched with data types, semantic column roles, and a contextual description, which preserves structural dependencies while significantly reducing the data size of the original table. Experiments on the TabFact dataset showed that the model with the LoRA adapter achieves the best balance between accuracy (80.15%) and code executability (95.38%). When comparing the model with the baseline solution without special error correction, the model with the LoRA adapter achieves better accuracy. The paper also provides a detailed analysis of the errors that prevent the model from demonstrating final results similar to the baseline solution with error correction. To confirm the transferability of the proposed approach to other data, additional experiments were conducted on the WikiTableQuestions dataset in the context of the task of answering questions on tabular data.

Keywords: tabular data; table reasoning; table fact verification; question-answering system; tabular representation; semantic table representation; query execution.

For citation: Poddubny I.A., Dorodnykh N.O., Armaev A.S. Study of Semantic Representation of Tabular Data for Fact Verification. Trudy ISP RAN/Proc. ISP RAS, vol. 38, issue 6, part 1, 2026, pp. 97-122 (in Russian). DOI: 10.15514/ISPRAS-2026-38(6)-7.

Acknowledgements. The research was carried out within the state assignment of Ministry of Science and Higher Education of the Russian Federation "Methods and technologies of representation learning and self-supervised learning for image processing and natural language tasks. Theme No. 124051500067-7".

1. Введение

Табличные данные остаются одним из фундаментальных форматов структурированного представления информации, широко применяемым в финансовой аналитике, научных исследованиях, корпоративном управлении и системах поддержки принятия решений. Их универсальность и компактность обуславливают повсеместное использование, однако неоднородность структуры, отсутствие явной семантики, скрытые логические зависимости между ячейками и многоуровневые иерархии заголовков существенно затрудняют автоматическую интерпретацию в различных интеллектуальных системах [1]. В ответ на эти вызовы в последние годы сформировалось целое научное направление, известное как – рассуждения на табличных данных (Table Reasoning) [2]. Данное направление охватывает широкий спектр задач обработки структурированных данных, среди которых центральное место занимают: автоматические ответы на вопросы по таблицам (Table Question Answering, TQA) [3], верификация фактов в таблицах (Table Fact Verification, TFV) [4], генерация текстовых описаний к таблицам (Table-to-Text) [5] и преобразование естественного языка в предложения языка запросов SQL (Text-to-SQL) [6]. Основная цель данной области заключается в точном извлечении, преобразовании и обоснованном выводе информации из табличных источников на основе пользовательских запросов, сформулированных на естественном языке.

Несмотря на впечатляющий прогресс, достигнутый благодаря переходу от эвристических систем к предварительно обученным языковым моделям и, в последнее время, к большим

языковым моделям (Large Language Model – LLM), современные подходы продолжают сталкиваться с рядом критических ограничений. Ключевыми проблемами остаются недостаточная точность выполнения многошаговых логических и арифметических операций, уязвимость к шуму и неоднозначности в схемах таблиц, слабое связывание терминов запроса с элементами структуры (*schema linking*), а также систематическая потеря двумерной топологии при принудительной линейаризации табличных данных. Кроме того, генеративные модели демонстрируют высокую зависимость от формулировок текстовых подсказок (*prompts*), склонность к каскадным ошибкам при пошаговом выводе и трудности с интеграцией внешнего контекста, окружающего таблицу. Исторически точность передовых TQA-систем на стандартных наборах данных (например, WikiTableQuestion (WTQ) [7], FeTaQA [8], TabFact [9]) редко превышала 60% по метрикам точности (*Denotation Accuracy* и *Flexible Denotation Accuracy*) [10-12], хотя внедрение LLM и повысило эти показатели до 70-80% [13-15], оно не устранило фундаментальный разрыв между статистическим генеративным выводом и детерминированным семантическим пониманием табличных данных.

Особую сложность в современных интеллектуальных системах, работающих с табличными данными, представляет обработка крупноразмерных таблиц, содержащих тысячи строк, что типично для реальных аналитических и корпоративных сценариев. Ограниченный объем контекстного окна языковых моделей в сочетании с обязательной линейаризацией двумерной структуры таблицы приводит к тому, что при подаче полной таблицы значительная часть данных либо принудительно отсекается, либо обрабатывается с существенной потерей пространственных и логических связей. По мере роста объема таблицы экспоненциально возрастает уровень информационного шума, что затрудняет точное связывание схемы с запросом, повышает вероятность галлюцинаций и провоцирует каскадные ошибки при многошаговых вычислениях. Существующие решения, пытающиеся смягчить эту проблему за счет предварительной фильтрации строк, применения RAG-подходов (Retrieval Augmented Generation) [16] или модульной декомпозиции задачи, часто опираются на эвристические ранжирования или внешние компоненты поиска, которые не гарантируют сохранения семантической целостности данных и могут пропускать критически важные, но неявные или распределенные по разным участкам таблицы зависимости. Таким образом, отсутствие масштабируемых и структурно-осознанных механизмов навигации по объемным табличным массивам остается одним из ключевых барьеров, сдерживающих внедрение TQA-систем в промышленную эксплуатацию.

В данной работе мы сосредотачиваемся на исследовании универсального компактного представления табличных данных, обогащенного дополнительной семантикой, как ключевого механизма преодоления указанных барьеров. Для этого предлагается подход, направленный на явное моделирование как структурных, так семантических аспектов таблиц с генерацией исполняемых рассуждений в виде Pandas-кода, что позволяет повысить точность интерпретации запросов, улучшить обоснованность числовых рассуждений и обеспечить устойчивую, верифицируемую генерацию ответов в контексте задач Table Reasoning. Предварительные эксперименты, проведенные на тестовом наборе данных TabFact [9] в рамках задачи TFV, подтверждают перспективность предлагаемого подхода. Таким образом, основной вклад работы включает:

- новый подход к семантическому представлению таблиц в компактном формате кодирования, обогащенный типами данных, семантическими ролями столбцов и контекстуальным описанием, что позволяет сохранять структурные зависимости при значительном сокращении объема данных входных таблиц, и как следствие, входных токенов для модели;
- LoRA-адаптер для модели, обученный на генерацию ответа в форме исполняемых рассуждений через преобразование запросов пользователя на естественном языке в

Pandas-запросы, обеспечивающие интерпретируемость и верифицируемость процесса вывода за счет явного кодирования логики рассуждения;

- комплексную оценку на тестовом наборе данных TabFact, демонстрирующая, что модель, обученная на предложенном представлении с использованием LoRA-адаптера, достигает точности 79.58%, что превосходит базовое решение без применения специальной корректировки ошибок по точности (78.02%), подтверждая перспективность компактного семантического кодирования для задач табличных рассуждений.

Статья организована следующим образом. Раздел 2 представляет современное состояние исследований в области табличных рассуждений и методов представления табличных данных. В разделе 3 описывается предлагаемый подход, включая формальную постановку задачи и детальное описание этапов создания компактного семантического представления таблиц. В разделе 4 приводятся результаты экспериментальной оценки и их обсуждение. В заключении дается краткий анализ полученных результатов и планы будущей работы.

2. Современное состояние исследований

2.1 Эволюция архитектур и современные методы табличных рассуждений

Исследования в области табличных рассуждений прошли несколько технологических этапов: от систем, основанных на ручных правилах, к нейросетевым архитектурам предварительного обучения и, наконец, к эре LLM и большим мультимодальным языковым моделям (Multimodal Large Language Model – MLLM) [17].

В до-LLM эпоху доминировали методы семантического разбора, трансформирующие естественный язык в исполняемые логические формы (преимущественно SQL), которые затем применялись к исходным данным. Подход TAPEX [12] продемонстрировал высокую эффективность предварительного обучения на масштабных синтетических корпусах, имитирующих выполнение SQL-запросов, а OmniTab [18] расширил эту парадигму за счет комбинирования естественных и сгенерированных данных в рамках сквозной архитектуры. Параллельно развивались методы, не требующие промежуточного логического представления: генеративные модели (например, FeTaQA [8]), синтезирующие ответ в свободной форме, и фреймворки прямого извлечения токенов (например, TaPaS [19], TAT-QA [20]), локализующие ответ непосредственно в линейаризованной таблице.

С появлением LLM парадигма табличных рассуждений сместилась в сторону использования эмерджентных способностей моделей. Современный стек методов включает: дообучение с учителем (*supervised fine-tuning*) на инструктивных наборах данных, обучение в контексте (*in-context learning*) с динамическим подбором демонстраций, ансамблирование результатов для повышения устойчивости, модульную декомпозицию задач и интеграцию внешних инструментов [21]. Особую популярность приобрели методы пошагового рассуждения, такие как «Chain-of-Table» [13], позволяющие моделям итеративно выбирать и применять операции трансформации таблицы, формируя цепочку промежуточных состояний. Подход TART [15] также направлен на пошаговые рассуждения, но использует при этом специализированные инструменты для работы с таблицами для повышения эффективности различных вычислений. В работе [22] предложен многоагентный подход TableCritic, состоящего из четырех специализированных агента для выявления и исправления различного рода ошибок на промежуточных этапах рассуждения над табличными данными до сходимости к правильным решениям. Подход включает саморазвивающееся дерево шаблонов, которое систематически накапливает знания о критических замечаниях посредством обучения на

основе опыта и направляет дальнейшие размышления модели. Подход RePanda [23] преобразует вопросы и утверждения на естественном языке в исполняемые запросы Pandas, обеспечивая интерпретируемые и проверяемые рассуждения. Это позволяет решать проблему так называемого «черного ящика», когда модель осуществляет обработку таблиц с непрозрачной логикой рассуждений. В работе [24] предлагается использовать возможности долгосрочного планирования LLM для улучшения понимания таблиц. Подход позволяет выполнять долгосрочный план, где этапы тесно взаимосвязаны и служат конечной цели, чего не хватает методам, основанным на «цепочке мыслей» и декомпозиции вопросов. Существуют примеры подходов (например, ELEET [25]), направленных на выполнение запросов одновременно к текстовым и табличным данным, обеспечивая бесшовную интеграцию текста и таблиц на основе мультимодальных операторов с использованием как больших, так и малых языковых моделей. В совокупности эти и другие решения обеспечили качественный скачок на различных стандартизированных наборах тестирования производительности (*benchmark*) [2]. В частности, точность на WTQ выросла с 37% до 75%, а на TabFact с 68% до 90% [23, 24].

2.2 Методы представления таблиц

На сегодняшний день исследователи выделяют четыре основных вида таблиц (матричные, иерархические, реляционные и таблицы с левым боковым), каждый из которых предъявляет уникальные требования к преобразованию (*parsing*) и пониманию структурных связей, что усложняет создание универсальных решений и требует адаптации методов под конкретную топологию данных [2]. Кроме того, работа [26] показала, что на точность ответа также оказывает влияние формат представления таблицы, которая обычно кодируется либо в виде текста путем процедуры линеаризации (развертывание двумерной структуры таблицы в виде одномерной последовательности символов), либо в виде изображения. Другой очень важной проблемой является размер таблиц, который может достигать тысяч строк в реальных практических задачах. Это делает такие таблицы сложными в обработке даже для современных решений, основанных на LLM. Все эти факторы привели к возникновению исследований, направленных на изучение эффективных представлений таблиц. Например, в работе [27] исследуется интеграция текстовых и визуальных представлений таблиц с использованием MLLM для решения задачи TQA. Авторы предлагают систему RITT с поддержкой поиска, которая сначала определяет наиболее релевантную часть таблицы для заданного вопроса, а затем динамически выбирает оптимальные представления таблицы на основе типа вопроса. Для улучшения понимания структуры таблиц в LLM во время прямого применения методов параметрически эффективной тонкой настройки (*parameter-efficient fine-tuning*) может быть применен модуль TableLoRA [28], который включает специальные токены для сериализации таблиц со специальным кодировщиком токенов и использует 2D LoRA (Low-Rank Adaptation) для кодирования информации о позициях ячеек. Архитектура CARTE (Context-Aware Representation of Table Entries) [29] представляет каждую строку таблицы в виде графа и использует контекстно-зависимые векторные представления для ячеек и названий столбцов (заголовков). Такой подход позволяет проводить предварительное обучение (*pre-training*) и трансферное обучение (*transfer learning*) на таблицах без необходимости предварительного приведения их к единой схеме (*schema matching*) или проводить сопоставление сущностей (*entity matching*). В работе [30] предлагается средство TableRAG, которое использует RAG-техники для поиска и извлечения из таблиц только релевантной информации перед передачей в LLM, делая длину текстовой инструкции-подсказки (*prompt*) независимой от размера таблицы. При этом TableRAG использует расширение запросов в сочетании с поиском схем и ячеек для точного определения важной информации и формирования итогового контекста для LLM. Это обеспечивает более эффективное кодирование данных и точный поиск, значительно сокращая длину запросов и уменьшая потерю информации.

2.3 Системные ограничения и перспективы семантического моделирования

Несмотря на очевидный прогресс, современные системы табличных рассуждений сталкиваются с фундаментальными ограничениями, обусловленными недостаточной формализацией внутренней структуры данных. Во-первых, принудительная линеаризация таблиц для подачи в языковые модели неизбежно приводит к потере пространственных и иерархических зависимостей. Это критично для задач с многоуровневыми заголовками, объединенными ячейками или кросс-табличными ссылками, где LLM часто затрудняется восстанавливать исходную двумерную топологию из линейной последовательности токенов. Во-вторых, хотя LLM демонстрируют превосходные способности в понимании структуры благодаря навыкам анализа иерархий и установлению связей схема-запрос (*schema linking*) за счет плотных общих знаний, генеративная природа моделей остается уязвимой к систематическим ошибкам в числовых рассуждениях и неявных вычислениях. В-третьих, методы пошагового рассуждения страдают от проблемы накопления ошибок, когда неверный промежуточный вывод необратимо искажает последующие шаги, а чувствительность к формулировкам текстовых инструкций и неточности при вызове внешних инструментов дополнительно снижают устойчивость систем в предметно-ориентированных предметных областях.

Перспективным направлением преодоления указанных барьеров выступает переход от плоской сериализации к явному семантическому представлению табличных данных. Формализация таблиц в виде обогащенных онтологических структур [31], графов знаний [32] или логических схем с явным указанием типов данных, ограничений целостности и семантических ролей строк и столбцов позволяет сохранить структурные отношения на уровне, доступном для точного автоматического анализа. Такое представление не только облегчает задачу модульной декомпозиции и автоматической верификации промежуточных состояний, но и создает надежную основу для интеграции RAG-техник и применения деревьев рассуждений (Tree-of-Thought) для минимизации эффекта накопления ошибок. Дальнейшие исследования в данной области должны быть сфокусированы на разработке гибридных архитектур, сочетающих семантическую строгость структурированных представлений с генеративной гибкостью LLM. Внедрение специализированных семантических форматов позволит создать интерпретируемые, устойчивые к шуму и масштабируемые системы, способные решать сложные задачи табличных рассуждений в реальных промышленных и научных приложениях.

3. Предлагаемый подход

3.1 Постановка задачи

В данной работе рассматривается задача рассуждений над табличной информацией (*Table Reasoning*), которая может быть сведена к задачам TQA и TFV, и в которой пользователь формулирует запросы к табличным данным или утверждения на естественном языке и ожидает получить корректный (правильный) результат в виде либо открытого текстового ответа, либо закрытого бинарного ответа («да» или «нет»).

В качестве исходных (входных) таблиц для данных задач обычно рассматриваются вертикальные таблицы, где данные систематически выровнены в виде вертикальных последовательностей (колонок/столбцов). Каждый столбец может включать описательную метку (заголовок) для уточнения его содержимого. В таких таблицах столбцы можно условно разделить на два типа:

- *категориальный столбец (столбец именованных сущностей)* – содержит названия (текстовые упоминания) некоторых именованных сущностей (персон, организаций, мест, событий и т.п.);

- *литеральный столбец* – содержит литеральные значения (даты, числа, ссылки и тому подобное).

Запросы или утверждения обычно формулируются на естественном языке, либо могут быть представлены с использованием промежуточных форм, например, в виде логических формул [7] или SQL-запросов.

Основное предположение – чтобы ответить на какой-либо вопрос по таблице человеку достаточно иметь описание структуры и особенностей формата табличных данных, после чего он может написать код, выполнение которого на всей таблице даст нужный ответ.

Следуя этому предположению, *основная идея* предлагаемого подхода – создать компактное представление таблицы, которое корректно отражает ее структуру, и наполнить его семантической информацией о таблице, которая может в ней содержаться.

Таким образом, входными данными для предлагаемого подхода является компактное семантически обогащенное представление таблицы T' , полученное на основе исходной вертикальной таблицы T . Тогда формально задача TQA/TFV формулируется как поиск функции f :

$$f(T', Q) \rightarrow A,$$

где Q – вопрос или утверждение пользователя; A – конечный итоговый ответ, удовлетворяющий условиям:

- соответствия интенции вопроса (семантической согласованности);
- логической согласованности с табличными данными;
- ответ может быть получен через операции, допустимые для структуры и типов данных T' (суммирование, фильтрация по условию, нахождение максимума и т.п.).

Как и в работах [12, 23], данный подход предполагает, что вопросы Q преобразуются в исполняемые цепочки запросов, составленных на языке SQL или Pandas, для получения ответа A . Такой подход явно кодирует процесс рассуждения, а все преобразования над таблицей могут проводиться программным (вычислительным) модулем, обеспечивая интерпретируемую проверку фактов.

В основе T' используется схема данных таблицы, к которой добавляется ее описание. К каждому имени колонки добавляется информация о типе данных, примеры данных, семантическая роль в контексте таблицы (описание), наличие или отсутствие пропусков в данных. Формально, это можно представить следующим образом:

$$T' = \{DS, d, f\}, DS = \{c_1, \dots, c_n\},$$

где f – машиночитаемый формат представления таблицы; d – общее описание всей таблицы (по умолчанию ее название); DS – схема табличных данных, представляющая собой множество описаний столбов, при этом:

$$c_i = \{h, DT, ST, E, p\}, i = \overline{1, n},$$

где c_i – i -описание столбца из исходной таблицы T ; h – название столбца (заголовок); DT – простой тип данных столбца, определенного в Pandas [33], который использует систему типов часто называемых «*dtypes*» для определения способа хранения информации и выполняемых над ней операций; ST – семантический тип, точно характеризующий данные столбца; E – набор из нескольких случайно отобранных примеров значений ячеек для данного столбца; p – информация о наличии или отсутствии пропусков в значениях ячеек.

К основным достоинствам такого представления таблиц можно отнести:

- компактность и легковесность с концентрированной семантической информацией, при этом эффективность такого формата растет с увеличением количества строк в таблице (см. Раздел 4 с экспериментами);
- использование схемы данных и генерация исполняемого кода позволяют

обрабатывать таблицы с очень большим объемом строк, который превышает контекст модели.

Однако такой подход накладывает ряд ограничений на его использование из-за того, что модель не видит всю таблицу целиком:

- модель должна уметь генерировать код или использовать функции для получения конечного ответа;
- требуется дополнительный контроль на совпадение написания сущностей в вопросе и в таблице, а также над типом данных.

Далее подробнее разберем детали реализации предлагаемого табличного представления.

3.2 Этапы создания компактного представления таблицы

Основные этапы автоматизированного создания универсального, компактного и семантически обогащенного представления таблицы показаны на рис. 1. Далее подробнее рассмотрим каждый из этапов.

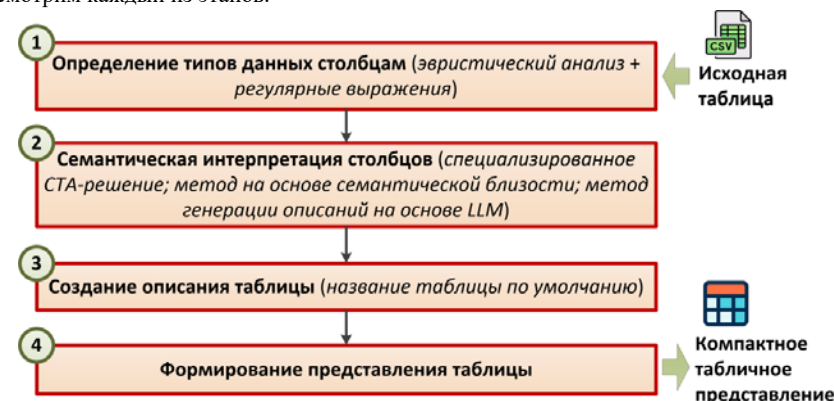


Рис. 1. Основные этапы создания компактного семантического представления таблицы.
Fig. 1. Main stages of creating a compact semantic representation of a table.

Этап 1: Определение типов данных столбцам. Предполагается, что исходные гетерогенные табличные данные хранятся в текстовом машиночитаемом формате CSV (Comma-Separated Values). Вначале выполняется первичный разбор исходной таблицы с предварительной обработкой строковых значений ячеек (например, удаляются специфические символы форматирования, такие как валютные знаки, «мусорные» символы, единицы измерения и так далее). Такой разбор основан на системе эвристического анализа и классификации простых типов данных столбцов на базе библиотеки Pandas [33] и иерархической проверки на принадлежность значения ячеек к категориям чисел, дат, временных меток или булевых величин. Этого используются предварительно скомпилированные регулярные выражения для распознавания сложных финансовых и региональных форматов данных, которые часто игнорируются стандартными средствами типизации. Для повышения производительности при обработке комплексных сложно-структурированных табличных наборов задействован механизм параллельных вычислений, позволяющий изолированно анализировать каждый признак в отдельном потоке. Итоговое решение о типе данных принимается на основе мажоритарного голосования с учетом специфики обработки пропущенных значений и автоматического повышения приоритета вещественных чисел над целыми при наличии смешанных данных в столбце.

Ключевым преимуществом данной реализации является высокая устойчивость к «шумным» входным данным и возможность корректной интерпретации неявных типов, таких как финансовая запись отрицательных чисел в скобках или локализованные буквенные даты. Архитектура решения обеспечивает прозрачную логику работы и легкую расширяемость за счет добавления новых паттернов без изменения ядра алгоритма. Тем не менее, жесткая опора на статические регулярные выражения ограничивает адаптивность системы при столкновении с принципиально новыми форматами.

Этап 2: Семантическая интерпретация столбцов. Данная задача является одной из основных в области семантической интерпретации (аннотирования) таблиц (*semantic table interpretation*), под которой в общем случае понимается процесс распознавания и связывания табличных данных с понятиями из некоторого целевого графа знаний, онтологии или другого внешнего словаря (например, DBpedia, Wikidata, Yago, Freebase, WordNet) [34]. Семантическая интерпретация столбцов (Column Table Annotation, CTA) заключается в сопоставлении столбцов таблицы с семантическими типами (классами и свойствами) из целевого графа знаний (*knowledge graph*) [32]. Это позволяет сопроводить таблицу явной семантикой необходимой для автоматической машинной интерпретации своего содержания. В данной работе разработано и применяется три метода семантической интерпретации столбцов:

- 1) Метод определения релевантных семантических типов с использованием специализированных CTA-решений (например, SOTA-решения – фреймворка Doduo [35]), основанных на тонко настроенных предварительно обученных языковых моделях (обычно применяются модели семейства BERT [36]). В Doduo классификация проводится по 78 типам из набора VizNet-Sato [37]. Помимо самого предсказанного типа модель также возвращает уверенность этого предсказания. Данный метод является относительно простым, однако при этом набор возможных типов имеют высокую абстрактность, что может негативно сказаться на результатах аннотирования.
- 2) Метод идентификации семантических типов через использование косинусного сходства между векторными представлениями названий столбцов (заголовков) и базовой таксономией классов и свойств, извлеченной из графа знаний общего назначения, например, DBpedia [38]. Преимущество такого решения в том, что можно подключать практически любые предметно-ориентированные графы знаний и использовать специфические семантические типы под каждую задачу. Однако недостатком такого метода является жесткая зависимость от качества и полноты названий заголовков столбцов, а также игнорирование табличного содержимого и контекста.
- 3) Метод генерации контекстуализированных семантических типов (описаний) при помощи LLM. Одним запросом модель формирует типы для каждого столбца таблицы. При этом начальный список типов также может быть сформирован на основе таксономии классов и свойств графа знаний. Основное преимущество данного метода в том, что он формирует типы с учетом, как самого описания таблицы, так и контекста других соседних столбцов. Фактически, метод ограничен областью знаний самой LLM, используемой для генерации.

Этап 3: Создание описания таблицы. На этом этапе по умолчанию используется название таблицы, однако такое описание может быть сгенерировано при помощи LLM путем суммаризации внутреннего содержания таблицы, а также дополнительным текстовым контекстом, в который погружена таблица (например, это может быть текст до и после таблицы, если сама таблица извлекается из PDF-документа или HTML-страницы).

Этап 4: Формирование представления таблицы. На последнем этапе осуществляется сборка компактного представления таблицы на основе определенных типов. При этом случайно отбираются несколько примеров значений ячеек для каждого столбца, а также фиксируется информация о наличии или отсутствии пропусков. Для записи этого табличного представления разработано три отдельных формата сериализации:

- 1) *HTML формат* – часто встречающийся формат представления различного рода таблиц в Интернете. Структура таблицы задается специальными тегами такими как: `<table>`, `<thead>`, `<tr>`, `<th>`. Описание таблицы описывается в теге `<caption>`, а названия столбцов и другая семантическая информация задаются как атрибуты тега `<th>`.
- 2) *XML-element формат* – формат XML также является распространенным способом представления структурированной информации в сети Интернет. В отличие от фиксированных тегов в HTML, данный формат дает больше гибкости в построении структуры таблицы за счет свободы выбора названия тегов. За счет этого можно с помощью названия тега сообщать модели, что находится внутри тега. В этом варианте формата для обозначения структуры таблицы используются теги `<TABLE>` и `<HEADER>`, а названия и семантическая информация указываются как дочерние элементы `<HEADER>` с отдельными тегами.
- 3) *XML-attribute формат* – отличие данного варианта формата от XML-element в том, что названия колонок и семантическое наполнение задаются как атрибуты тега `<HEADER>`, а описание таблицы как атрибут `<TABLE>`.

Основным достоинством использования всех трех форматов записи таблицы является то, что модели имеют базовые представления о его структуре. Существует ряд исследований, в частности [39, 40], в которых показана эффективность форматов XML/HTML для представления таблиц и передачи их в контекст модели.

4. Эксперименты

4.1 Детали реализации подхода

Для этапа семантического аннотирования столбцов, в качестве специализированного CTA-решения выбрана система (*framework*) – Doduo [35], которая является одним из лидеров в этой области.

Для метода формирования семантического описания через использование косинусного сходства, в качестве целевого графа знаний использовалась DBpedia [38], из которой была извлечена онтологическая схема, состоящая из 812 классов (*classes*), 1170 свойств-значений (*datatype properties*) и 1861 объектных свойств (*object properties*). Также для каждого из этих элементов была собрана текстовая метка (название) и комментарий (описание элемента), а для каждого класса дополнительно собиралась информация о его родительском классе. Для генерации векторных представлений применялась LLM – Qwen3-Embedding-4B [41], а библиотека и векторная база данных FAISS (Facebook AI Similarity Search) [42] использовалась для хранения построенных векторов и организации эффективного поиска по косинусному сходству в многомерном пространстве. Результирующая векторная база состоит из 3837 описаний онтологической схемы, например: «*This is actor. It is class of Artist. An actor or actress is a person who acts in a dramatic production and who works in film, television, theatre, or radio in that capacity*».

Для формирования контекстуализированного семантического описания столбцов использовалась модель Qwen3-VL-30B-A3B-Thinking [43] и специальная текстовая инструкция, фрагмент которой приведен на рис. 2.



Рис. 2. Фрагмент текстовой инструкции для генерации контекстуализированного семантического описания столбца.

Fig. 2. Prompt fragment for generating a contextualized semantic description of a column.

4.2 Наборы данных для тестирования

Для экспериментальной оценки эффективности предлагаемого подхода компактного семантического представления табличных данных был использован тестовый набор данных TabFact [9], разработанный специально для задачи верификации фактов на основе полуструктурированных табличных доказательств (TFV). В отличие от традиционных наборов данных, ориентированных на проверку утверждений по неструктурированным текстам или документации, TabFact фокусируется на сложном смешанном рассуждении, требующем одновременного понимания естественного языка и точной символической интерпретации двумерной структуры таблиц.

Набор данных содержит 118 275 различных утверждений, составленных на естественном языке и 16 573 аннотированных вручную таблиц, извлеченных из англоязычного раздела Wikipedia. Каждое утверждение классифицируется по бинарной шкале: ENTAILED (логически следует из данных таблицы) или REFUTED (противоречит данным таблицы). Данные разделены на обучающую, валидационную и тестовую выборки в соотношении приблизительно 8:1:1, обеспечивая сбалансированное распределение классов в контрольных подмножествах.

Для тонкой настройки (*fine-tuning*) модели верификации фактов на основе табличных данных был использован специализированный корпус PanTabFact [23], представляющий собой структурированную версию эталонного набора данных TabFact, обогащенную явными

логическими формами рассуждений в виде исполняемых запросов на языке библиотеки Pandas [33]. Исходный набор TabFact не предоставляет промежуточных шагов вывода, необходимых для обучения моделей с явным семантическим представлением. В связи с этим авторы подхода RePanda [23] предложили методику автоматического преобразования естественных утверждений в программные выражения, обеспечивающие их детерминированную верификацию.

Каждый экземпляр PanTabFact представляет собой кортеж:

$$PanTabFact = \{T, s, q, l\},$$

где T – исходная таблица в формате pandas.DataFrame; s – утверждение на естественном языке; q – исполняемый запрос, кодирующий логику верификации утверждения; l – бинарная метка истинности, при этом: $l \in \{ENTAILED, REFUTED\}$.

Наличие явного программного представления q обеспечивает не только возможность детерминированной проверки ответа, но и интерпретируемость процесса рассуждения. Так пользователь может проанализировать сгенерированный код, выявить ошибки в связывании схемы или логики операций и при необходимости скорректировать текстовые инструкции или архитектуру модели.

4.3 Метрика оценки

В качестве основной метрики оценки для решения задачи TFV используется *точность* (Accuracy) [9]. Данная метрика вычисляет долю верно верифицированных утверждений от общего числа проверенных экземпляров и определяется по следующей формуле:

$$Accuracy = \frac{1}{N} \sum_{i=1}^N f(P_i = GT_i),$$

где N – общее количество пар «таблица-утверждение» в тестовой выборке; $GT_i \in \{0,1\}$ – истинная метка (*ground truth*) для i -утверждения (1 соответствует классу ENTAILED, 0 – классу REFUTED); $P_i \in \{0,1\}$ – метка, предсказанная моделью для того же утверждения; f – индикаторная функция, которая возвращает 1, если предсказание совпадает с истинным значением и 0 в противном случае.

Для оценки способности модели генерировать синтаксически корректный и исполняемый код, независимо от семантической правильности результата, в литературе по генерации программ и семантическому разбору обычно используется метрика (Code Execution Rate – CER) или (Executable Success Rate – ESR) [46].

В данной работе метрика оценки ESR фиксирует долю сгенерированных pandas-запросов, которые успешно выполняются интерпретатором без возникновения исключений (*SyntaxError*, *NameError*, *ValueError*, *KeyError* и др.). Формализуем данную метрику оценки следующим образом:

$$ESR = \frac{1}{N} \sum_{i=1}^N \mu(Exec(q_i, T_i) \neq \varepsilon),$$

где N – общее количество пар «таблица-утверждение» в тестовой выборке; q_i – pandas-запрос, сгенерированный моделью для i -утверждения; T_i – соответствующая таблица в паре с i -утверждением, представленная в виде pandas.DataFrame; $Exec(q_i, T_i)$ – функция, осуществляющая выполнение кода q_i в изолированном окружении Python с использованием таблицы T_i ; ε – перехваченное исключение времени выполнения или синтаксической ошибки; μ – индикаторная функция, возвращающая 1, если выполнение завершилось штатно (без выброса исключений), и 0 в противном случае.

4.4 Экспериментальные настройки

Все эксперименты с компактным представлением таблиц проводились в контексте решения задачи TFV [4] на кластере, в конфигурацию которого входят два 16-ти ядерных процессора Intel Xeon Gold 6326 «Ice Lake» 2.9 GHz, четыре графических процессора NVIDIA A100 80 GB PCIe и 2 TB оперативной памяти DDR4-3200.

В качестве базовой LLM использовалась модель – Deepseek-Coder-7B-Instruct-v1.5 [44], включая как обычное применение с текстовыми инструкциями, так и тонкую настройку с адаптером LoRA. При этом в процессе тонкой настройки веса модели замораживались, а обучался только LoRA-адаптер. В качестве функции потерь (*loss function*) использовалась стандартная функция перекрестной энтропии (*cross-entropy*). Автоматизированный подбор гиперпараметров дообучения проводился с помощью библиотеки Optuna [45]. Наиболее результативным стал набор параметров:

- скорость обучения: 6.59e-5;
- размер пакета (батча): 16;
- затухание весов (*weight_decay*): 0.002;
- пропорциональный коэффициент разогрева (*warmup_ratio*): 0.0328;
- тип планировщика скорости обучения: «*cosine*»;
- ранг LoRA (*lora_rank*): 32;
- параметр регуляризации (*LoRA Dropout*) от 0.061.

С данными параметрами тонкая настройка модели на наборе данных PanTabFact была выполнена в течение 3 эпох на одном графическом процессоре NVIDIA A100 80GB в течение 17 часов. При этом набор был разделен случайным образом на обучающую и валидационную выборки. Разделение выполнялось через список уникальных таблиц в соотношении 20% валидационный (17 720 пар примеров) и 80% обучающий (70 254 пар примеров), а затем по таблицам разделялся весь набор так, чтобы таблицы и вопросы в выборках не пересекались. После завершения тонкой настройки была выбрана лучшая контрольная точка (*checkpoint*) по минимальному значению функции потерь (*loss*) на этапе валидации.

Для проверки качества самого LoRA-адаптера дополнительно была проведена тонкая настройка непосредственно всей базовой LLM. Параметры обучения взяты из работы [23]. Обучение проводилось на том же наборе данных, что и для LoRA-адаптера в течение 3 эпох на протяжении 20 часов. Лучшая контрольная точка также выбрана по минимальному значению функции потерь на этапе валидации.

Текстовая инструкция, как для обучения LoRA-адаптера, так и для обучения непосредственно всей модели сформирована на основе инструкции из работы [23] и представлена на рис. 3.

4.5 Основные результаты и их обсуждение

Для понимания потенциала использования предлагаемого подхода в целом и выбора оптимального формата записи компактного семантического представления таблиц для дальнейшего тестирования была проведена серия экспериментов, в рамках которых оценивались ответы базовой модели на наборе данных TabFact. Помимо трех предложенных вариантов форматов записи компактного семантического представления

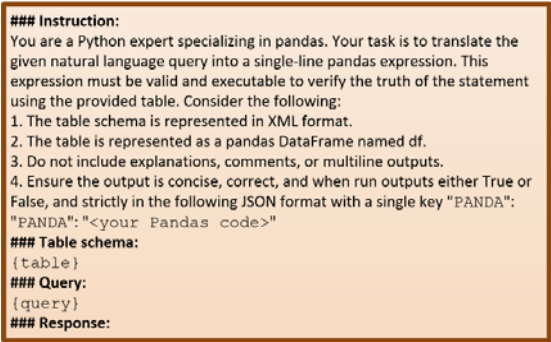


Рис. 3. Текстовая инструкция для тонкой настройки LoRA-адаптера и базовой модели на основе [23].
Fig. 3. Prompt for fine-tuning the LoRA adapter and backbone model based on [23].

(XML attributes, XML elements, HTML), также оценивались форматы, которые полностью сериализуют всю таблицу:

- *NLsep* – распространенный формат сериализации таблиц, предложенный в работе [12], который линеаризует таблицу в одномерную последовательность с помощью специальных разделителей «COL» и «ROW»;
- *Space* – формат похожий на NLsep, однако в данном случае разделителями являлись пробелы и переносы строк таблицы.

Результаты полученной экспериментальной оценки приведены в табл. 1 (результаты с учетом валидации кода).

Табл. 1. Результаты экспериментальной оценки формата записи компактного семантического представления таблиц на наборе данных TabFact.
Table 1. Results of the experimental evaluation of the compact semantic table representation format on the TabFact dataset.

Формат записи представления таблиц	Accuracy
XML attributes	54.85
XML elements	52.60
HTML	53.86
NLsep	54.23
Space	53.28

На рис. 4 представлена зависимость числа токенов от количества строк и столбцов таблицы для различных форматов сериализации. Анализ графиков показывает, что предлагаемые компактные форматы (XML attributes, XML elements, HTML) демонстрируют существенно более пологий рост длины последовательности по мере увеличения размера таблицы по сравнению с форматами полной линеаризации (NLsep, Space). В частности, при фиксированном числе столбцов увеличение количества строк в 10 раз приводит к росту токенов менее чем в 2 раза для компактных представлений, тогда как для NLsep рост практически линейный. Это подтверждает ключевое преимущество предложенного подхода – масштабируемость к крупноразмерным таблицам, содержащим тысячи строк, без превышения ограничений контекстного окна языковой модели.

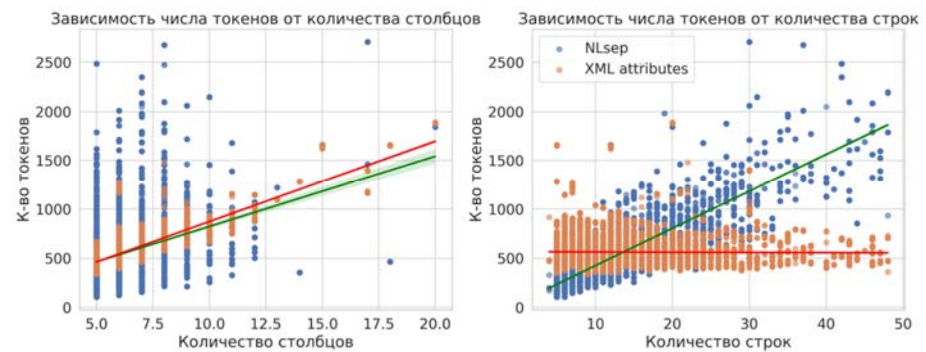


Рис. 4 Зависимости числа токенов от количества столбцов и строк для различных форматов сериализации. На графике формат полной линейаризации (NLsep) сравнивается с предлагаемым компактным семантическим представлением (XML attributes). График демонстрирует, что компактные форматы обладают существенно более пологим ростом длины последовательности, обеспечивая масштабируемость к крупноразмерным таблицам.

Fig. 4 Dependence of the number of tokens on the number of columns and rows for various serialization formats. The diagram compares the full linearization format (NLsep) with the proposed compact semantic representation (XML attributes). It demonstrates that compact formats exhibit a significantly slower rate of increase in sequence length, ensuring scalability to large tables.

Поскольку формат записи XML attributes получил наилучшие оценки, он был выбран в качестве основного для процедуры тонкой настройки базовой модели и проведения дальнейших экспериментов. Формат NLsep используется для сравнения как альтернативный формат передачи в модель полной таблицы.

Далее отдельно тестировалась возможность базовой модели генерировать однострочный Pandas-код для выполнения утверждений. Оценки были также получены на наборе TabFact с использованием текстовой инструкции, предложенной в работе [23]. При этом если код не проходил валидацию, то он возвращался модели на коррекцию (специальная инструкция для корректировки ошибок также представлен в [23]). Всего использовалось до пяти циклов валидации, которые позволили повысить точность в среднем на 12.4%, а выполняемость кода на 21.8%.

Полученные результаты оценки представлены в табл. 2.

Табл. 2. Результаты экспериментальной оценки выполнения Pandas-кода утверждений на наборе данных TabFact. (*) «C» (Correction) – это процент ответов подвергшихся корректировке относительно числа примеров в наборе; (**) «TC» (True Correction) – это процент скорректированных ответов относительно Correction.

Table 2. Results of the experimental evaluation of Pandas assertion code execution on the TabFact dataset. (*) “C” (Correction) is the percentage of answers that underwent correction relative to the total number of examples in the dataset; (**) “TC” (True Correction) is the percentage of corrected answers relative to “C” (Correction).

Формат записи представления таблиц	Accuracy	ESR	C*	TC**
XML attributes	40.64	69.32	–	–
XML attributes (с корректировкой)	<u>52.60</u>	91.25	20.91	55.35
NLsep	41.34	66.69	–	–
NLsep (с корректировкой)	54.23	<u>88.39</u>	21.50	58.79

После процедуры тонкой настройки модель оценивалась на тестовой выборке TabFact. В качестве базового решения (baseline) для сравнения разработанного подхода был выбран фреймворк RePanda [23], так как он также придерживается концепции генерации исполняемого Pandas-кода для ответов на запрос (утверждение). Выбор RePanda в качестве основного ориентира обусловлен методологической общностью. Как и предлагаемое в нашей работе решение, RePanda преобразует естественный язык в исполняемые запросы Pandas, обеспечивая интерпретируемые и проверяемые рассуждения. Это позволяет проводить прямое и корректное сравнение эффективности компактного семантического представления таблиц против полноразмерной линейаризации в рамках единой парадигмы. Хотя в литературе существуют подходы, демонстрирующие более высокую абсолютную точность за счет использования многоагентных систем или LLM (например, TableCritic [22] или Chain-of-Table [13]), RePanda является репрезентативным SOTA-решением именно для класса методов, основанных на генерации программного кода, что делает сравнение с ним наиболее обоснованным для доказательства преимуществ предлагаемого формата представления таблиц. Таким образом, все эксперименты табличного представления проводились с применением третьего метода семантического наполнения (генерации контекстуализированных семантических описаний при помощи LLM), как на инструкции из оригинальной работы RePanda (repanda), так и на инструкции (рис. 3) из процедуры дообучения (train). Тестирование также включало в себя фазу коррекции невыполнимого кода. Полученные результаты представлены в табл. 3.

Результаты, в целом, демонстрируют эффективность предложенного подхода после процедуры тонкой настройки. Конфигурация с LoRA-адаптером и форматом XML attributes достигает наилучшего баланса между точностью (80.15%) и исполнимостью кода (95.38%). Так LoRA-адаптер показал себя на 3% лучше, чем тонкая настройка всей модели, при этом улучшилась исполнимость кода на 4% по сравнению с необученной моделью. При этом нужно отметить, что качество первичной генерации по сравнению с необученной моделью возросло на 20.15%. Модель хорошо выучила входную инструкцию, это подтверждают результаты с разными текстовыми инструкциями-подсказками. LoRA-адаптер лучше следует формату вывода, как видно по метрике ESR, имея стабильно высокие значения вне зависимости от формата промта и входных данных.

Важно отметить, что модель RePanda на данных TabFact без коррекции ошибок дает точность в 78.02%, при этом предлагаемый подход позволяет модели в конфигурации с LoRA показать точность выше (79.58%). Этот факт позволяет сделать основной вывод – модель, обученная на предлагаемом компактном семантическом представлении таблиц, лучше справляется с решением задачи, чем модель RePanda, которая использует всю таблицу целиком. Однако в конечном итоге точность получилась ниже, поскольку система коррекции ошибок в подходе RePanda оказалась продуктивнее. В связи с этим, был проведен анализ полученных ошибочных ответов.

Для наглядной демонстрации интерпретируемости предлагаемого подхода далее рассматривается пример верификации утверждения на основе исходных табличных данных из набора TabFact с генерацией исполняемого Pandas-кода (рис. 5). Приведенный пример наглядно демонстрирует ключевое преимущество разработанного подхода, а именно: сгенерированный Pandas-код явно отражает логику верификации утверждения, показывая, какие именно столбцы («tournament», «top-5») и условия фильтрации были использованы моделью для принятия решения. В отличие от методов, выдающих лишь бинарный ответ с вероятностью его истинности, данная парадигма позволяет пользователю или внешнему валидатору легко отследить ход рассуждений, выявить потенциальные ошибки в связывании схемы (schema linking) и полностью устраняет проблему так называемого «черного ящика».

Табл. 3. Основные результаты экспериментальной оценки подходов на наборе данных TabFact после тонкой настройки. (*) «C» (Correction) – это доля ответов, подвергшихся коррективке относительно числа примеров в наборе; (**) «TC» (True Correction) – это процент скорректированных ответов относительно Correction.
Table. 3. Main results of the experimental evaluation of approaches on the TabFact dataset after fine-tuning. (*) “C” (Correction) represents the proportion of answers subjected to correction relative to the number of examples in the dataset; (**) “TC” (True Correction) represents the percentage of corrected answers relative to “Correction”.

Подход	Формат записи таблицы	Тип инструкции	Accuracy	ESR	C*	TC**
RePanda	NLsep	repanda	84.09	–	–	–
Deepseek-Coder	XML attributes	train	77.35	96.31	3.69	54.66
		repanda	50.55	94.87	2.18	46.24
LoRA-based	NLsep	train	49.6	72.52	11.05	52.90
	XML attributes	train	80.15	95.38	0.76	70.10
		repanda	72.2	95.06	1.07	56.93
	NLsep	train	72.27	90.67	2.31	65.76
		repanda	61.55	91.98	2.60	60.24

Исходная таблица из набора данных TabFact:

tournament	wins	top-5	top-10	top-25	events	cuts made
0 masters tournament	0	1	2	4	4	4
1 us open	0	2	3	4	6	5
2 the open championship	1	2	2	2	3	3
3 pga championship	0	0	1	2	5	4
4 totals	1	5	8	12	18	16

Проверяемое утверждение:

Tony Lema be in the top 5 for the master tournament, the us open, and the open championship.

Тони Лема войдет в пятерку лучших на турнире Мастерс, Открытом чемпионате США и Открытом чемпионате.

Полученное компактное табличное представление (XML-attributes):

```
<TABLE DESCRIPTION="tony lema">
  <HEADER NAME="tournament" COLUMN DESCRIPTION="The proper name of a golf tournament, presented as text."
  DATA_TYPE="str" HAS_NONE="0" EXAMPLES=["totals", "masters tournament", "the open championship"]/>
  <HEADER NAME="wins" COLUMN DESCRIPTION="Number of tournament wins achieved by Tony Lema (integer)."
  DATA_TYPE="int" HAS_NONE="0" EXAMPLES=["1", "1", "0"]/>
  <HEADER NAME="top - 5" COLUMN DESCRIPTION="Number of times Tony Lema finished in the top 5 positions in
  the tournament (integer)." DATA_TYPE="int" HAS_NONE="0" EXAMPLES=["2", "0", "2"]/>
  ... [остальные столбцы для краткости опущены] ...
</TABLE>
```

Сгенерированный Pandas-код:

```
all(df[df['tournament'].isin(['masters tournament', 'us open', 'the open championship'])['top - 5'] > 0)
```

Рис. 5. Наглядный пример генерации Pandas-кода на основе исходных табличных данных.
Fig. 5. An illustrative example of generating Pandas code based on source tabular data.

В дополнение было проведено сравнение предлагаемого подхода (в конфигурации базовой модели с тонко настроенным LoRA-адаптером и инструкцией «train» на основе формата записи компактного представления таблиц в «XML attributes») с другими мировыми аналогами из области TFV (табл. 4). Эксперименты показали, что модель обладает сопоставимыми результатами с передовыми решениями, что открывает новые возможности для создания интеллектуальных систем анализа таблиц, сочетающих высокую точность и интерпретируемость результатов с компактным табличным представлением.

Табл. 4. Сравнение различных моделей по точности на наборе данных TabFact.
Table 4. Comparison of various models by accuracy on the TabFact dataset.

Подход	Accuracy
TableCritic (Qwen2.5-72B) [22]	92.6
Chain-of-Table (PaLM-2) [13]	86.61
TAPEX (BART-large) [12]	84.20
RePanda [23]	84.09
TAPAS-base [19, 47]	78.50
LERGV [48]	75.50
SASP [49]	74.90
Предлагаемый подход	80.15

Для классификации ошибок из тестовой выборки набора TabFact отбирались вопросы, где модель сгенерировала код, но итоговый ответ оказался неверным. Для формирования описания ошибки, приведшей к неправильному результату, извлеченный из ответов модели код вместе с вопросом передается в LLM (Qwen3-VL-30B-A3B-Thinking [43]). Из полученных описаний формировался граф, где вершинами являются описания, а в качестве ребер используется значение косинусного сходства, вычисленное по векторным представлениям описаний. Векторные представления были также получены с помощью модели Qwen3-Embedding-4B [41]. Далее по этому графу проводилась кластеризация по методу Лейдена. Результаты кластеризации при различных параметрах Gamma представлены на рис. 6. При Gamma = 1.0 достигается баланс между обобщением и детализацией. Дальнейшее увеличение параметра превращает осмысленную группировку в хаотичное разбиение на шум. Таким образом, в результате было выделено 4 основных кластера. После чего, с помощью той же самой LLM Qwen3, осуществлялась суммаризация определений ошибок внутри кластеров и составлялись обобщенные описания. В результате анализа обобщенных определений выявлены три больших группы ошибок:

- 1) Символьные – связаны с ошибками (например, сокращениями в названиях) в упоминаемых в запросе пользователе сущностей, содержащихся в таблице.
- 2) Логические – связаны, либо с инверсией запроса (например, вместо требуемого утверждения «Футболист не имеет максимальное число голов», модель строит корректный код, но для некорректного утверждения «Футболист имеет максимальное число голов»), либо с полным непониманием модели логики запроса.
- 3) Числовые – связаны с несогласованностью способа использования данных с их типом (неверный формат даты, сравнение числа со строкой, математические операции над строками).

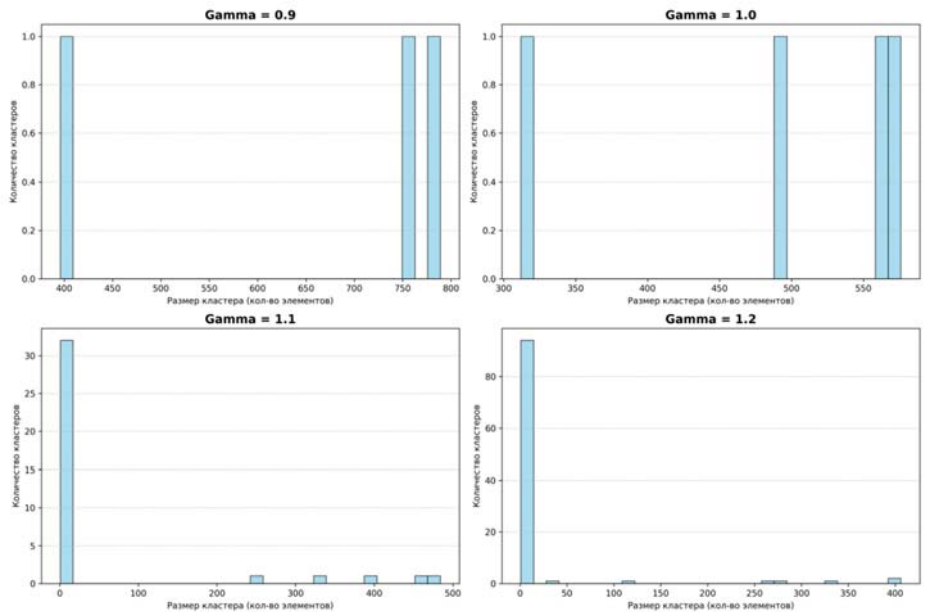


Рис. 6. Результаты кластеризации описаний ошибок по методу Лейдена при различных параметрах разрешения (Gamma). Параметр Gamma = 1.0 обеспечивает оптимальный баланс между обобщением и детализацией, позволяющий устойчиво выделить четыре основных кластера типовых ошибок. Дальнейшее увеличение параметра приводит к потере семантической связности и хаотичному разбиению кластеров на шум.

Fig. 6. Results of clustering error descriptions using the Leiden method at various resolution (Gamma) parameters. A Gamma value of 1.0 provides an optimal balance between generalization and granularity, enabling the robust identification of four main clusters of typical errors. Further increasing the parameter leads to a loss of semantic coherence and the chaotic fragmentation of clusters into noise.

4.6 Оценка на наборе данных WTQ

Для подтверждения универсальности и переносимости предложенного подхода, было проведено дополнительное тестирование на наборе данных WikiTableQuestions (WTQ) [7], ориентированном на задачу ответов на вопросы по табличным данным (TQA). В отличие от бинарной верификации фактов, задача TQA требует генерации открытого ответа, что предъявляет более высокие требования к логическому выводу и точному связыванию сущностей. В свою очередь, набор WTQ состоит из 22 033 сложных вопросов, составленных на естественном языке и сопоставленных с 2 108 полу-структурированными таблицами, извлеченными из Википедии. Ключевой характеристикой WTQ является его открытая предметная схема, где таблицы для обучения и тестирования не пересекаются, что требует от моделей способности к обобщению на невиданных ранее заголовках столбцов и значениях сущностей во время вывода (рассуждения). Однако в данном наборе отсутствует разметка в виде какого-либо исполняемого кода (SQL или Pandas), который соответствовал вопросу на естественном языке и приводил к правильному (корректному) ответу. Для создания подобного рода разметки использовалась модель Qwen3-Coder-Next [50] для генерации кода на Pandas. Модель получала на вход сериализованную таблицу, названия столбцов и исходный вопрос. Цель модели – сгенерировать код, который возвращает правильный ответ.

Процесс генерации кода для WTQ проводился в два этапа. На каждом этапе применялась трехступенчатая схема генерации и исправления кода:

- 1) Прямая генерация – модель генерирует однострочный Pandas-код на основе сериализованной таблицы и вопроса.
- 2) Коррекция синтаксиса – если код вызывает ошибку выполнения, он вместе с текстом ошибки возвращается модели для исправления (до 4 попыток с повышением температуры).
- 3) Логическая коррекция – если код синтаксически верен, но дает неверный результат, модель выполняет логическую коррекцию без изменения синтаксиса.

Результаты трехэтапной схемы генерации и исправления кода на WTQ приведены в табл. 5. Важно отметить, что из общего числа всех вопросов была использована выборка в 14 148 вопросов. Размеченный таким образом набор данных использовался при получении экспериментальной оценки.

Табл. 5. Результаты трехэтапной схемы генерации и исправления кода на WTQ с использованием модели Qwen3-Coder-Next.

Table 5. Results of the three-stage code generation and correction scheme on WTQ using the Qwen3-Coder-Next model.

Показатель	Значение
Всего вопросов в собранном наборе данных	14 148
Итоговое количество правильных ответов	10 256
Итоговый процент правильных ответов	72.49%

Методика оценивания на WTQ была адаптирована из работы TAPEX [12]. Так ответ модели и истинное значение представляются как типизированные частотные словари элементов ответа. Тип элемента определяется функцией, аналогичной той, что используется для вычисления типов столбцов таблицы. Использование частотного словаря позволяет учитывать наличие элемента и его количество, игнорируя при этом порядок следования в списке.

Предварительно было проведено тестирование на собранном наборе WTQ базовой модели Deepseek-Coder-7B-Instruct-v1.5 [44], которой передавалась вся таблица целиком, а также ранее тонко настроенной этой же модели на задаче TQV. Однако вместо полной таблицы передавалась ее компактное семантическое представление. Полученные результаты приведены в табл. 6.

Табл. 6. Предварительная оценка моделей на наборе данных WTQ. При этом показана точность без коррекции ошибок (Accuracy) и с коррекцией (Accuracy-C).

Table 6. Preliminary evaluation of models on the WTQ dataset. Accuracy is shown both without error correction (Accuracy) and with correction (Accuracy-C).

Модель	ESR	Accuracy	Accuracy-C
Deepseek-Coder	78.47	31.46	35.47
Тонко настроенный Deepseek-Coder на задаче TQV	74.95	38.19	38.53

Последующая тонкая настройка модели непосредственно для задачи TQA с использованием специально разработанных вспомогательных функций типизации и поиска сущностей позволила существенно повысить качество (табл. 7).

Обучение модели для задачи TQA осуществлялось таким же способом, как и для задачи TFV, описанной выше (тонкая настройка с LoRA). Параметры обучения взяты те же самые, что и для TFV. Размеченная часть набора WTQ разделена на тренировочную и валидационную в соотношении 80/20.

Дополнительно для повышения точности были разработаны вспомогательные функции: (1) *титизатор* числовых значений и дат в запросе, а также (2) *корректор* написаний сущностей, определяющий наиболее релевантное упоминание сущности в целевом столбце по расстоянию Левенштейна.

Табл. 7. Результаты полученной экспериментальной оценки тонко настроенной модели для задачи TQA на наборе данных WTQ. При этом показана точность без коррекции ошибок (Accuracy) и с коррекцией (Accuracy-C).

Table 7. Results of the experimental evaluation of the fine-tuned model for the TQA task on the WTQ dataset. Accuracy is shown both without error correction (Accuracy) and with error correction (Accuracy-C).

Конфигурация модели	ESR	Accuracy	Accuracy-C
Модель без дополнительных функций	85.43	54.69	55.82
Модель с дополнительными функциями	86.84	56.60	57.25

Полученные результаты на WTQ подтверждают, что предлагаемое компактное семантическое представление таблиц успешно масштабируются на задачи открытого вопросно-ответного взаимодействия, обеспечивая конкурентоспособный уровень точности.

5. Заключение

В данной статье предложен новый подход к решению задач табличных рассуждений, основанный на компактном семантическом представлении табличных данных и генерации исполняемых Pandas-запросов. Проведенные эксперименты на тестовом наборе данных TabFact подтвердили, что обогащение схемы таблицы типами данных, семантическими ролями и контекстуальными описаниями позволяет существенно сократить объем входной последовательности таблиц без потери информативности, обеспечивая масштабируемость к крупным таблицам. Использование LoRA-адаптера для тонкой настройки модели DeepSeek-coder-7B продемонстрировало конкурентоспособную точность (80.15%) при высокой исполнимости кода (95.38%) в сравнении с базовым решением RePanda.

Следует отметить, что в данном исследовании результаты приводятся для лучших контрольных точек (*checkpoint*), отобранных по минимальному значению функции потерь на валидации, без усреднения по нескольким случайным начальным приближениям (*seeds*). Обусловлено это высокими вычислительными затратами и временными ресурсами, необходимыми для многократного полного цикла дообучения и получения итоговых ответов от LLM на объемных табличных данных. Тем не менее, использование настройки при генерации кода и строгая фиксация гиперпараметров обеспечивают высокую воспроизводимость полученных результатов. Отсутствие статистических допусков (+/-) и дисперсии является ограничением текущей работы, и проведение экспериментов с множественными запусками модели для оценки устойчивости результатов запланировано в качестве одного из приоритетных направлений дальнейших исследований.

По результатам проведенных исследований и анализа выявленных групп ошибок в дальнейшем также планируется:

- осуществить поиск способа исправления выявленных ошибок;
- улучшить текущую систему коррекции ошибок для генерации корректного

исполняемого Pandas-кода;

- провести анализ применимости компактного семантического представления таблиц на других моделях.

Список литературы

[1]. Borisova E., Barth F., Feldhus N., Ahmad R.A., Ostendorff M., Suarez P.O., Rehm G., Möller S. Table Understanding and (Multimodal) LLMs: A Cross-Domain Case Study on Scientific vs. Non-Scientific Data. Proc. the 4th Table Representation Learning Workshop (TRL). Association for Computational Linguistics, Vienna, Austria, 2025, pp. 109-142. DOI: 10.18653/v1/2025.trl-1.10.

[2]. Zhang X., Wang D., Dou L., Zhu Q., Che W. A survey of table reasoning with large language models. Frontiers of Computer Science, 2025, vol. 19, 199348. DOI: 10.1007/s11704-024-40330-z.

[3]. Jin N., Siebert J., Li D., Chen Q. A Survey on Table Question Answering: Recent Advances. Knowledge Graph and Semantic Computing: Knowledge Graph Empowers the Digital Economy (CCKS'2022). Communications in Computer and Information Science, 2022, vol. 1669, pp. 174-186. DOI: 10.1007/978-981-19-7596-7_14.

[4]. Zhang H., Wang Y., Wang S., Cao X., Zhang F., Wang Z. Table Fact Verification with Structure-Aware Transformer. Proc. the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 2020, pp. 1624-1629. DOI: 10.18653/v1/2020.emnlp-main.126.

[5]. Chen M., Lu X., Xu T., Li Y., Jingbo Z., Dou D., Xiong H. Towards Table-to-Text Generation with Pretrained Language Model: A Table Structure Understanding and Text Deliberation Approach. Proc. the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), Abu Dhabi, UAE, 2022, pp. 8199-8210. DOI: 10.18653/v1/2022.emnlp-main.562.

[6]. Shi L., Tang Z., Zhang N., Zhang X., Yang Z. A Survey on Employing Large Language Models for Text-to-SQL Tasks. ACM Computing Surveys, 2025, vol. 58, no. 2, pp. 1-37. DOI: 10.1145/3737873.

[7]. Pasupat P., Liang P. Compositional Semantic Parsing on Semi-Structured Tables. Proc. the 53rd Annual Meeting of the Association for Computational Linguistics (ACL) and the 7th International Joint Conference on Natural Language Processing (IJCNLP), Beijing, China, 2015, pp. 1470-1480. DOI: 10.3115/v1/P15-1142.

[8]. Nan L., Hsieh C., Mao Z., Lin X.V., Verma N., Zhang R., Kryściński W., Schoelkopf H., Kong R., Tang X., Mutuma M., Rosand B., Trindade I., Bandaru R., Cunningham J., Xiong C., Radev D. FeTaQA: Free-form Table Question Answering. Transactions of the Association for Computational Linguistics, 2022, vol. 10, pp. 35-49. DOI: 10.1162/tacl_a_00446.

[9]. Chen W., Wang H., Chen J., Zhang Y., Wang H., Li S., Zhou X., Wang W.Y. TabFact: A Large-scale Dataset for Table-based Fact Verification. Proc. the International Conference on Learning Representations (ICLR'2020), Addis Ababa, Ethiopia, 2020, pp. 1-25.

[10]. Zhong V., Xiong C., Socher R. Seq2SQL: Generating Structured Queries From Natural Language Using Reinforcement Learning. Proc. 6th International Conference on Learning Representations (ICLR'2018), Vancouver, Canada, 2018, pp.1-12.

[11]. Sun Y., Tang D., Duan N., Ji J., Cao G., Feng X., Qin B., Liu T., Zhou M. Semantic Parsing with Syntax- and Table-Aware SQL Generation. Proc. the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 361-372. DOI: 10.18653/v1/P18-1034.

[12]. Liu Q., Chen B., Guo J., Ziyadi M., Lin Z., Chen W., Lou J.-G. TAPEX: Table Pre-training via Learning a Neural SQL Executor. Proc. 10th International Conference on Learning Representations (ICLR'22), Online, 2022, pp. 1-19. DOI: 10.48550/arXiv.2107.07653.

[13]. Wang Z., Zhang H., Li C.-L., Eisenschlos J. M., Perot V., Wang Z., Miculicich L., Fujii Y., Shang J., Lee C.-Y., Pfister T. Chain-of-Table: Evolving Tables in the Reasoning Chain for Table Understanding. Proc. the 12th International Conference on Learning Representations (ICLR'2024), Vienna, Austria, 2024, pp. 1-24.

[14]. Jin Z., Lu W. Tab-CoT: Zero-shot Tabular Chain of Thought. Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, 2023, pp. 10259-10277. DOI: 10.18653/v1/2023.findings-acl.651.

[15]. Lu X., Pan L., Ma Y., Nakov P., Kan M.-Y. TART : An Open-Source Tool-Augmented Framework for Explainable Table-Based Reasoning. Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, 2025, pp. 4323-4339. DOI: 10.18653/v1/2025.findings-naacl.244.

- [16]. Yu X., Jian P., Chen C. TableRAG: A Retrieval Augmented Generation Framework for Heterogeneous Document Reasoning. Proc. the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP), Suzhou, China, 2025, pp. 14063-14082. DOI: 10.18653/v1/2025.emnlp-main.710.
- [17]. Zhang D., Yu Y., Dong J., Li C., Su D., Chu C., Yu D. MM-LLMs: Recent Advances in MultiModal Large Language Models. Findings of the Association for Computational Linguistics: ACL 2024, Bangkok, Thailand, 2024, pp. 12401-12430. DOI: 10.18653/v1/2024.findings-acl.738.
- [18]. Jiang Z., Mao Y., He P., Neubig G., Chen W. OmniTab: Pretraining with Natural and Synthetic Data for Few-shot Table-based Question Answering. Proc. the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'2022), Seattle, USA, 2022, pp. 932-942. DOI: 10.18653/v1/2022.naacl-main.68.
- [19]. Herzig J., Nowak P. K., Müller T., Piccinno F., Eisenschlos J. TaPas: Weakly Supervised Table Parsing via Pre-training. Proc. the 58th Annual Meeting of the Association for Computational Linguistics (ACL), Online, 2020, pp. 4320-4333. DOI: 10.18653/v1/2020.acl-main.398.
- [20]. Zhu F., Lei W., Huang Y., Wang C., Zhang S., Lv J., Feng F., Chua T.-S. TAT-QA: A Question Answering Benchmark on a Hybrid of Tabular and Textual Content in Finance. Proc. the 59th Annual Meeting of the Association for Computational Linguistics (ACL) and the 11th International Joint Conference on Natural Language Processing (IJCNLP), Online, 2021, pp. 3277-3287. DOI: 10.18653/v1/2021.acl-long.254.
- [21]. Liu Y.-Y., Zheng Z., Zhang F., Feng J.-C., Fu Y.-Y., Zhai J.-D., He B.-S., Zhang X., Du X.-Y. A comprehensive taxonomy of prompt engineering techniques for large language models. *Frontiers of Computer Science*, 2026, vol. 20, 2003601. DOI: 10.1007/s11704-025-50058-z.
- [22]. Yu P., Chen G., Wang J. Table-Critic: A Multi-Agent Framework for Collaborative Criticism and Refinement in Table Reasoning. Proc. the 63rd Annual Meeting of the Association for Computational Linguistics (ACL), Vienna, Austria, 2025, pp. 17432-17451. DOI: 10.18653/v1/2025.acl-long.853.
- [23]. Chegini A., Rezaei K., Eghbalzadeh H., Feizi S. RePanda: Pandas-powered Tabular Verification and Reasoning. Proc. the 63rd Annual Meeting of the Association for Computational Linguistics (ACL), Vienna, Austria, 2025, pp. 32200-32212. DOI: 10.18653/v1/2025.acl-long.1549.
- [24]. Nguyen T.-N., Ngo H., Phung D., Vu T.-T., Nguyen D.Q. Planning for Success: Exploring LLM Long-term Planning Capabilities in Table Understanding. Proc. the 29th Conference on Computational Natural Language Learning (CoNLL), Association for Computational Linguistics, Vienna, Austria, 2025, pp. 81-92. DOI: 10.18653/v1/2025.conll-1.6.
- [25]. Urban M., Binnig C. ELEET: Efficient Learned Query Execution over Text and Tables. *Proceedings of the VLDB Endowment*, 2025, vol. 17, no. 13, pp. 4867-4880. DOI: 10.14778/3704965.3704989.
- [26]. Sui Y., Zhou M., Zhou M., Han S., Zhang D. Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study. Proc. the 17th ACM International Conference on Web Search and Data Mining (WSDM'24), Merida, Mexico, 2024, pp. 645-654. DOI: 10.1145/3616855.3635752.
- [27]. Zhou W., Mesgar M., Adel H., Friedrich A. RITT: A Retrieval-Assisted Framework with Image and Text Table Representations for Table Question Answering. Proc. the 4th Table Representation Learning Workshop (TRL), Association for Computational Linguistics, Vienna, Austria, 2025, pp. 86-97. DOI: 10.18653/v1/2025.trl-1.8.
- [28]. He X., Liu Y., Zhou M., He Y., Dong H., Han S., Yuan Z., Zhang D. TableLoRA: Low-rank Adaptation on Table Structure Understanding for Large Language Models. Proc. the 63rd Annual Meeting of the Association for Computational Linguistics (ACL), Vienna, Austria, 2025, pp. 22376-22391. DOI: 10.18653/v1/2025.acl-long.1090.
- [29]. Kim M.J., Grinsztajn L., Varoquaux G. CARTE: pretraining and transfer for tabular learning. Proc. the 41st International Conference on Machine Learning (ICML'24), Vienna, Austria, 2024, no. 956, pp. 23843-23866. DOI: 10.5555/3692070.3693026.
- [30]. Chen S., Miculicich L., Eisenschlos J.M., Wang Z., Wang Z., Chen Y., Fujii Y., Lin H.-T., Lee C.-Y., Pfister T. TableRAG: Million-Token Table Understanding with Language Models. Proc. the 38th Conference on Neural Information Processing Systems (NIPS'24), Vancouver, Canada, 2024, pp. 74899-74921. DOI: 10.5555/3737916.3740298.
- [31]. Mateiu P., Groza A. Ontology engineering with Large Language Models. Proc. the 25th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Nancy, France, 2023, pp. 226-229. DOI: 10.1109/SYNASC61333.2023.00038.
- [32]. Ji S., Pan S., Cambria E., Marttinen P., Yu P.S. A Survey on Knowledge Graphs: Representation, Acquisition and Applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021, vol. 33, no. 2, pp. 494-514. DOI: 10.1109/TNNLS.2021.3070843.
- [33]. Pandas, Available at: <https://pandas.pydata.org/>, accessed 09.04.2026.
- [34]. Liu J., Chabot Y., Troncy R., Huynh V.-P., Labbe T., Monnin P. From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods. *Journal of Web Semantics*, 2023, vol. 76, 100761. DOI: 10.1016/j.websem.2022.100761.
- [35]. Suhara Y., Li J., Li Y. Annotating Columns with Pre-trained Language Models. Proc. the 2022 International Conference on Management of Data (SIGMOD'22), 2022, pp. 1493-1503. DOI: 10.1145/3514221.3517906.
- [36]. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proc. the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), 2019, pp. 4171-4186. DOI: 10.18653/v1/N19-1423.
- [37]. Zhang D., Suhara Y., Li J., Hulsebos M., Demiralp C., Tan W.-C. Sato: Contextual semantic type detection in tables. *Proceedings of the VLDB Endowment*, 2020, vol. 13, no. 11, pp. 1835-1848. DOI: 10.14778/3407790.3407793.
- [38]. Bizer C., Lehmann J., Kobilarov G., Auer S., Becker C., Cyganiak R., Hellmann S. DBpedia - A Crystallization Point for the Web of Data. *Journal of Web Semantics*, 2009, vol. 7, no. 3, pp. 154-165. DOI: 10.1016/j.websem.2009.07.002.
- [39]. Singha A., Cambronero J., Gulwani S., Le V., Parnin C. Tabular Representation, Noisy Operators, and Impacts on Table Structure Understanding Tasks in LLMs. Proc. the Table Representation Learning Workshop at NeurIPS 2023, New Orleans, USA, 2023, pp. 1-14.
- [40]. Sui Y., Zhou M., Zhou M., Han S., Zhang D. Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study. Proc. the 17th ACM International Conference on Web Search and Data Mining (WSDM'24), Merida, Mexico, 2024, pp. 645-654. DOI: 10.1145/3616855.3635752.
- [41]. Qwen3-Embedding-4B, Available at: <https://huggingface.co/Qwen/Qwen3-Embedding-4B>, accessed 09.04.2026.
- [42]. FAISS (Facebook AI Similarity Search), Available at: <https://github.com/facebookresearch/faiss>, accessed 09.04.2026.
- [43]. Qwen3-VL-30B-A3B-Thinking, Available at: <https://huggingface.co/Qwen/Qwen3-VL-30B-A3B-Thinking>, accessed 09.04.2026.
- [44]. Deepseek-Coder-7B-Instruct v1.5, Available at: <https://huggingface.co/deepseek-ai/deepseek-coder-7b-instruct-v1.5>, accessed 09.04.2026.
- [45]. Optuna: A hyperparameter optimization framework, Available at: <https://github.com/optuna/optuna>, accessed 09.04.2026.
- [46]. Dong Y., Ding J., Jiang X., Li G., Li Z., Jin Z. CodeScore: Evaluating Code Generation by Learning Code Execution. *ACM Transactions on Software Engineering and Methodology*, 2025, vol. 34, no. 3, pp. 1-22. DOI: 10.1145/3695991.
- [47]. Eisenschlos J., Krichene S., Müller T. Understanding tables with intermediate pre-training. Findings of the Association for Computational Linguistics: EMNLP 2020, Online, 2020, pp. 281-296. DOI: 10.18653/v1/2020.findings-emnlp.27.
- [48]. Shi Q., Zhang Y., Yin Q., Liu T. Logic-level Evidence Retrieval and Graph-based Verification Network for Table-based Fact Verification. Proc. the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), Punta Cana, Dominican Republic, 2021, pp. 175-184. DOI: 10.18653/v1/2021.emnlp-main.16.
- [49]. Ou S., Liu Y. Learning to Generate Programs for Table Fact Verification via Structure-Aware Semantic Parsing. Proc. the 60th Annual Meeting of the Association for Computational Linguistics (ACL), Dublin, Ireland, 2022, pp. 7624-7638. DOI: 10.18653/v1/2022.acl-long.525.
- [50]. Qwen3-Coder-Next, Available at: <https://huggingface.co/Qwen/Qwen3-Coder-Next>, accessed 07.07.2026.

Информация об авторах / Information about authors

Иван Александрович ПОДДУБНЫЙ – аспирант Института динамики систем и теории управления им. В.М. Матросова Сибирского отделения РАН (ИДСТУ СО РАН) с 2024 года и младший научный сотрудник ИДСТУ СО РАН с 2026 года. Сфера научных интересов: табличные рассуждения, ответы на вопросы по табличным данным, верификация фактов в таблицах, табличные представления, большие языковые модели, глубокое обучение.

Ivan Aleksandrovich PODDUBNYY is a postgraduate student at the Matrosov Institute of System Dynamics and Control Theory named SB RAS (ISDCT SB RAS) since 2024 and junior researcher at ISDCT SB RAS since 2026. Research interests: table reasoning, table question answering, table fact verification, table representation, large language models, and deep learning.

Никита Олегович ДОРОДНЫХ – кандидат технических наук ИДСТУ СО РАН с 2018 года, и старший научный сотрудник ИДСТУ СО РАН с 2021 года. Сфера научных интересов: автоматизация создания интеллектуальных систем и баз знаний, получение знаний на основе преобразования концептуальных моделей и электронных таблиц, графы знаний и семантические технологии, автоматическое понимание табличной информации, табличные рассуждения, большие языковые модели.

Nikita Olegovich DORODNYKH – Cand. Sci. (Tech.) at ISDCT SB RAS since 2018 and senior associate researcher at ISDCT SB RAS since 2021. Research interests: computer-aided development of intelligent systems and knowledge bases, knowledge acquisition based on the transformation of conceptual models and tables, knowledge graphs and semantic web, table understanding, table reasoning, large language models.

Алдар Сергеевич АРМАЕВ – стажер-исследователь ИДСТУ СО РАН с 2026 года. Сфера научных интересов: табличные рассуждения, ответы на вопросы по табличным данным, верификация фактов в таблицах, большие языковые модели.

Aldar Sergeevich ARMAEV is research intern at ISDCT SB RAS since 2026. Research interests: table reasoning, table question answering, table fact verification, large language models.