

ОГЛАВЛЕНИЕ

Т. Р. Айсин, Т. В. Шамардина ДЕТЕКЦИЯ ГАЛЛЮЦИНАЦИЙ НА ОСНОВЕ ВНУТРЕННИХ СОСТОЯНИЙ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ	1282–1305
О. М. Атаева, М. Г. Кобук ФОРМИРОВАНИЕ СТРУКТУРИРОВАННЫХ ПРЕДСТАВЛЕНИЙ НАУЧНЫХ ЖУРНАЛОВ ДЛЯ ИНТЕГРАЦИИ В ГРАФ ЗНАНИЙ И СЕМАНТИЧЕСКОГО ПОИСКА	1306–1323
О. М. Атаева, Н. П. Тучкова, К. Б. Теймуразов, А. Абдышов, М. Г. Кобук БИБЛИОТЕКА НАУЧНЫХ ПРЕДМЕТНЫХ ОБЛАСТЕЙ SciLibRu	1324–1345
П. А. Гавриков, А. К. Усманов, Д. Реваев, С. Н. Бузыканов ИНСТРУМЕНТ ДЛЯ ОПЕРАТИВНОЙ ДИАГНОСТИКИ ПАМЯТИ НЕЙРОСЕТЕВЫХ АРХИТЕКТУР ЯЗЫКОВЫХ МОДЕЛЕЙ	1346–1367
А. А. Докукин СИСТЕМА ТЕСТИРОВАНИЯ КОНТРОЛЛЕРОВ, ОСНОВАННАЯ НА РАСПОЗНАВАНИИ ТЕКСТА НА ЭКРАНЕ	1368–1384
В. П. Зыков, Л. М. Местецкий ПОСТ-КОРРЕКЦИЯ СЛАБОЙ РАСШИФРОВКИ БОЛЬШИМИ ЯЗЫКОВЫМИ МОДЕЛЯМИ В ИТЕРАЦИОННОМ ПРОЦЕССЕ РАСПОЗНАВАНИЯ РУКОПИСЕЙ	1385–1414
С. Ма, О. В. Сенько НЕКОТОРЫЕ ПОДХОДЫ К ПОВЫШЕНИЮ ТОЧНОСТИ ПРОГНОЗИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ АНСАМБЛЕВЫХ МЕТОДОВ	1415–1434
И. Д. Морозов, Л. М. Местецкий ПОИСК СЛОВ В РУКОПИСНОМ ТЕКСТЕ НА ОСНОВЕ ШТРИХОВОЙ СЕГМЕНТАЦИИ	1435–1453

Н. М. Пронина

АТРИБУЦИЯ АРХИВНЫХ РУКОПИСНЫХ ПИСЕМ

С ИСПОЛЬЗОВАНИЕМ СИАМСКИХ НЕЙРОННЫХ СЕТЕЙ

1454–1480

А. П. Халов, О. М. Атаева

АВТОМАТИЧЕСКИЕ И ПОЛУАВТОМАТИЧЕСКИЕ

МЕТОДЫ ПОСТРОЕНИЯ ГРАФА ЗНАНИЙ ПРЕДМЕТНОЙ ОБЛАСТИ

И РАСШИРЕНИЯ ОНТОЛОГИИ

1481–1519

СЛОВО О ВЛАДИМИРЕ АЛЕКСЕЕВИЧЕ СЕРЕБРЯКОВЕ

(12.05.1946 — 20.12.2024)

Сегодня никто не удивляется тому, что у академического института есть сайт, оцифрованная библиотека и все материалы можно найти удаленно, задав вопрос на естественном языке. Весь этот набор услуг стал возможен после внедрения технологии Semantic Web, которая позволила создавать структуры данных на основе онтологий на языке OWL. Именно этими технологиями семантического анализа занималась группа разработчиков под руководством профессора Владимира Алексеевича Серебрякова начиная с 1998 г.



В. А. Серебряков

Интерес к этому подходу в программировании, новому на тот период, был не случайным выбором модной темы. Весь период научной жизни Владимира Алексеевича был связан с изучением проблем представления знаний, интеллектуальным анализом данных, организацией доступа к ним. Научная и педагогическая жизнь Владимира Алексеевича всегда была связана с ведущими российскими вузами и академическими институтами. После окончания аспирантуры в 1975 г. он был принят на работу в Вычислительный центр Академии наук СССР (ныне Федеральный исследовательский центр «Информатика и управление» Российской академии наук). В этом академическом институте он работал, двигался по научной и административной лестнице, стал доктором физико-математических наук, профессором, руководителем многих проектов и научных коллективов, трудился до последних дней своей жизни.

Владимир Алексеевич начинал свой путь в науке с такими классиками отечественного программирования, как Святослав Сергеевич Лавров, Владимир Михайлович Курочкин, Николай Павлович Трифонов. Это была школа программирования,



В. А. Серебряков
и В. М. Курочкин

где ученые не только развивали свои идеи, но и преподавали в ведущих университетах. В этом коллективе зародились идеи единого языка программирования и «программирующей программы», было расширено классическое определение атрибутов Д. Кнута и введено понятие структурных атрибутов. Разработаны трансляторы на БЭСМ-6 для языков: Алгол-60, Паскаль, APL, Алгол-W, АБВ, Снобол-A, ЛИСП, Modula-2, Система программирования Супер, Геомал. Долгое время Владимир Алексеевич успешно преподавал теорию построения трансляторов в МГУ им. М. В. Ломоносова и на ФИЗТЕХе.

С 2000-х годов основным направлением деятельности В. А. Серебрякова и руководимого им коллектива стало создание распределенных информационных систем, в частности цифровых (электронных) библиотек. Была создана интегрированная система информационных ресурсов (ИСИР), которая включала несколько десятков структур и институтов Российской академии наук. На принципах ИСИР реализован информационный портал Российской академии наук (ИСИР РАН, чаще именуемый, например, в Википедии, как «Научный портал РАН») – информационный ресурс, который эксплуатируется уже более 20 лет; разработана семантическая библиотека научных предметных областей.

Владимир Алексеевич Серебряков останется к нашей памяти как талантливый ученый и педагог, специалист в области системного программирования, теории и практики построения компиляторов, семантических цифровых библиотек и интеграции данных. Идеи его исследований нашли применение в теории и практике программирования. Современное продолжение его работ реализуется в проектах его учеников и коллег.

Специальный выпуск журнала содержит работы, представленные на секции «Актуальные проблемы семантического анализа», посвященной памяти Владимира Алексеевича Серебряков, и рекомендованные программным комитетом конференции ММРО-2025 (22-я Всероссийская конференция с международным участием «Математические методы распознавания образов», г. Муром, 22–26 сентября 2025 года).

Редакторы-составители специального выпуска

д. ф.-м. н. О. В. Сенько, к. ф.-м. н. А. А. Докукин, к. ф.-м. н. Н. П. Тучкова

ДЕТЕКЦИЯ ГАЛЛЮЦИНАЦИЙ НА ОСНОВЕ ВНУТРЕННИХ СОСТОЯНИЙ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ

Т. Р. Айсин¹ [0009-0001-5863-3252], Т. В. Шамардина² [0009-0008-9033-6646]

^{1, 2}Московский физико-технический институт, г. Долгопрудный,
Московская обл., Россия

¹aysin.timur@gmail.ru, ²shamardina.tatiana@gmail.com

Аннотация

В последние годы большие языковые модели (Large Language Models, LLM) достигли значительных успехов в области обработки естественного языка и стали ключевым инструментом для решения широкого спектра прикладных и исследовательских задач. Однако с ростом их масштабов и возможностей все более острой становится проблема галлюцинаций – генерации ложной, недостоверной или несуществующей информации, представленной в достоверной форме. В связи с этим вопросы анализа природы галлюцинаций и разработки методов их выявления приобретают особую научную и практическую значимость.

В работе изучен феномен галлюцинаций в больших языковых моделях, рассмотрены их существующая классификация и возможные причины. На базе модели Flan-T5 также исследованы различия внутренних состояний модели при генерации галлюцинаций и верных ответов. На основе этих расхождений представлены два способа детектирования галлюцинаций: с помощью карт внимания и скрытых состояний модели. Эти методы протестированы на данных из бенчмарков HaluEval и Shroom 2024 в задачах суммаризации, ответов на вопросы, перефразирования, машинного перевода и генерации определений. Кроме того, исследована переносимость обученных детекторов между различными типами галлюцинаций, что позволило оценить универсальность предложенных методов для различных типов задач.

Ключевые слова: *большие языковые модели, галлюцинации, детекция, Flan-T5, обработка естественного языка, карты внимания, внутренние состояния, HaluEval, Shroom.*

ВВЕДЕНИЕ

Развитие технологий обработки естественного языка (Natural Language Processing, NLP) в последние годы неразрывно связано с появлением и совершенствованием больших языковых моделей (Large Language Models, LLM), основанных на архитектуре Трансформер (Transformer, [1]). LLM способны моделировать сложные семантические зависимости и демонстрируют высокий уровень генеративных возможностей.

Однако стремительное развитие таких систем выявило и ряд новых проблем, среди которых особое место занимает феномен галлюцинаций [2]. Под галлюцинациями понимаются случаи, когда модель генерирует фактически неверную, вымышленную или логически несогласованную информацию, которая при этом выглядит правдоподобно и убедительно. В результате галлюцинации снижают достоверность создаваемых систем, что критически важно при использовании LLM в сферах, где ошибки недопустимы, например в медицине или праве.

Актуальность настоящей работы обусловлена необходимостью разработки методов детекции галлюцинаций, позволяющих повысить надежность и прозрачность работы языковых моделей. Существующие подходы зачастую ограничиваются анализом выходного текста без учета внутренних механизмов работы модели или используют обращения к внешним источникам данных при генерации. Между тем анализ внутренних представлений, таких как скрытые состояния или карты внимания, может дать дополнительную информацию о том, как модель формирует распределение вероятностей токенов и в какой момент формируется недостоверный контент.

Целью исследования являются разработка и экспериментальная проверка методов детекции галлюцинаций, основанных на внутренних состояниях и механизмах внимания слоев моделей. В рамках работы исследована взаимосвязь между внутренними признаками модели и достоверностью генерируемого текста, а также разработаны способы выявления галлюцинаций без обращения к внешним источникам данных. Эксперименты проведены на открытых датасетах HaluEval [3] и Shroom 2024 [4], что позволило оценить эффективность и переносимость предложенных методов на различных типах задач.

ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

Проблема галлюцинаций в больших языковых моделях имеет комплексный характер и обусловлена сочетанием факторов, связанных с данными [5–7], процессом обучения [8, 9] и особенностями генерации текста [10, 11]. В зависимости от характера и степени отклонения от достоверности галлюцинации можно разделить на два основных типа [2]:

- 1) фактические галлюцинации (factuality): ответы модели расходятся с фактами реального мира, которые можно проверить;
- 2) верность инструкциям или контексту (faithfulness): ответы модели расходятся с инструкциями пользователя или контекстом.

Детектирование фактических галлюцинаций

Для детектирования фактических галлюцинаций можно использовать пайплайны контроля данных [12], которые выполняют разбиение сгенерированного ответа на отдельные фактические утверждения и осуществляют их автоматизированную проверку с опорой на источник (например, Википедию). Недостатком таких методов является необходимость в специальной инфраструктуре и дополнительной разработке специализированных компонентов для реализации указанных этапов обработки.

Когда нет возможности обращаться к другим источникам, можно использовать методы, анализирующие неуверенность модели во время генерации ответа [13]. Для работы этих методов необходим непосредственный доступ к весам и активациям моделей. В ситуациях, когда веса моделей недоступны, имеется несколько техник детекции галлюцинаций на основе промптинга [14] или использования LLM-«экзаменатора» [15].

Детекция галлюцинаций, не соответствующих инструкциям или контексту

При детекции галлюцинаций, связанных с искажением контекста или инструкций пользователя, применяются в том числе методы на основе анализа неуверенности модели путем проверки энтропии сгенерированного текста [16] или вероятностей токенов на различных стадиях генерации ответов [17]. Известны также различные техники промптинга, в которых используются дополнительные

запросы, сформированные по специальной структуре для оценивания соответствия ответа запросу [18].

Дополнительно можно определять различные показатели, которые указывают, насколько пересекаются сгенерированный контент и исходный запрос. Эти показатели могут отражать количество n -грамм [19], именованных сущностей (*NER*) [20] и отношений между этими сущностями [21]. Кроме того, можно использовать модели, обученные на задаче текстового следования [22], исходя из предположения, что верный ответ модели должен являться продолжением запроса пользователя.

При детекции галлюцинаций данного типа можно также рассматривать паттерны карт внимания, которые модель распределяет между словами входной и выходной последовательностей. В [23] авторы отмечают, что при возникновении галлюцинаций в задаче машинного перевода модель склонна фокусироваться лишь на небольшом числе начальных слов входной последовательности. В [24] предложено выделять ключевые слова в исходном тексте и анализировать, насколько большое внимание им уделялось при генерации ответа. В [25] использована мера доли внимания, направленного на входные токены, относительно общего объема внимания. Такой подход позволяет не только выявлять галлюцинации после генерации, но и интегрировать механизм контроля в сам процесс декодирования, предотвращая их появление на ранних этапах генерации.

Существуют также подходы, направленные на исследование взаимосвязи между внутренними состояниями моделей и проявлениями галлюцинаций. В [26] авторы анализируют корреляцию между внутренними представлениями модели и ее уверенностью в собственных ответах. В [27] выявлены структурные закономерности в пространстве скрытых состояний, соответствующих ответам на бинарные («да/нет») вопросы. В [28] предложен фреймворк для детекции галлюцинаций, включающий автоматическую генерацию данных с галлюцинациями и последующий анализ различий во внутренних состояниях модели при обработке сгенерированных последовательностей.

ИСПОЛЬЗОВАНИЕ ВНУТРЕННИХ СОСТОЯНИЙ МОДЕЛЕЙ ПРИ ДЕТЕКЦИИ ГАЛЛЮЦИНАЦИЙ

Настоящая работа посвящена детекции галлюцинаций, не соответствующих инструкциям или контексту. Ранее для проверки корректности ответов преимущественно использовались статистические критерии или крупные специализированные модели. В отличие от них, нашей целью является исследование пространства внутренних состояний языковых моделей при генерации и использование этих состояний в качестве признаков для обучения легковесных нейросетевых классификаторов. С их помощью решается задача бинарной классификации: определить, является ли сгенерированный ответ галлюцинацией или корректным.

В качестве признаков рассмотрим два типа внутренних состояний: карты перекрестного внимания в соответствующих блоках и выходы различных слоев модели при генерации. Для каждого из этих типов обучен специальный классификатор и проведена оценка его работы в различных задачах.

ИСПОЛЬЗУЕМЫЕ ДАТАСЕТЫ И МОДЕЛИ

Для детекции галлюцинаций были использованы два датасета: данные из соревнования Shroom 2024 [4] и бенчмарк HaluEval [3].

Данные из соревнования Shroom содержат сложнодетектируемые примеры галлюцинаций в отдельных задачах, таких как моделирование определений (definition modeling, DM), машинный перевод (machine translation, MT) и перефразирование (paraphrase generation, PG). Несмотря на большое количество данных в этом соревновании, число размеченных примеров невелико – 400 примеров каждого вида.

HaluEval – это масштабный бенчмарк для оценки галлюцинаций больших языковых моделей. Мы использовали подвыборки этого бенчмарка, соответствующие задачам «вопрос – ответ» (question answering, QA) и суммаризации (summarization), в каждой из которых содержится 10000 размеченных примеров. Каждый пример подвыборки состоит из запроса, передаваемого на вход модели, примера корректного ответа и примера галлюцинации для данного запроса.

В качестве моделей для экспериментов были использованы модели из семейства Flan-T5 [29], представляющие собой улучшенные версии модели T5 (Text-To-Text-Transfer-Transformer). Модели этого семейства, как и большинство больших языковых моделей на сегодняшний день, были обучены для решения большого числа различных задач в области NLP, поэтому внутренние состояния этих моделей потенциально репрезентативны и имеют развитую структуру, что позволяет использовать их для анализа. Благодаря этому также возможно оценить работу методов при различных сценариях. Для экспериментов была использована версия flan-t5-base, которая имеет 248 млн параметров.

ПРИЗНАКИ НА ОСНОВЕ CROSS-ATTENTION

В рамках экспериментов было установлено, что в случаях галлюцинаций и корректного ответа перекрестное внимание (cross-attention) модели фокусируется на разных частях входной последовательности.

Каждый декодер-слой $l \in 1, \dots, L$ в сети модели имеет несколько голов (heads) $[A^1_1, A^1_2, \dots, A^1_h]$, где h – количество голов, A^l_i – матрица, у которой количество столбцов равно числу токенов входной последовательности, а строка – количеству токенов, сгенерированных в данный момент времени. Каждый элемент этой матрицы означает *важность* входного токена для выходного, которая выражается вещественным числом от 0 до 1. Нахождение максимума среди слоев и голов для входного токена a и выходного токена b – $\max_{i=1..h, j=1..L} (A^j_i)_{b,a}$ – позволяет выделить важные токены из контекста.

Ниже на двух примерах данных из HaluEval QA визуализированы различия в работе механизма перекрестного внимания при генерациях корректных ответов и галлюцинаций (см. рис. 1 и 2).

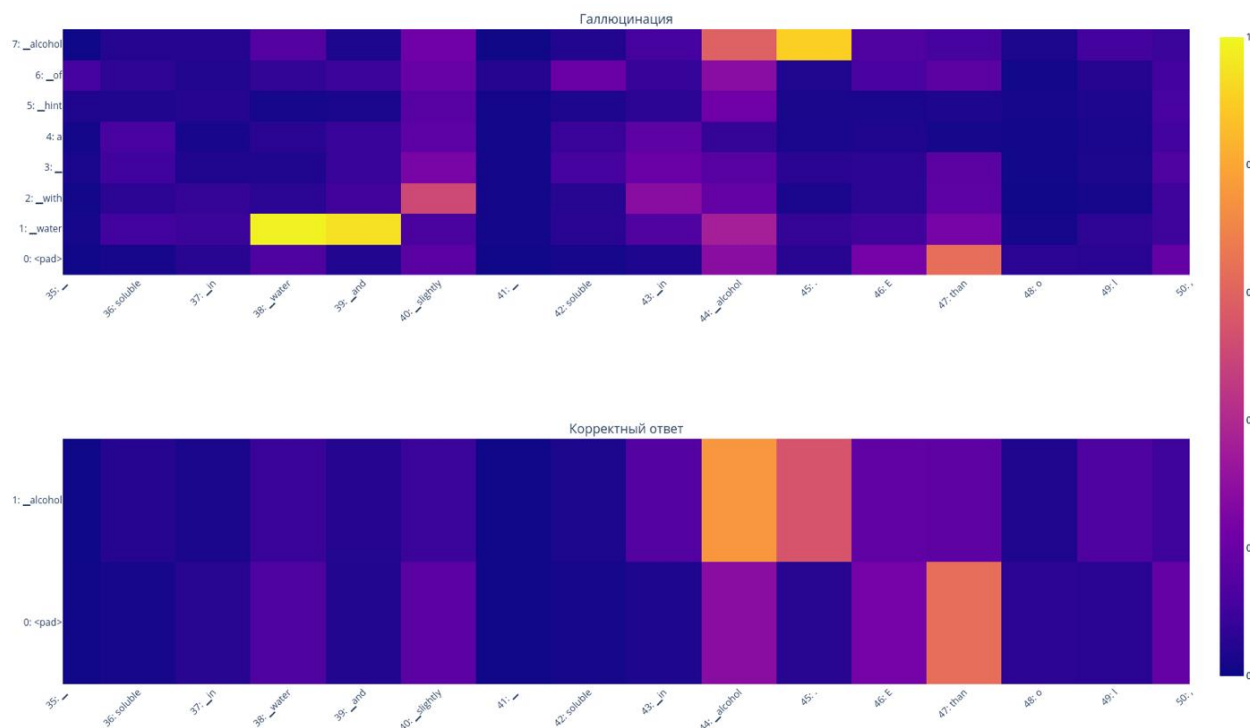


Рис. 1. При галлюцинации модель сильно фокусировалась на 38-м токене ‘_water’ и 39-м токене ‘_and’, а при корректном ответе на 44-м токене ‘_alcohol’.

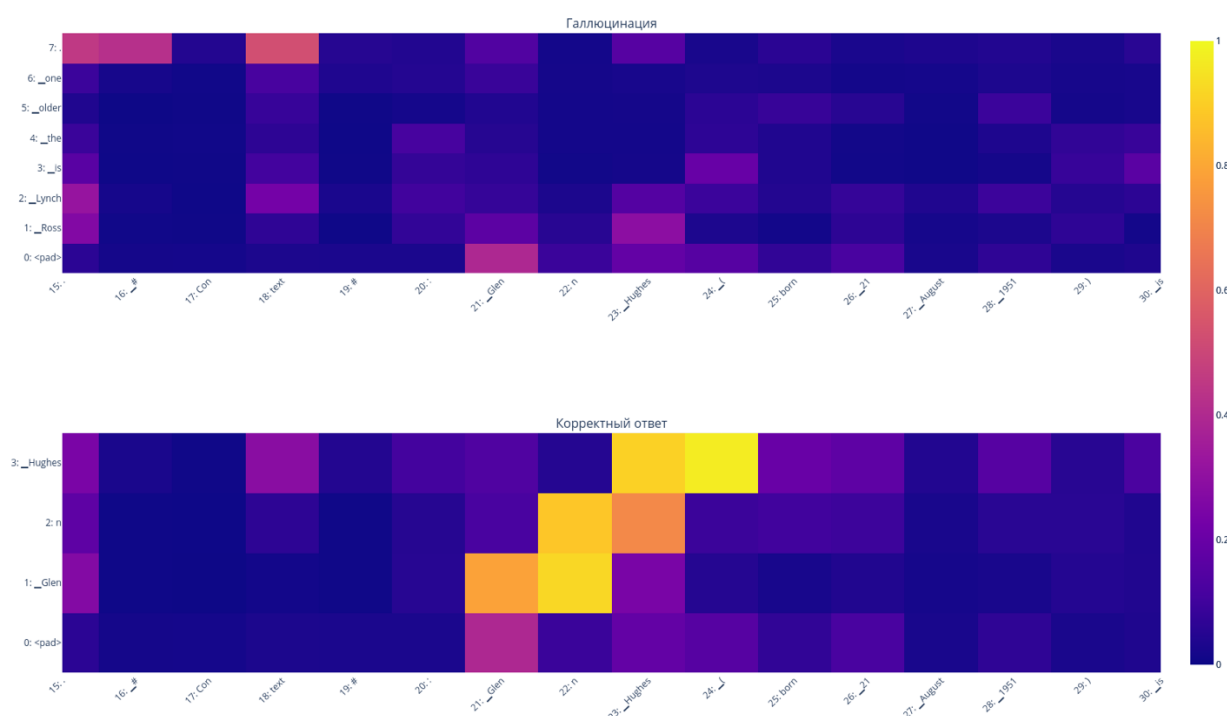


Рис. 2. При корректном ответе модель должна сильно фокусироваться на токенах 21 – 24: ‘_Glen’, ‘n’, ‘_Hughes’, ‘_('. В случае галлюцинации модель не уделила этому сегменту текста внимания.

Карты перекрестного внимания заметно различаются между корректными ответами и галлюцинациями. Поэтому для проверки гипотезы об их использовании для детекции галлюцинаций была предложена следующая архитектура (см. рис. 3).

1. Входные данные x представляют собой тензор, состоящий из карт активаций внимания $[[A^l_{i_1}, A^l_{i_2}, \dots, A^l_{i_h}], \dots, [A^{l+k}_{i_1}, A^{l+k}_{i_2}, \dots, A^{l+k}_{i_h}]]$ между токенами выходной и входной последовательностей, взятых с k последовательных слоев сети.

2. Первый блок детектора обрабатывает карты внимания различных голов модели. Он состоит из линейных преобразований, размер которых последовательно уменьшается (2790 \rightarrow 1395 \rightarrow 697 \rightarrow 348 \rightarrow 174), с функцией активации LeakyReLU между ними. В результате получается тензор x_r , в котором информация из карт внимания различных голов агрегирована в вектор меньшей размерности послойно.

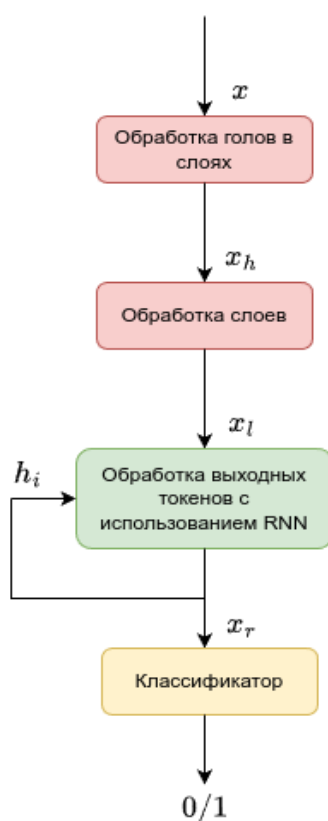


Рис. 3. Архитектура классификатора на основе перекрестного внимания.

3. После обработки голов получившиеся тензоры конкатенируются по-слоино, и для них аналогично применяются линейные трансформации с размерами (512 \rightarrow 256) и функции активации, как в предыдущем пункте. Таким образом, в x_l для каждого выходного токена собрана информация о важных токенах из контекста из разных слоев языковой модели.

4. Затем рекуррентная сеть LSTM [30] обрабатывает последовательности векторов из предыдущего пункта для каждого выходного токена. Такой способ позволяет анализировать, как использование информации из контекста изменяется во время генерации новых токенов.

5. Последнее внутреннее состояние данного слоя передается в классификатор. На выходе классификатор формирует вероятность того, что ответ является галлюцинацией.

Так как длина ответов может быть разной, необходимо обрезать слишком длинные или дополнять короткие ответы до одинаковой длины. В HaluEval QA галлюцинированные ответы в среднем длиннее, поэтому был также проведен эксперимент с обрезанием ответа до 4 выходных токенов.

Результаты данного алгоритма на HaluEval QA в зависимости от выбранных слоев представлены в табл. 1. Все эксперименты проводились в конфигурации, указанной в табл. 2.

Табл. 1. Результаты различных конфигураций алгоритма на HaluEval QA

Длина выходной последовательности	Слои flan-t5-base	F1-Score
32	[0, 1, 2, 3]	0.978
	[4, 5, 6, 7]	0.983
	[8, 9, 10, 11]	0.974
4	[0, 1, 2, 3]	0.852
	[4, 5, 6, 7]	0.883
	[8, 9, 10, 11]	0.849

Получившаяся сеть имеет порядка 22М параметров, что составляет примерно 9 % от размера `flan-t5-base`. Такой подход к детекции галлюцинаций не требует переобучения большой модели и показывает высокое качество классификации по F1-мере на тестовой выборке.

Несмотря на высокую точность, алгоритм имеет архитектурное ограничение: входные векторы чрезвычайно разрежены и обладают большой размерностью, что усложняет обработку длинных последовательностей. Поэтому для корректной работы с длинными текстами необходимы дополнительные изменения (например, разделение текста на куски более короткой длины и агрегирование результатов), выбор которых требует отдельного исследования.

Табл. 1. Параметры обучения

Параметр	Значение
Размер батча	64
Количество эпох	10
Размер обучающей выборки	16000
Размер тестовой выборки	4000
Скорость обучения (<i>learning rate</i>)	0.0001
Размер входной последовательности (с учетом паддинга)	465

ПРИЗНАКИ НА ОСНОВЕ ВЫХОДОВ РАЗЛИЧНЫХ СЛОЕВ МОДЕЛЕЙ

В ходе экспериментов было установлено, что выходные представления слоев модели при генерации галлюцинаций и корректных ответов также различаются. В отличие от карт перекрестного внимания, размер этих векторов не зависит от длины входной и выходной последовательностей и фиксируется на этапе обучения языковой модели. Благодаря этому использование признаков, построенных на основе внутренних представлений, не накладывает ограничений на размер контекста и позволяет эффективно детектировать галлюцинации в задачах с большим объемом текста, например, в `HaluEval Summarization`.

На рис. 4 представлены проекции скрытых внутренних состояний модели при генерации последнего токена в 1000 примерах суммаризации, полученные

с помощью PCA [31] (0-й слой – это слой, куда приходят выходы энкодера). Красным цветом выделены состояния, соответствующие галлюцинациям, синим – корректным ответам.

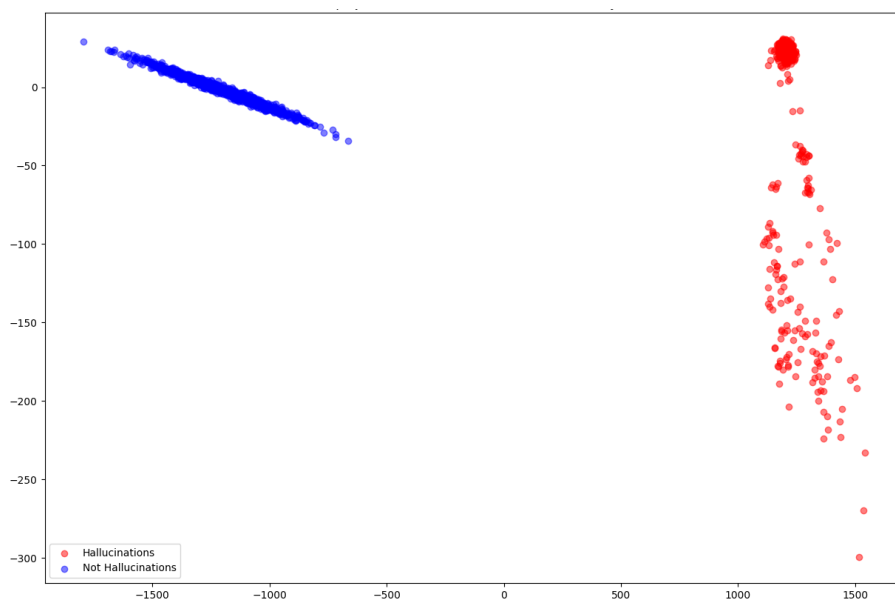


Рис. 4. PCA: Выходы 0-го слоя

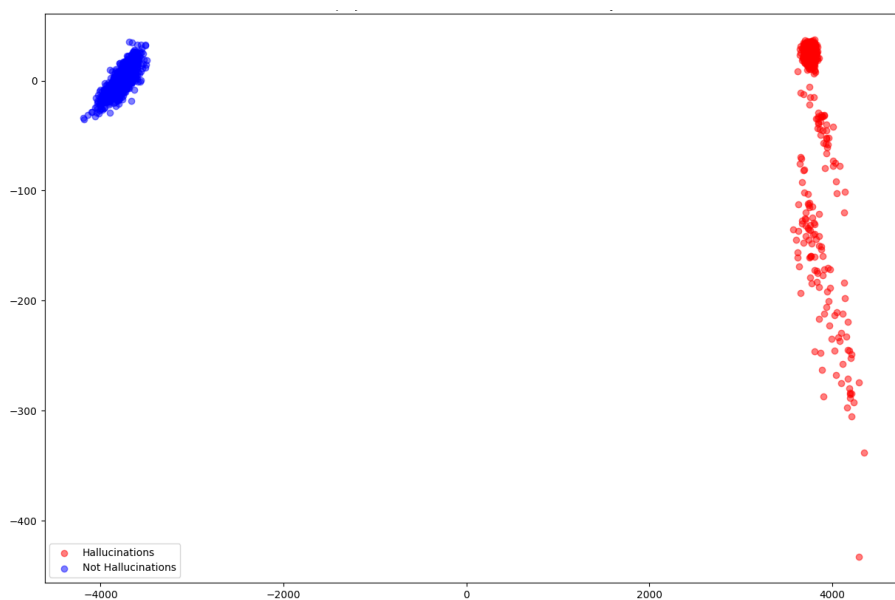


Рис. 5. PCA: Выходы 1-го слоя.

Как видно из проекций, внутренние состояния оказались довольно хорошо различимы. На основе этих значений также был обучен детектор галлюцинаций. Детектор представляет собой нейронную сеть, состоящую из 3 линейных слоев и функций активации LeakyReLU между ними, на вход которой приходят активации с различных слоев при генерации последнего токена. В табл. 3 и 4 ниже представлены избранные результаты детектора, использующего активации с различных слоев в качестве входных данных, на задаче HaluEval Summarization.

Табл. 2. Результаты детектора на HaluEval Summarization в зависимости от слоя модели flan-t5-base (нумерация с 0)

Номер слоя	F1-Score
0	0.305
1	0.757
2	0.859
6	0.924
7	0.944
8	0.951
11	0.861

Табл. 3. Параметры обучения

Параметр	Значение
Размер батча	128
Количество эпох	10
Размер обучающей выборки	18000
Размер тестовой выборки	2000
Скорость обучения (<i>learning rate</i>)	0.0001
Размер входной последовательности (с учетом паддинга)	1600
Размер выходной последовательности (с учетом паддинга)	128

Из результатов эксперимента видно, что распределения внутренних состояний слоев сильно отличаются при генерации галлюцинаций и верных ответов почти во всех слоях сети. Лучшие результаты получаются для слоев 6–8. Кроме того, достоинство этого метода заключается в том, что он работает независимо от длин входной и выходной последовательностей: выходы слоев всегда имеют одну размерность, задаваемую архитектурой исходной языковой модели. Получившийся детектор имеет порядка 566 тыс. параметров, что составляет примерно 0.2 % от размера исходной языковой модели.

ПЕРЕНОСИМОСТЬ МЕЖДУ РАЗЛИЧНЫМИ ТИПАМИ ГАЛЛЮЦИНАЦИЙ

Для проверки переносимости результатов детекторы, обученные на задаче HaluEval Summarization, были запущены на галлюцинациях других типов: HaluEval QA, Shroom MT, Shroom PG, Shroom DM (табл. 5 и 6).

Табл. 4. Результаты классификатора (F1-Score), обученного на HaluEval Summarization, на других задачах.

Номер слоя	HaluEval Summarization	HaluEval QA	Shroom PG	Shroom DM	Shroom MT
0	0.305	0.002	0.362	0.661	0.562
1	0.757	0.194	0.337	0.531	0.553
2	0.859	0.365	0.267	0.294	0.464
6	0.924	0.56	0.256	0.094	0.46
7	0.944	0.632	0.358	0.472	0.531
8	0.951	0.595	0.283	0.313	0.482
11	0.861	0.514	0.283	0.607	0.554

Эксперименты показали, что особенности распределений слоев сети, присущие галлюцинациям в задаче суммаризации, плохо переносятся на галлюцинации других типов. Таким образом можно сделать вывод, что состояния модели при генерации галлюцинаций различных типов имеют разное распределение, и для построения «универсального» детектора необходимы примеры галлюцинаций в разных задачах.

Табл. 5. Результаты классификатора (F1-Score), обученного на HaluEval QA, на других задачах.

Номер слоя	HaluEval Summarization	HaluEval QA	Shroom PG	Shroom DM	Shroom MT
0	0.666	0.665	0.362	0.662	0.562
1	0.662	0.910	0.352	0.658	0.534
2	0.665	0.959	0.362	0.66	0.561
6	0.666	0.975	0.361	0.661	0.562
7	0.666	0.978	0.361	0.661	0.562
8	0.667	0.980	0.362	0.662	0.561
11	0.663	0.965	0.36	0.661	0.561

При обучении детектора на задаче HaluEval QA также наблюдаются различия в зависимости используемых слоев, но в значительно меньшей степени, чем в задаче суммаризации. Аналогично задаче QA, детекторы на основе слоев с 6 по 8 демонстрируют наилучшие результаты. Лучшее качество получается при использовании 8-го слоя, что незначительно меньше, чем качество, полученное с помощью детектора на основе перекрестного внимания: 0.98 (табл. 6) против 0.983 (табл. 1).

При последующем применении данного детектора к другим задачам была снова подтверждена плохая переносимость между различными типами галлюцинаций. Примечательно, что при использовании классификатора, обученного на задаче QA, выбор входного слоя модели практически не влияет на качество детекции в других задачах. Это может свидетельствовать о том, что при генерации суммаризаций распределения активаций по слоям отличаются между собой значительно сильнее (табл. 5), чем в задаче QA.

ЗАКЛЮЧЕНИЕ

Рассмотрен феномен галлюцинаций в больших языковых моделях с фокусом на случаи, возникающие из-за несоответствия между входными и выходными данными. Для автоматической детекции таких галлюцинаций на датасетах

Shroom и HaluEval была использована внешняя модель *flan-t5-base*, а также предложены и исследованы подходы на основе скрытых состояний и карт внимания.

Проведенные эксперименты показали, что внутренние представления модели при генерации галлюцинированных и корректных ответов имеют различия, которые можно использовать для построения эффективных детекторов. В частности, методы, опирающиеся на скрытые состояния, продемонстрировали более широкую применимость и аналогичную точность по сравнению с подходами на основе внимания, особенно при работе с длинными входами, где последние требуют предварительной агрегации признаков. Это указывает на перспективность использования скрытых представлений как более универсального признакового пространства для задач детекции. В ходе экспериментов было установлено, что скрытые состояния промежуточных слоев (6–8 для модели *flan-t5-base*) наиболее информативны для задачи детекции галлюцинаций.

В рамках работы был реализован и обучен классификатор, использующий скрытые состояния модели и демонстрирующий высокое качество детекции галлюцинаций на ряде задач. Кроме того, была предпринята попытка построения модели на основе LSTM, принимающей визуализации карт внимания в качестве входа. Этот подход, несмотря на высокое качество классификации, имеет ограничения при работе с длинными последовательностями.

Отдельное внимание было уделено вопросу переносимости: детекторы, обученные на одном типе галлюцинаций, показывают ограниченную способность к переносу на другие задачи, что подчеркивает необходимость в более универсальных подходах и корпусах для обучения. Это направление остается открытым и требует дальнейшего изучения.

Одним из потенциально перспективных путей продолжения работы является применение статистических методов анализа и использование специальных выборок для изучения внутренних состояний моделей. Такие подходы могут способствовать выявлению более общих закономерностей и повышению переносимости детекторов между задачами.

СПИСОК ЛИТЕРАТУРЫ

1. Vaswani A., Shazeer N., Parmar N. et al. Attention is all you need // Advances in Neural Information Processing Systems. 2017. Vol. 30.
2. Huang L., Yu W., Ma W. et al. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions // ACM Transactions on Information Systems. 2025. Vol. 43, No. 2. P. 1–55.
<https://doi.org/10.1145/3703155>
3. Li J., Cheng X., Zhao W. X. et al. HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 6449–6464. <https://doi.org/10.18653/v1/2023.emnlp-main.397>
4. Mickus T., Zosa E., Vázquez R. et al. SemEval-2024 Task 6: SHROOM, a Shared-task on Hallucinations and Related Observable Overgeneration Mistakes // International Workshop on Semantic Evaluation. 2024.
<https://doi.org/10.18653/v1/2024.semeval-1.273>
5. Carlini N., Ippolito D., Jagielski M. et al. Quantifying Memorization Across Neural Language Models // The Eleventh International Conference on Learning Representations. 2023. <https://doi.org/10.48550/arXiv.2202.07646>
6. Lin S., Hilton J., Evans Q. TruthfulQA: Measuring How Models Mimic Human Falsehoods // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022. Vol. 1. P. 3214–3252.
<https://doi.org/10.18653/v1/2022.acl-long.229>
7. Li D., Rawat A.S., Zaheer M. et al. Large Language Models with Controllable Working Memory // Findings of the Association for Computational Linguistics: ACL 2023. 2023. P. 1774–1793.
<https://doi.org/10.18653/v1/2023.findings-acl.112>
8. Sharma M., Tong M., Korbak T. et al. Towards Understanding Sycophancy in Language Models // The Twelfth International Conference on Learning Representations. 2024. <https://doi.org/10.48550/arXiv.2310.13548>

9. Reinforcement Learning from Human Feedback: Progress and Challenges // YouTube. URL: https://www.youtube.com/watch?v=hhiLw5Q_UFg (дата обращения: 04.05.2025)
10. *Chuang Y.S., Xie Y., Luo H. et al.* DoLa: Decoding by Contrasting Layers Improves Factuality in Large Language Models // ArXiv. 2023. Vol. abs/2309.03883. <https://doi.org/10.48550/arXiv.2309.03883>
11. *Voita E., Talbot D., Moiseev F. et al.* Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 5797–5808. <https://doi.org/10.18653/v1/P19-1580>
12. *Min S., Krishna K., Lyu X. et al.* FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 12076–12100. <https://doi.org/10.18653/v1/2023.emnlp-main.741>
13. *Luo Z., Xie Q., Ananiadou S.* ChatGPT as a Factual Inconsistency Evaluator for Text Summarization // ArXiv. 2023. Vol. abs/2303.15621. <https://doi.org/10.48550/arXiv.2303.15621>
14. *Manakul P., Liusie A., Gales M.J.* SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 9004–9017. <https://doi.org/10.18653/v1/2023.emnlp-main.557>
15. *Cohen R., Hamri M., Geva M. et al.* LM vs LM: Detecting Factual Errors via Cross Examination // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 12621–12640. <https://doi.org/10.18653/v1/2023.emnlp-main.778>
16. *Xiao Y., Wang W.Y.* On Hallucination and Predictive Uncertainty in Conditional Language Generation // Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume. 2021. P. 2734–2744. <https://doi.org/10.18653/v1/2021.eacl-main.236>
17. *Miao N., Teh Y.W., Rainforth T.* SelfCheck: Using LLMs to Zero-Shot Check Their Own Step-by-Step Reasoning // The Twelfth International Conference on Learning Representations. 2024. <https://doi.org/10.48550/arXiv.2308.00436>

18. *Adlakha V., BehnamGhader P., Lu X.H. et al.* Evaluating Correctness and Faithfulness of Instruction-Following Models for Question Answering // Transactions of the Association for Computational Linguistics. 2024. Vol. 12. P. 681–699.
https://doi.org/10.1162/tacl_a_00667
19. *Lin Chin-Yew.* ROUGE: A Package for Automatic Evaluation of Summaries // Text Summarization Branches Out. 2004. P. 74-81. ISBN: 9781932432466
20. *Venkit P.N., Gautam S., Panchanadikar R. et al.* Nationality Bias in Text Generation // Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics. 2023. P. 116–122.
<https://doi.org/10.18653/v1/2023.eacl-main.9>
21. *Goodrich B., Rao V., Liu P.J. et al.* Assessing The Factual Accuracy of Generated Text // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2019. P. 166–175.
<https://doi.org/10.1145/3292500.3330955>
22. *Laban P., Schnabel T., Bennett P.N. et al.* SummaC: Re-Visiting NLI-based Models for Inconsistency Detection in Summarization // Transactions of the Association for Computational Linguistics. 2022. Vol. 10. P. 163–177.
https://doi.org/10.1162/tacl_a_00453
23. *Xu W., Agrawal S., Briakou E. et al.* Understanding and Detecting Hallucinations in Neural Machine Translation via Model Introspection // Transactions of the Association for Computational Linguistics. 2023. Vol. 11. P. 546–564.
https://doi.org/10.1162/tacl_a_00563
24. *Zhang T., Qiu L., Guo Q. et al.* Enhancing Uncertainty-Based Hallucination Detection with Stronger Focus // Conference on Empirical Methods in Natural Language Processing. 2023. <https://doi.org/10.18653/v1/2023.emnlp-main.58>
25. *Chuang Y.S., Qiu L., Hsieh C.Y. et al.* Lookback Lens: Detecting and Mitigating Contextual Hallucinations in Large Language Models Using Only Attention Maps // Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024. P. 1419–1436.
<https://doi.org/10.18653/v1/2024.emnlp-main.84>

26. Yin Z., Sun Q., Guo Q. et al. Do Large Language Models Know What They Don't Know? // Annual Meeting of the Association for Computational Linguistics. 2023. <https://doi.org/10.18653/v1/2023.findings-acl.551>
 27. Marks S., Tegmark M. The Geometry of Truth: Emergent Linear Structure in Large Language Model Representations of True/False Datasets // First Conference on Language Modeling. 2024. <https://doi.org/10.48550/arXiv.2310.06824>
 28. Su W., Wang C., Ai Q. et al. Unsupervised Real-Time Hallucination Detection based on the Internal States of Large Language Models // Annual Meeting of the Association for Computational Linguistics. 2024. <https://doi.org/10.48550/arXiv.2403.06448>
 29. Chung H.W., Hou L., Longpre S. et al. Scaling Instruction-Finetuned Language Models // Journal of Machine Learning Research. 2024. Vol. 25, No. 70. P. 1–53. <https://doi.org/10.5555/3722577.3722647>
 30. Hochreiter Sepp, Schmidhuber Jürgen. Long Short-Term Memory // Neural Computation. 1997. Vol. 9, No. 8. P. 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
 31. PCA // Wikipedia.
URL: https://en.wikipedia.org/wiki/Principal_component_analysis (дата обращения: 13.06.2025).
-

DETECTION OF HALLUCINATIONS BASED ON THE INTERNAL STATES OF LARGE LANGUAGE MODELS

T. R. Aisin¹ [0009-0001-5863-3252], **T. V. Shamardina**² [0009-0008-9033-6646]

^{1, 2}*Moscow Institute of Physics and Technology, Dolgoprudny, Moscow Region, Russia*

¹aysin.timur@gmail.ru, ²shamardina.tatiana@gmail.com

Abstract

In recent years, large language models (LLMs) have achieved substantial progress in natural language processing tasks and have become key instruments for addressing a wide range of applied and research problems. However, as their scale and

capabilities grow, the issue of hallucinations — i.e., the generation of false, unreliable, or nonexistent information presented in a credible manner—has become increasingly acute. Consequently, analyzing the nature of hallucinations and developing methods for their detection has acquired both scientific and practical significance.

This study examines the phenomenon of hallucinations in large language models, reviews their existing classification, and investigates potential causes. Using the Flan-T5 model, we analyze differences in the model's internal states when generating hallucinations versus correct responses. Based on these discrepancies, we propose two approaches for hallucination detection: one leveraging attention maps and the other utilizing the model's hidden states. These methods are evaluated on data from HaluEval and Shroom 2024 benchmarks in tasks such as summarization, question answering, paraphrasing, machine translation, and definition generation. Additionally, we assess the transferability of the trained detectors across different hallucination types, in order to evaluate the robustness of the proposed methods.

Keywords: *large language models, hallucinations, detection, Flan-T5, natural language processing, attention maps, hidden states, HaluEval, Shroom.*

REFERENCES

1. Vaswani A., Shazeer N., Parmar N. et al. Attention is all you need // Advances in Neural Information Processing Systems. 2017. Vol. 30.
2. Huang L., Yu W., Ma W. et al. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions // ACM Transactions on Information Systems. 2025. Vol. 43, No. 2. P. 1–55.
<https://doi.org/10.1145/3703155>
3. Li J., Cheng X., Zhao W. X. et al. HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 6449–6464. <https://doi.org/10.18653/v1/2023.emnlp-main.397>
4. Mickus T., Zosa E., Vázquez R. et al. SemEval-2024 Task 6: SHROOM, a Shared-task on Hallucinations and Related Observable Overgeneration Mistakes // International Workshop on Semantic Evaluation. 2024.
<https://doi.org/10.18653/v1/2024.semeval-1.273>

5. *Carlini N., Ippolito D., Jagielski M. et al.* Quantifying Memorization Across Neural Language Models // The Eleventh International Conference on Learning Representations. 2023. <https://doi.org/10.48550/arXiv.2202.07646>
6. *Lin S., Hilton J., Evans Q.* TruthfulQA: Measuring How Models Mimic Human Falsehoods // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022. Vol. 1. P. 3214–3252. <https://doi.org/10.18653/v1/2022.acl-long.229>
7. *Li D., Rawat A.S., Zaheer M. et al.* Large Language Models with Controllable Working Memory // Findings of the Association for Computational Linguistics: ACL 2023. 2023. P. 1774–1793. <https://doi.org/10.18653/v1/2023.findings-acl.112>
8. *Sharma M., Tong M., Korbak T. et al.* Towards Understanding Sycophancy in Language Models // The Twelfth International Conference on Learning Representations. 2024. <https://doi.org/10.48550/arXiv.2310.13548>
9. Reinforcement Learning from Human Feedback: Progress and Challenges // YouTube. URL: https://www.youtube.com/watch?v=hhiLw5Q_UFg (дата обращения: 04.05.2025)
10. *Chuang Y.S., Xie Y., Luo H. et al.* DoLa: Decoding by Contrasting Layers Improves Factuality in Large Language Models // ArXiv. 2023. Vol. abs/2309.03883. <https://doi.org/10.48550/arXiv.2309.03883>
11. *Voita E., Talbot D., Moiseev F. et al.* Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 5797–5808. <https://doi.org/10.18653/v1/P19-1580>
12. *Min S., Krishna K., Lyu X. et al.* FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 12076–12100. <https://doi.org/10.18653/v1/2023.emnlp-main.741>
13. *Luo Z., Xie Q., Ananiadou S.* ChatGPT as a Factual Inconsistency Evaluator for Text Summarization // ArXiv. 2023. Vol. abs/2303.15621. <https://doi.org/10.48550/arXiv.2303.15621>

14. *Manakul P., Liusie A., Gales M.J.* SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models // *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023. P. 9004–9017. <https://doi.org/10.18653/v1/2023.emnlp-main.557>
15. *Cohen R., Hamri M., Geva M. et al.* LM vs LM: Detecting Factual Errors via Cross Examination // *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023. P. 12621–12640. <https://doi.org/10.18653/v1/2023.emnlp-main.778>
16. *Xiao Y., Wang W.Y.* On Hallucination and Predictive Uncertainty in Conditional Language Generation // *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 2021. P. 2734–2744. <https://doi.org/10.18653/v1/2021.eacl-main.236>
17. *Miao N., Teh Y.W., Rainforth T.* SelfCheck: Using LLMs to Zero-Shot Check Their Own Step-by-Step Reasoning // *The Twelfth International Conference on Learning Representations*. 2024. <https://doi.org/10.48550/arXiv.2308.00436>
18. *Adlakha V., BehnamGhader P., Lu X.H. et al.* Evaluating Correctness and Faithfulness of Instruction-Following Models for Question Answering // *Transactions of the Association for Computational Linguistics*. 2024. Vol. 12. P. 681–699. https://doi.org/10.1162/tacl_a_00667
19. *Lin Chin-Yew.* ROUGE: A Package for Automatic Evaluation of Summaries // *Text Summarization Branches Out*. 2004. P. 74–81. ISBN: 9781932432466
20. *Venkit P.N., Gautam S., Panchanadikar R. et al.* Nationality Bias in Text Generation // *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. 2023. P. 116–122. <https://doi.org/10.18653/v1/2023.eacl-main.9>
21. *Goodrich B., Rao V., Liu P.J. et al.* Assessing The Factual Accuracy of Generated Text // *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019. P. 166–175. <https://doi.org/10.1145/3292500.3330955>
22. *Laban P., Schnabel T., Bennett P.N. et al.* SummaC: Re-Visiting NLI-based Models for Inconsistency Detection in Summarization // *Transactions of the Association for Computational Linguistics*. 2022. Vol. 10. P. 163–177.

https://doi.org/10.1162/tacl_a_00453

23. Xu W., Agrawal S., Briakou E. et al. Understanding and Detecting Hallucinations in Neural Machine Translation via Model Introspection // Transactions of the Association for Computational Linguistics. 2023. Vol. 11. P. 546–564.

https://doi.org/10.1162/tacl_a_00563

24. Zhang T., Qiu L., Guo Q. et al. Enhancing Uncertainty-Based Hallucination Detection with Stronger Focus // Conference on Empirical Methods in Natural Language Processing. 2023. <https://doi.org/10.18653/v1/2023.emnlp-main.58>

25. Chuang Y.S., Qiu L., Hsieh C.Y. et al. Lookback Lens: Detecting and Mitigating Contextual Hallucinations in Large Language Models Using Only Attention Maps // Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024. P. 1419–1436.

<https://doi.org/10.18653/v1/2024.emnlp-main.84>

26. Yin Z., Sun Q., Guo Q. et al. Do Large Language Models Know What They Don't Know? // Annual Meeting of the Association for Computational Linguistics. 2023. <https://doi.org/10.18653/v1/2023.findings-acl.551>

27. Marks S., Tegmark M. The Geometry of Truth: Emergent Linear Structure in Large Language Model Representations of True/False Datasets // First Conference on Language Modeling. 2024. <https://doi.org/10.48550/arXiv.2310.06824>

28. Su W., Wang C., Ai Q. et al. Unsupervised Real-Time Hallucination Detection based on the Internal States of Large Language Models // Annual Meeting of the Association for Computational Linguistics. 2024.

<https://doi.org/10.48550/arXiv.2403.06448>

29. Chung H.W., Hou L., Longpre S. et al. Scaling Instruction-Finetuned Language Models // Journal of Machine Learning Research. 2024. Vol. 25, No. 70. P. 1–53. <https://doi.org/10.5555/3722577.3722647>

30. Hochreiter Sepp, Schmidhuber Jürgen. Long Short-Term Memory // Neural Computation. 1997. Vol. 9, No. 8. P. 1735–1780.

<https://doi.org/10.1162/neco.1997.9.8.1735>

31. PCA // Wikipedia.

URL: https://en.wikipedia.org/wiki/Principal_component_analysis (дата обращения: 13.06.2025).

СВЕДЕНИЯ ОБ АВТОРАХ



АЙСИН Тимур Рустемович – инженер машинного обучения и исследователь в области обработки естественного языка. Область научных интересов: оценка языковых моделей, большие языковые модели, интерпретируемость моделей, оптимизация инференса языковых моделей.

Timur Rustemovich AISIN – ML Engineer & NLP researcher. Research interests: language models evaluation, large language models, benchmarking, language models interpretability, inference optimization.

email: aysin.timur@gmail.com

ORCID: 0009-0001-5863-3252



ШАМАРДИНА Татьяна Вячеславовна – исследователь в области обработки естественного языка. Область научных интересов: оценка языковых моделей, большие языковые модели, интерпретируемость моделей. Число научных публикаций – 3.

Tatiana Vyacheslavovna SHAMARDINA – NLP researcher. Research interests: language models evaluation, large language models, benchmarking, language models interpretability. The number of publications – 3.

email: shamardina.tatiana@gmail.com

ORCID: 0009-0008-9033-6646

Материал поступил в редакцию 6 ноября 2025 года

ФОРМИРОВАНИЕ СТРУКТУРИРОВАННЫХ ПРЕДСТАВЛЕНИЙ НАУЧНЫХ ЖУРНАЛОВ ДЛЯ ИНТЕГРАЦИИ В ГРАФ ЗНАНИЙ И СЕМАНТИЧЕСКОГО ПОИСКА

О. М. Атаева¹ [0000-0003-0367-5575], М. Г. Кобук² [0009-0002-9834-8218]

¹Федеральный исследовательский центр «Информатика и управление»
Российской академии наук, г. Москва, Россия

^{1, 2}Московский университет имени С.Ю. Витте, г. Москва, Россия

¹oataeva@frccsc.ru, ²mikhail.kobuk@mail.ru

Аннотация

Работа посвящена проблеме развития библиотеки научных предметных областей SciLibRu, как продолжения семантического описания научных трудов проекта LibMeta. В основе этой библиотеки лежит концептуальная модель данных, структура и семантика которой сформированы на принципах онтологического моделирования. Такой подход обеспечивает строгое описание предметной области, формализацию взаимосвязей между сущностями и возможность дальнейшего автоматизированного анализа данных. Целью настоящего исследования были разработка и экспериментальное применение методов структуризации содержимого научных журналов в формате LaTeX для их интеграции в онтологию библиотеки и обеспечения семантического поиска.

Предложен алгоритм трансляции в формат XML данных, представленных множеством файлов, для интеграции в онтологию библиотеки. Реализован модуль векторного поиска, основанный на вычислении эмбедингов с использованием языковых моделей. Выявлены закономерности распределения эмбедингов и факторы, влияющие на точность ранжирования результатов поиска. Проведено тестирование двух названных компонентов.

Разработанный метод составляет основу для автоматического включения содержимого научных журналов в граф знаний SciLibRu и создания обучающих корпусов для языковых моделей, ограниченных рамками научных предметных областей. Полученные результаты способствуют развитию систем навигации по графу

знаний журналов, а также рекомендательных механизмов и инструментов интеллектуального поиска по русскоязычным научным текстам.

Ключевые слова: полуструктурированные данные, онтология текста, LaTeX, векторное представление текста, полнотекстовый поиск, семантический поиск.

ВВЕДЕНИЕ

Развитие цифровых библиотек нового поколения связано с переходом от классических репозиториев публикаций к семантическим системам, обеспечивающим смысловую интеграцию и навигацию по данным. В рамках этого направления разрабатывается библиотека научных предметных областей SciLibRu. В основе этой библиотеки лежит концептуальная модель данных, структура и семантика которой сформированы на принципах онтологического моделирования. На базе этой онтологии строится граф знаний, обеспечивающий представление семантических связей между объектами предметной области и поддержку интеллектуальной обработки данных. Онтологическое проектирование составляет часть технологии строгого описания предметных областей и взаимосвязей между сущностями, служит для дальнейшей автоматизации процессов анализа данных. В библиотеке SciLibRu предлагается создание унифицированного пространства знаний научных предметных областей на основе научных публикаций, энциклопедий, классификаторов и тезаурусов. Развитие экосистемы SciLibRu требует интеграции разнородных источников, представленные в различных форматах. Анализ разнородных представлений данных и их автоматическое включение в онтологию библиотеки составляет предмет исследования.

В рамках работы рассматриваются прикладные задачи: разработка алгоритма трансляции слабоструктурированных публикаций в формат XML; реализация модуля семантического полнотекстового поиска по полученным данным. Этот модуль предназначен не только для поиска по содержанию, но и для оценки семантических связей между элементами текста. Модуль семантического полнотекстового поиска применяется в дальнейшем для интеграции данных в онтологию SciLibRu с последующим использованием в работе с большими языковыми моделями (Large Language Models, LLM).

В настоящем исследовании рассмотрен процесс структуризации содержимого научных журналов, включающего метаданные журналов и публикаций, а также структурные и смысловые единицы публикаций, такие как разделы, определения, теоремы и т. д. Этот процесс является одним из этапов формирования корпусов данных для библиотечной экосистемы SciLibRu, перехода от документного представления к формату, пригодному для машинного анализа и семантического поиска.

1. ПОСТАНОВКА ЗАДАЧИ

Построение модуля семантического поиска по научным текстам основано на приведении исходных документов к унифицированной форме с сохранением логической и структурной целостности материала. В рамках этой задачи предложен алгоритм структуризации данных и преобразования исходных файлов научных статей в представление, пригодное для последующей векторизации и анализа.

Особенность настоящего исследования заключается в обработке составных документов (журнальных выпусков, сборников статей) без многократного запуска конвертирующих процедур. Перечисленные документы содержат данные в различных форматах и используют авторские варианты TeX-макрокоманд. Предобработка данных учитывает структуру составных документов и таким образом обеспечивает целостную интерпретацию взаимосвязей между файлами и позволяет избежать типичных ограничений существующих инструментов, таких как tex4ht [1] и LaTeXML [2].

1.1. Аналогичные исследования

В области обработки научных текстов в настоящее время преобладают подходы с использованием различных языковых моделей, а классические подходы семантического анализа данных остаются предметом исследования и применяются в специфических задачах, где важны эффективность и работа со структурой языка.

Применительно к семантическому преобразованию неструктурированного LaTeX-текста активная работа сейчас ведется с использованием конвертера tex4ht [1]. Это решение использует не исходный LaTeX-текст, а его промежуточное

представление, формируемое компилятором. При этом упрощается задача анализа текста, так как не требуется работы с авторскими макросами или сложными вложенными структурами документа. Однако в дальнейшем возникает необходимость повторения среды компиляции для каждого файла, что увеличивает процедуру предобработки, особенно при работе с большим количеством разнородных источников.

Второе известное решение, LaTeXML [2], которое поддерживается проектом arXiv и применяется для преобразования тяжелых PDF-версий статей в более легкие HTML. С историей в более чем 20 лет разработки это самое зрелое и полное решение из применяемых в системах накопления текстов. Однако стремление разработчиков сделать LaTeXML универсальным инструментом оказалось одновременно и существенным недостатком: конфигурация оказалась сложна и избыточна для отдельных источников файлов. Обработка любых нестандартных макрокоманд или стилей приводит к необходимости модификации конфигурации этого инструмента. Расширение строгих шаблонов позволяет повысить строгость вывода, однако любое несоответствие, даже не критичное, прерывает процесс преобразования.

Несмотря на длительную историю развития методов накопления оцифрованной информации, остаются востребованными исследования в области алгоритмов предобработки научных текстов с целью их интеграции, семантического представления и использования для LLM.

2. АЛГОРИТМ СТРУКТУРИЗАЦИИ СОСТАВНЫХ ДОКУМЕНТОВ

Разработанный алгоритм структуризации для потоков цифровых архивов журналов состоит из нескольких последовательных этапов. Определяется структура входного набора файлов и формируется дерево включений для многофайловых структур. Главный файл, содержащий ссылки на другие, идентифицируется как корневой элемент, процесс структуризации выполняется для каждого главного файла отдельно. Далее производится регистрация пользовательских макрокоманд, определенных в каждом файле дерева, с возможностью настройки приоритетов по принадлежности и времени обнаружения. Это облегчает корректное разрешение конфликтов при объединении различных областей документа [3].

В итоге формируется абстрактное синтаксическое дерево (AST) [4, 5] с выборочным раскрытием макрокоманд, которое используется для генерации целевого формата базы знаний, строгого XML-представления, предназначенного для последующего анализа и индексирования.

Для оценки корректности работы алгоритма слабой структуризации была проведена серия экспериментов на тестовом множестве из более чем 1000 файлов, представленных в формате LaTeX. В результате 89 файлов не были идентифицированы как формирующие XML-вывод, поэтому не учитывались при анализе.

Оставшиеся 1010 документов были организованы в виде научных статей, объединенных в журнальные выпуски. Полученные результаты показали, что безошибочно конвертирован был 521 файл ($\approx 51.6\%$ от общего количества). Еще 169 файлов ($\approx 16.7\%$ от общего количества) содержали исправимые ошибки, связанные преимущественно с особенностями макрокоманд и вложенной структурой включений.

В 219 случаях ($\approx 21.7\%$) алгоритм завершился с фатальной ошибкой, вызванной нарушением внутренней структуры исходного документа (чаще всего отсутствием точки входа или выхода для конструкций естественного языка, например для описаний теорем и др.). Еще 101 файл ($\approx 10\%$) потребовал доработки модулей лексического анализа и парсера (синтаксического анализатора), отвечающих за обработку примитивов формата LaTeX и выявление зависимостей между макрокомандами.

Анализ выходных XML-файлов показал, что около 15% результатов содержат структурные неточности, а в трети случаев сохранялись остаточные LaTeX-конструкции. Эти дефекты не препятствуют общей интерпретации данных, но требуют уточнения механизма фильтрации и нормализации синтаксических элементов, что и было выполнено на этапе доработки алгоритма.

Доработка алгоритма была направлена на повышение точности и устойчивости парсинга. Основное внимание уделялось расширению покрытия примитивов языка LaTeX и внедрению более гибких механизмов конфигурации, позволяющих управлять процессом раскрытия макрокоманд без усложнения пользовательского интерфейса.

Для повышения надежности было предложено использование стекового контекстного механизма и опережающего просмотра (lookahead). Такой обновленный подход позволяет корректно обрабатывать вложенные конструкции и выявлять ошибки на ранних этапах синтаксического анализа. Улучшения позволили обеспечить сокращение числа ошибок доработки примерно на 6%, а также уменьшить количество некорректных преобразований, связанных с конфигурацией, на 3%. Количество фатальных ошибок снизилось до $\approx 15\%$, что свидетельствует о повышении общей стабильности алгоритма.

Однако оптимизация породила новые трудности. Применение «жадного» алгоритма (greedy algorithm) [5] слияния текстовых фрагментов привело к неравномерному распределению размера тегов `<text>`. Кроме того, в некоторых случаях тег опускается для заголовков и структурных элементов, таких как `<theorem>` или `<lemma>`, что вызывает пропуски текстовых блоков при последующем поиске.

Несмотря на возможность включения полного содержимого каждого тега в результирующий XML, на данном этапе основной интерес представляет текстовая составляющая документа, исключая формулы и другие элементы разметки. Такое строгое ограничение не оптимально, но эффективно для задач полнотекстового и семантического поиска.

3. АРХИТЕКТУРА ПОИСКОВОГО МОДУЛЯ

Следующим этапом исследования было построение полнотекстового семантического поиска по сформированной XML-базе данных. Исходные файлы, полученные в результате трансляции LaTeX-документов, содержат структурированные теги, среди которых основной интерес представляет тег `<text>`. Он включает фрагменты основного текста и может содержать вложенные теги `<formula>`, которые на этом этапе анализа исключались из рассмотрения.

3.1. Построение векторной базы данных

Для каждого выделенного текстового фрагмента формировалось числовое векторное представление – эмбединг, описывающий смысловое содержание текста, при помощи языковых моделей. Это представление отражает семантическую близость между фрагментами. Предполагается, что полученные векторы

должны образовывать хорошо разделенные и интерпретируемые кластеры, в которых тексты, близкие по содержанию, группируются в пределах одного смыслового «облака». На рис. 1 представлен схематически процесс формирования векторных представлений и поиска по ним.

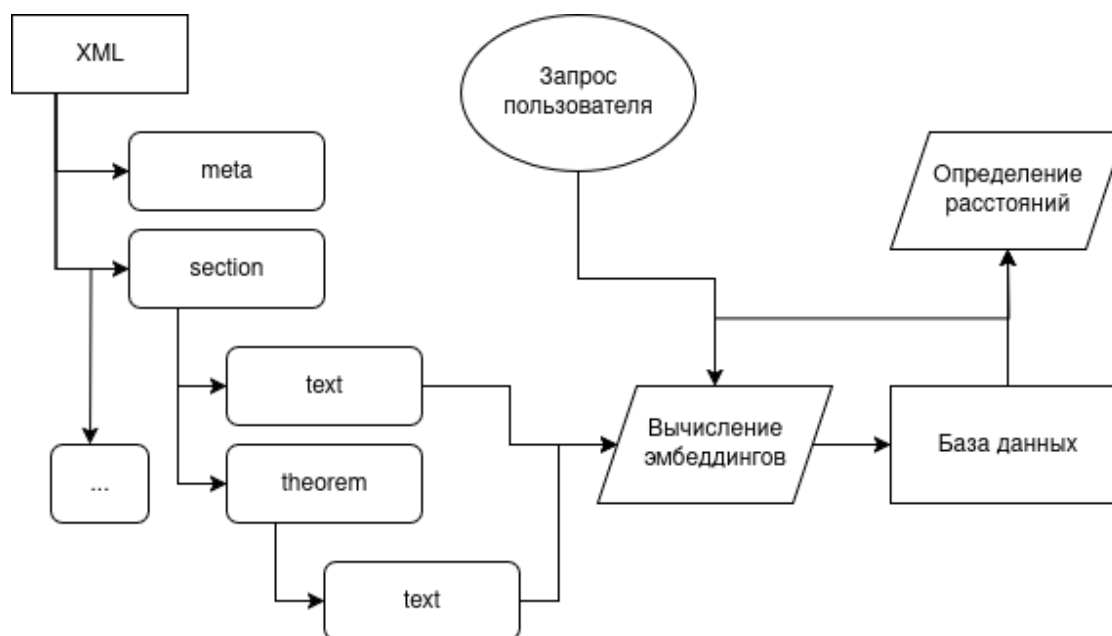


Рис. 1. Схематическое представление процесса формирования векторных представлений и поиска.

Опишем этот процесс на примере рассмотрения одной статьи. Каждый тег `<text>`, обнаруженный в выходном XML-файле, извлекается и передается в процедуру вычисления эмбедингов, которая запускает выбранные методы векторизации, результат записывается в базу данных (БД) вместе с полным текстом фрагмента и метаданными источника. После заполнения БД, пользователь может осуществлять поиск. Для этого поисковый запрос (строка) векторизуется процедурой, которая применялась для вычисления эмбедингов текста и сравнивается с содержимым БД.

Для вычисления эмбедингов были использовано несколько языковых моделей: word2vec [6], GloVe [7], fastText [8], LaBSE [9], ruAdapt [10], ruBERT-base [11], ruBERT-tiny и ruGPT-3-medium [12]. В моделях первого типа (word2vec, GloVe, fastText) векторизация осуществлялась на уровне отдельных слов; итоговый вектор вычислялся как среднее арифметическое векторов всех слов фрагмента. В мо-

делях второго типа (LaBSE, ruAdapt, ruBERT, ruGPT-3) использовалось представление последнего скрытого состояния трансформера, что обеспечивало более сложное кодирование контекста [14, 15].

После получения векторов текстовых фрагментов формировалась векторная база данных. Запрос пользователя, представляющий собой строку произвольной длины, также преобразовывался в вектор (для этого применялась процедура со строго такими же параметрами, как для вычисления эмбеддингов), после чего вычислялось сходство между ним и векторами текстовых фрагментов. В качестве метрики использовалось косинусное расстояние [16–18], которое, несмотря на известные ограничения [2], остается одной из наиболее устойчивых и интерпретируемых мер близости в векторных пространствах.

Для повышения наглядности и удобства восприятия результатов пользователю предоставлялся ранжированный список: тексты сортировались по возрастанию разницы между векторами запроса и найденных фрагментов, что позволяло выделить наиболее релевантные соответствия.

3.2. Результаты оценки моделей поиска

Для первичной оценки качества сформированных эмбеддингов использовалось подмножество из корпуса, включающее 2088 тегов `<text>`, извлеченных из 29 предварительно обработанных XML-файлов. На основе полученных векторов была выполнена двумерная проекция с использованием метода UMAP (Uniform Manifold Approximation and Projection) [19], что позволило визуально оценить распределение фрагментов текста в пространстве признаков (см. рис. 2, 3).

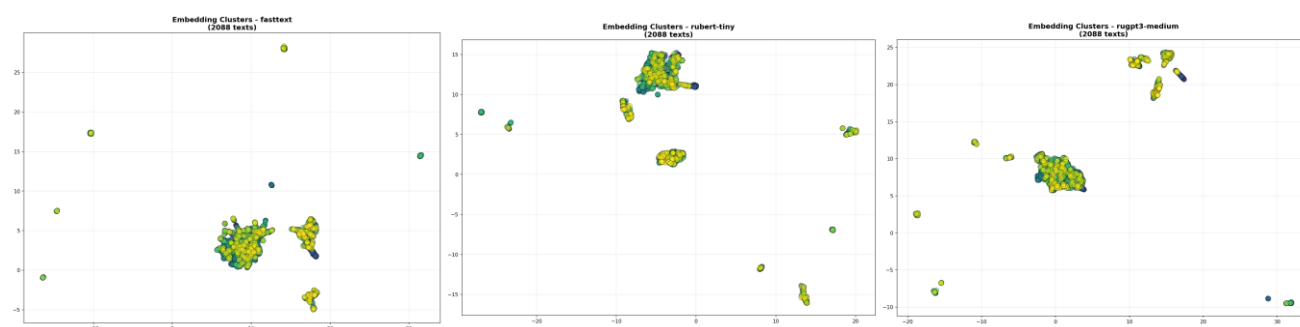


Рис. 2. UMAP-визуализация распределения эмбеддингов для fastText, ruBERT-tiny, ruGPT-3-medium.

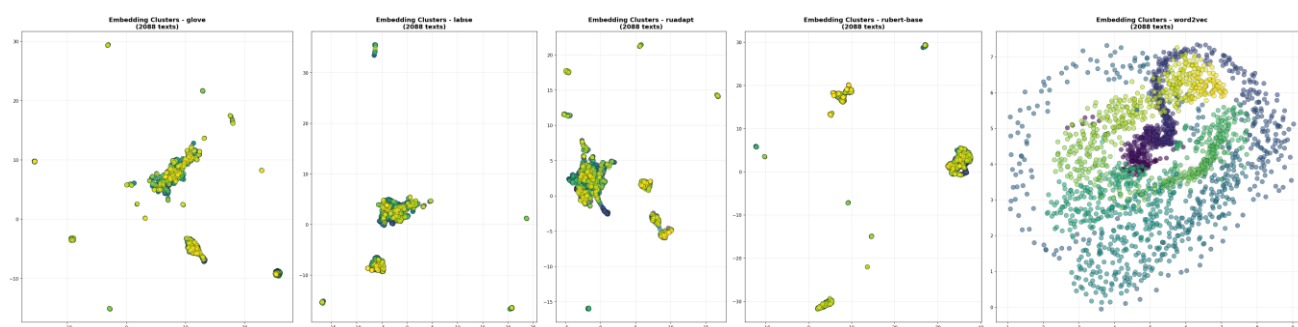


Рис. 3. UMAP-визуализация распределения эмбедингов для GloVe, LaBSE, ruAdapt, ruBERT-base, word2vec.

Анализ показал, что наибольшая кластеризация эмбедингов наблюдается при использовании модели word2vec, тогда как модели ruGPT-3-medium, ruBERT и ruAdapt продемонстрировали более выраженную способность к кластеризации по типам фрагментов, например к выделению аннотаций, подписей к изображениям и таблицам, а также основного научного текста.

Используемая модель word2vec-ruscorpora-300 [13] основана на морфологически аннотированном корпусе и применяет PoS-тегирование (Part of Speech), при котором каждый токен представлен в виде комбинации слова и части речи, например, «вода_NOUN» вместо «вода». Нестандартные токены, не входящие в словарь модели, маркировались как символы, что могло снижать точность представлений.

Качество поиска в целом оказалось ограниченным. Короткие запросы (из одного-двух слов) зачастую приводили к нерелевантным результатам, близким по эффективности к случайным совпадениям. Однако более длинные запросы (4–8 взаимосвязанных слов), особенно при использовании word2vec, позволили достигать смыслового соответствия между запросом и найденными фрагментами. При этом результаты содержали преимущественно длинные отрывки текста, включающие несколько смысловых единиц, что показано в табл. 1 на примере запроса «поверхность шар».

Так, при запросе «теория вероятностей» в число десяти наиболее близких фрагментов входили три последовательных абзаца одной статьи (объединенные как единый результат), где рассматривался метод теории массового обслуживания с многочисленными упоминаниями вероятностных подходов.

Табл. 1. Результаты ранжирования трёх наиболее близких векторов по запросу «поверхность шар» над word2vec

Ранжирование	Результат
1	Preview: «Таким образом, в однодипольной модели обратная задача решается следующим способом. Нужно найти на поверхности сферы две точки с максимальным значением радиальной компоненты магнитного поля»
2	Preview: «т. е. функция ширины экрана непрерывна и в окрестности точки»
3	Preview: «Функция () выпукла вниз по»

3.3. Доработка алгоритма

Результаты первоначального эксперимента показали, что качество поиска заметно зависит от длины текстового фрагмента, используемого при векторизации: проявилась проблема «жадности» алгоритма, упомянутая выше. Несмотря на то что количество относительно длинных фрагментов было невелико (см. рис. 4), длинные участки текста, включающие несколько смысловых единиц, приводят к размыванию семантического центра и, как следствие, к снижению точности сопоставления запросов с релевантными контекстами.

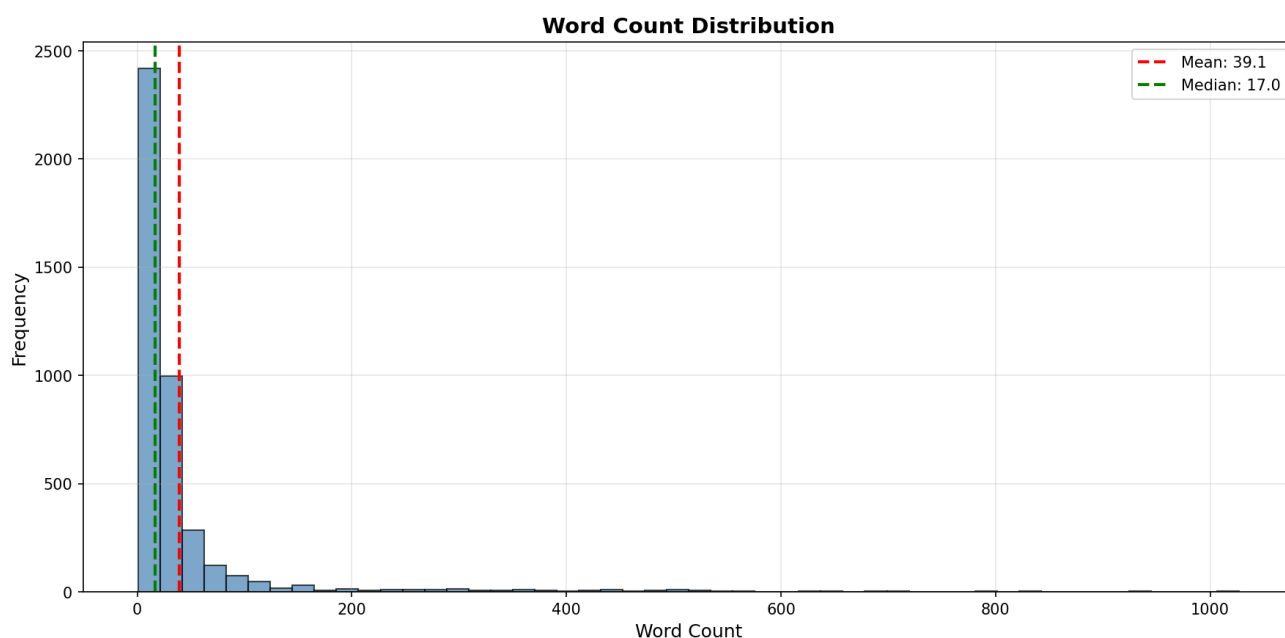


Рис. 4. Распределение количества слов в подмножестве из основного корпуса.

Для устранения этой проблемы была предложена двухступенчатая схема поиска, при которой наряду с усредненным вектором всего фрагмента дополнительно вычисляются векторы для каждого предложения, входящего в его состав.

В рамках эксперимента предложением считался текст, ограниченный знаками препинания (точкой, вопросительным или восклицательным знаком и т. д.). Такой подход позволяет повысить точность ранжирования и локализовать наиболее релевантное предложение в пределах найденного текста, однако все еще подвержен проблеме размывания семантического центра. Схема дополненного алгоритма представлена на рис. 5.

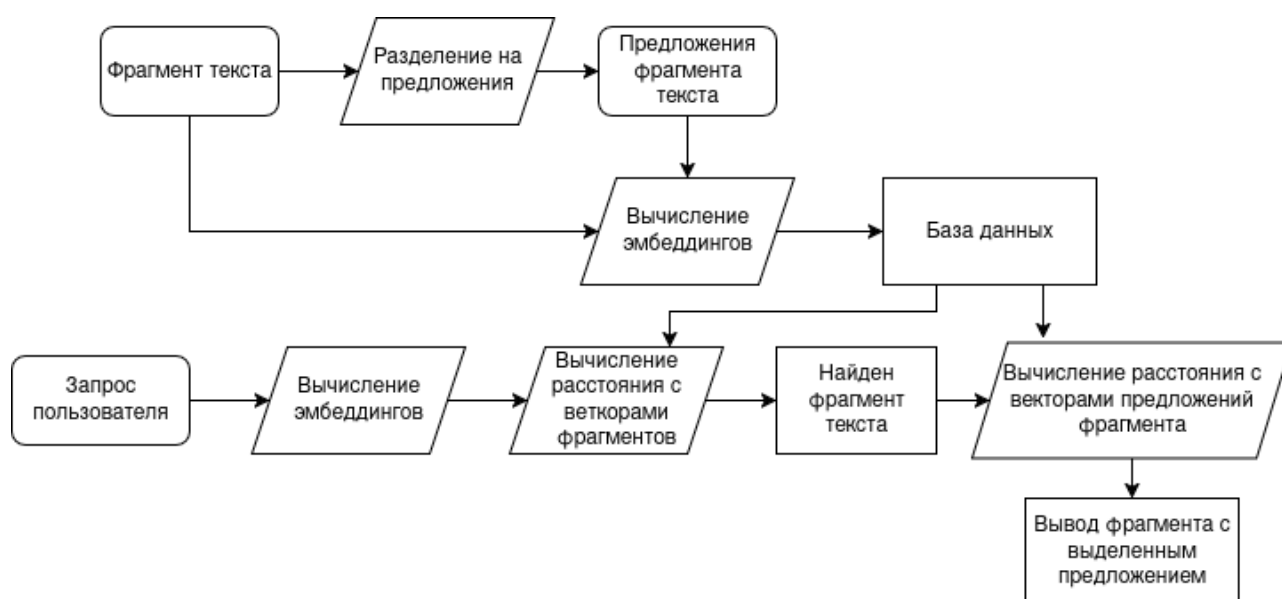


Рис. 5. Схема дополненного алгоритма.

3.4. Результаты анализа алгоритма поиска

Внедрение двухступенчатой системы поиска и ограничения фрагментов до уровня предложения позволило достичь заметного повышения точности ранжирования. В большинстве случаев система корректно выделяла предложение, наиболее близкое по смыслу к поисковому запросу, что обеспечило более интерпретируемые и релевантные результаты. Улучшение качества проявилось прежде всего в способности точнее определять локальные смысловые соответствия. В частности, при поиске по развернутым фразам релевантные тексты стали зани-

мать более высокие позиции в списке результатов, а в пределах найденных документов фокус сместился к предложениям, содержащим ключевые термины запроса.

Примечательно, что в ряде случаев в ранжировании наблюдались совпадения по артефактам, вызванные особенностями корпуса: например, включением отдельных знаков препинания или единичных слов. Подобные элементы могли появляться в результатах поиска из-за высокой частоты встречаемости и статистического сходства коротких векторов.

Качество ранжирования по-прежнему существенно зависит от длины поискового запроса. Более устойчивые результаты достигались при использовании либо частотных терминов корпуса, либо сложных запросов, включающих несколько взаимосвязанных слов. Такая зависимость наиболее выражена для моделей на основе контекстных эмбеддингов, тогда как word2vec демонстрирует большую стабильность при коротких и средних запросах.

Полученные результаты показывают, что использование представлений уровня предложения существенно повышает практическую применимость метода, особенно при работе с корпусами научных текстов, характеризующимися сложной внутренней структурой и разнообразием формулировок.

ЗАКЛЮЧЕНИЕ

Проведенное исследование представляет один из этапов работ в рамках построения библиотеки SciLibRu, направленный на автоматизацию интеграции содержимого научных журналов в ее онтологическую структуру. Разработанный алгоритм трансляции LaTeX-документов в структурированные XML-модели и последующий модуль семантического поиска подтвердили применимость гибридных алгоритмических и нейросетевых подходов для анализа научных текстов.

Полученные результаты, унифицированные наборы структурированных данных, эмбеддинговые представления текстов и прототип поискового механизма создают основу для последующего включения этих данных в граф знаний SciLibRu и их использования при обучении и адаптации языковых моделей LLM к русскоязычным предметным областям.

Развитие описанного направления включает совершенствование механизмов предобработки данных, повышение качества онтологического выравнивания

и разработку рекомендательных систем на основе объединения графа знаний и LLM. Таким образом, работа по структурированию и семантическому поиску журналов является частью комплексной стратегии SciLibRu по созданию интеллектуальной информационной среды для анализа, поиска и извлечения знаний из научных публикаций.

Дальнейшее развитие видится в нескольких направлениях: проведение настройки моделей на специализированных корпусах научных текстов; расширение набора признаков, используемых для ранжирования; а также интеграция механизма автоматического определения типа текстовых фрагментов (аннотация, основное содержание, подпись, теорема, лемма и др.). Такое развитие позволит повысить точность поиска и использовать его в системе навигации по графам знаний научных публикаций, основанных на семантическом описании.

СПИСОК ЛИТЕРАТУРЫ

1. *Hoftich M. TEX4ht: LATEX to Web Publishing // TUGboat. 2019. Vol. 40. No. 1. P. 76–81.*
2. *Frankston C. et al. Using HTML Papers on arXiv: Why It's Important, and How We Made It Happen // arXiv preprint 2024. <https://doi.org/10.48550/arXiv.2402.08954>*
3. *Серебряков В.А., Галочкин М.П., Гончар Д.Р., Фуругян М.Г. Теория и реализация языков программирования. 2-е изд. Москва: Изд-во МЗ-Пресс, 2006. 352 с. ISBN 978-5-4488-1013-8*
4. *Хопкрофт Дж., Мотвани Р., Ульман Дж. Введение в теорию автоматов, языков и вычислений. Москва: Изд-во Вильямс, 2002. 528 с. ISBN: 5-8459-0261-4*
5. *Ахо А.В., Лам М.С., Сети Р., Ульман Дж.Д. Компиляторы: принципы, технологии и инструментарий. 2-е изд. М.: Вильямс, 2008. 1184 с.*
6. *Mikolov T., Sutskever I., Chen K., Corrado G., Dean J. Distributed Representations of Words and Phrases and their Compositionality // Advances in Neural Information Processing Systems (NIPS 26). 2013. P. 3111–3119. <https://dl.acm.org/doi/10.5555/2999792.2999959> (дата обращения: 08.11.2025)*
7. *Pennington J., Socher R., Manning C. GloVe: Global Vectors for Word Representation // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014. P. 1532–1543. <https://doi.org/10.3115/v1/D14-1162>*

8. Joulin A., Grave E., Bojanowski P., Mikolov T. Bag of Tricks for Efficient Text Classification // Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Vol. 2, Short Papers. Valencia, Spain, April 2017. P. 427–431. <https://doi.org/10.18653/v1/E17-2068>

9. Feng F., Yang Y., Cer D., Arivazhagan N., Wang W. Language-agnostic BERT Sentence Embedding // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL). Dublin, Ireland, May 2022. P. 878–891. <https://doi.org/10.18653/v1/2022.acl-long.62>

10. Zmitrovich D. et al. A Family of Pretrained Transformer Language Models for Russian // Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024). Torino, Italia, May 2024. P. 507–524. <https://doi.org/10.48550/arXiv.2309.10931>

11. Kuratov Y., Arkhipov M. Adaptation of Deep Bidirectional Multilingual Transformers for Russian Language // Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference “Dialogue 2019” Moscow, May–June 2019 <https://doi.org/10.48550/arXiv.1905.07213>

12. Nikolich A., Puchkova A. Fine-tuning GPT-3 for Russian Text Summarization // arXiv preprint 2021. <https://doi.org/10.48550/arXiv.2108.03502>

13. Kutuzov A., Kuzmenko E. WebVectors: A Toolkit for Building Web Interfaces for Vector Semantic Models // In: Ignatov D. et al. (eds.) Analysis of Images, Social Networks and Texts (AIST 2016). Communications in Computer and Information Science. Vol. 661. Springer, Cham, 2017. https://doi.org/10.1007/978-3-319-52920-2_15

14. Kasenchak R.T. What is Semantic Search? and Why Is It Important? // Information Services and Use. 2019. Vol. 39. No. 3. P. 205–213. <https://doi.org/10.3233/ISU-190045>

15. Shelke P. et al. A Systematic and Comparative Analysis of Semantic Search Algorithms // International Journal on Recent and Innovation Trends in Computing and Communication. 2023. Vol. 11, No. 11s. P. 222–229. <https://doi.org/10.17762/ijritcc.v11i11s.8094>

16. Weckmüller D., Dunkel A., Burghardt D. Embedding-Based Multilingual Semantic Search for Geo-Textual Data in Urban Studies // Journal of Geovisualization and Spatial Analysis. 2025. Vol. 9. No.31. P. 1-18. <https://doi.org/10.1007/s41651-025-00232-5>

17. *Siddharth Pratap Singh*. Vector Search in the Era of Semantic Understanding: A Comprehensive Review of Applications and Implementations // *International Journal of Computer Engineering and Technology*. 2024. Vol. 15. No. 6. P. 1794–1805. https://doi.org/10.34218/IJCET_15_06_153
 18. *Zhou Y. et al.* Problems with Cosine as a Measure of Embedding Similarity for High Frequency Words // 2022. <https://doi.org/10.48550/arXiv.2205.05092>
 19. *Healy J., McInnes L.* Uniform manifold approximation and projection // *Nature Reviews Methods Primers*. 2024, Vol. 4. No. 82. P. 1–15. <https://doi.org/10.1038/s43586-024-00363-x>
-

FORMATION OF STRUCTURED REPRESENTATIONS OF SCIENTIFIC JOURNALS FOR INTEGRATION INTO A KNOWLEDGE GRAPH AND SEMANTIC SEARCH

O. M. Ataeva¹ [0000-0003-0367-5575], **M. G. Kobuk**² [0009-0002-9834-8218]

¹*Federal Research Center “Computer Science and Control” of the Russian Academy of Sciences, Moscow, Russia*

^{1, 2}*S. Y. Witte University of Moscow, Moscow, Russia*

¹oataeva@frccsc.ru, ²mikhail.kobuk@mail.ru

Abstract

This paper examines the development of the SciLibRu library of scientific subject areas, as a continuation of the semantic description of scientific works from the library LibMeta project. This library is based on a conceptual data model, the structure and semantics of which are formed based on the principles of ontological modeling. This approach ensures a strict description of the subject area, formalization of the relationships between entities, and the possibility of further automated data analysis. The goal of the study is to develop and experimentally apply methods for structuring scientific journal data in LaTeX format for their integration into the library ontology and to support semantic search.

An algorithm for translating data represented by multiple files into XML format is proposed for integration into the library ontology. A vector search module based on

embedding calculation using language models is implemented. Patterns in the distribution of embeddings and factors influencing the accuracy of search results ranking are identified. Testing of the two components is conducted.

The developed method forms the basis for automatically incorporating scientific journal data into the SciLibRu knowledge graph and creating training corpora for language models limited to scientific subject areas. The obtained results contribute to the development of journal knowledge graph navigation systems, recommendation engines, and intelligent search tools for Russian-language scientific texts.

Keywords: *semi-structured data, text structuring, LaTeX, vector representations of text, full-text search, semantic search.*

REFERENCES

1. *Hoftich M. TEX4ht: LATEX to Web Publishing // TUGboat. 2019. Vol. 40, No. 1. P. 76–81.*
2. *Frankston C. et al. Using HTML Papers on arXiv: Why It's Important, and How We Made It Happen // arXiv preprint 2024. <https://doi.org/10.48550/arXiv.2402.08954> (In Russ.)*
3. *Serebryakov V.A., Galochkin M.P., Gonchar D.R., Furugyan M.G. Theory and Implementation of Programming Languages. 2nd ed. Moscow: MZ-Press, 2006. 352 p. (In Russ.)*
4. *Hopcroft J., Motwani R., Ullman J. Introduction to Automata Theory, Languages, and Computation. Moscow: Williams, 2002. 528 p. (In Russ.)*
5. *Aho A.V., Lam M.S., Sethi R., Ullman J.D. Compilers: Principles, Techniques, and Tools. 2nd ed. Moscow: Williams, 2008. 1184 p. (In Russ.)*
6. *Mikolov T., Sutskever I., Chen K., Corrado G., Dean J. Distributed Representations of Words and Phrases and their Compositionality // Advances in Neural Information Processing Systems (NIPS 26). 2013. P. 3111–3119. <https://dl.acm.org/doi/10.5555/2999792.2999959> (date accessed: 08.11.2025)*
7. *Pennington J., Socher R., Manning C. GloVe: Global Vectors for Word Representation // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014. P. 1532–1543. <https://doi.org/10.3115/v1/D14-1162>*

8. *Joulin A., Grave E., Bojanowski P., Mikolov T.* Bag of Tricks for Efficient Text Classification // Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. Valencia, Spain, April 2017. P. 427–431. <https://doi.org/10.18653/v1/E17-2068>
9. *Feng F., Yang Y., Cer D., Arivazhagan N., Wang W.* Language-agnostic BERT Sentence Embedding // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL). Dublin, Ireland, May 2022. P. 878–891. <https://doi.org/10.18653/v1/2022.acl-long.62>
10. *Zmitrovich D. et al.* A Family of Pretrained Transformer Language Models for Russian // arXiv preprint 2023. <https://doi.org/10.48550/arXiv.2309.10931>
11. *Kuraton Y., Arkhipov M.* Adaptation of Deep Bidirectional Multilingual Transformers for Russian Language // arXiv preprint 2019. <https://doi.org/10.48550/arXiv.1905.07213>
12. *Nikolich A., Puchkova A.* Fine-tuning GPT-3 for Russian Text Summarization // arXiv preprint 2021. <https://doi.org/10.48550/arXiv.2108.03502>
13. *Kutuzov A., Kuzmenko E.* WebVectors: A Toolkit for Building Web Interfaces for Vector Semantic Models // In: Ignatov D. et al. (Eds.) Analysis of Images, Social Networks and Texts (AIST 2016). Communications in Computer and Information Science. Vol. 661. Springer, Cham, 2017. https://doi.org/10.1007/978-3-319-52920-2_15
14. *Kasenchak R.T.* What is Semantic Search? and Why Is It Important? // Information Services and Use. 2019. Vol. 39. No. 3. P. 205–213. <https://doi.org/10.3233/ISU-190045>
15. *Shelke P. et al.* A Systematic and Comparative Analysis of Semantic Search Algorithms // International Journal on Recent and Innovation Trends in Computing and Communication. 2023. Vol. 11, No. 11s. P. 222–229. <https://doi.org/10.17762/ijritcc.v11i11s.8094>
16. *Weckmüller D., Dunkel A., Burghardt D.* Embedding-Based Multilingual Semantic Search for Geo-Textual Data in Urban Studies // Journal of Geovisualization and Spatial Analysis. 2025. Vol. 9. No. 31. P. 1–18. <https://doi.org/10.1007/s41651-025-00232-5>
17. *Siddharth Pratap Singh.* Vector Search in the Era of Semantic Understanding: A Comprehensive Review of Applications and Implementations // International

Journal of Computer Engineering and Technology. 2024. Vol. 15. No. 6. P. 1794–1805.
https://doi.org/10.34218/IJCET_15_06_153

18. Zhou Y. et al. Problems with Cosine as a Measure of Embedding Similarity for High Frequency Words // 2022. <https://doi.org/10.48550/arXiv.2205.05092>

19. Healy J., McInnes L. Uniform manifold approximation and projection // Nature Reviews Methods Primers. 2024, Vol. 4. No. 82. P. 1–15.
<https://doi.org/10.1038/s43586-024-00363-x>

СВЕДЕНИЯ ОБ АВТОРАХ



АТАЕВА Ольга Муратовна – старший научный сотрудник Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН, к. т. н. Область научных интересов: системное программирование, базы данных, инженерия знаний и онтологии.

Olga Muratovna ATAeva – Senior Researcher, Dorodnicyn Computing Center, Federal Research Center “Computer Science and Control” of the Russian Academy of Sciences; Candidate of Technical Sciences (PhD). Research interests: systems programming, databases, knowledge engineering, ontologies. Number of scientific publications – 80.

email: oataeva@frccsc.ru

ORCID: 0000-0003-0367-5575



КОБУК Михаил Геннадьевич – студент бакалавриата Московского университета имени С.Ю. Витте, область интересов NLP, анализ данных, системное программирование.

Mikhail Gennadievich KOBUK – bachelor at S. Witte University of Moscow. Research interests: NLP, data analysis, system programming.

email: mikhail.kobuk@mail.ru

ORCID: 0009-0002-9834-8218

Материал поступил в редакцию 8 ноября 2025 года

БИБЛИОТЕКА НАУЧНЫХ ПРЕДМЕТНЫХ ОБЛАСТЕЙ SCILIBRU

О. М. Атаева¹ [0000-0003-0367-5575], Н. П. Тучкова² [0000-0001-5357-9640],
К. Б. Теймуразов³ [0009-0000-0876-4222], А. Абдышов⁴ [0009-0000-4971-0378],
М. Г. Кобук⁵ [0009-0002-9834-8218]

^{1–3}Федеральный исследовательский центр «Информатика и управление»
Российской академии наук, г. Москва, Россия

^{1, 4, 5}Московский университет имени С.Ю. Витте, г. Москва, Россия

¹OAtaeva@frccsc.ru, ²NTuchkova@frccsc.ru, ³KTeymurazov@frccsc.ru,

⁴abdysovajdin@gmail.com, ⁵mikhail.kobuk@mail.ru

Аннотация

Работа посвящена проблеме интеграции данных для представления научных предметных областей на основе их семантического описания в цифровой библиотеке SciLibRu. В качестве модели данных использованы онтология и граф знаний библиотеки LibMeta. Наполнение библиотеки SciLibRu осуществляется путем добавления данных научных журналов. Показано, как реализованы этапы анализа слабоструктурированных научных публикаций для их встраивания в онтологию библиотеки. При прохождении всех этапов предобработки данных формируется датасет, который может быть использован в обучении языковых моделей для запросов в русскоязычных научных предметных областях.

Приложение работы заключается в создании рекомендательных систем для работы с научными русскоязычными журналами.

Ключевые слова: прикладная онтология, граф знаний, источники данных, анализ слабоструктурированных научных публикаций.

ВВЕДЕНИЕ

Библиотека предметных областей SciLibRu сформирована в рамках подхода, который использовался в семантической библиотеке LibMeta [1], на основе онтологического проектирования [2] и навигации по данным с помощью графа

знаний (knowledge graph, KG) [3]. LibMeta – это цифровая библиотека, данные которой связаны иерархическими и ассоциативными отношениями в соответствии с *онтологией предметной области*.

Онтология цифровой семантической библиотеки представляет собой формальное описание множества данных (типы данных, связи) предметной области (subject domain, SjD) [4]. Для построения онтологии был использован язык OWL (https://www.w3.org/2006/04/OWL_UseCases-ru.html).

Онтология на OWL представлена в виде RDF-графа (<https://www.w3.org/TR/rdf12-schema/>). Описание на OWL содержит *структуры* данных, а не *самих данных*. Каждый экземпляр данных – это экземпляр элемента онтологии. В библиотеке LibMeta тезаурус [5] представлен онтологией на OWL.

Процесс редактирования библиотеки LibMeta заключается в редактировании *онтологии*. В результате интеграции данных в библиотеке LibMeta накоплены описания SjD математики и смежных областей, включая описания классификаторов, научных журналов и других источников, которые семантически связаны в онтологии библиотеки SciLibRu. Развитие навигации в библиотеке с применением KG позволяет перейти к поиску, задав в поисковом запросе принадлежность данных SjD.

В настоящей работе предложено описание коллекции данных SciLibRu, которые отличаются от данных LibMeta наличием новых связей, полученных в результате метрического анализа [6, 7] KG LibMeta и достраивания онтологии, хотя эти множества, конечно, пересекаются. Описание данных SciLibRu стало необходимым исследованием, поскольку дальнейшее развитие библиотеки предполагает постановку задач о применении больших языковых моделей (Large Language Models, LLM) и создании рекомендаций для извлечения знаний, накопленных в этой библиотеке. Множество данных LLM, предварительно обученной на общих данных, предполагается дополнять данными KG SciLibRu, чтобы ограничивать LLM рамками предметной области, задаваемой KG.

В работе представлены описание семантической модели данных и KG, оценка качества данных для интегрированных в библиотеку энциклопедий, пример интеграции LLM и KG для SjD обыкновенных дифференциальных уравнений

(ordinary differential equation, ODE), описана процедура гибридного алгоритма загрузки слабоструктурированных данных журналов, содержащих символьную информацию (формулы).

1. СЕМАНТИЧЕСКАЯ МОДЕЛЬ И ДАННЫЕ SCILIBRU

1.1. Семантическая модель SciLibRu

Будем использовать понятия *тезауруса* предметной области [8, 9], *онтологии* [4, 7], *KG (графа знаний)* [10, 11], *знаний* [12, 13]. Данные SciLibRu состоят из множества научных статей, энциклопедий, классификаторов, словарей, тезаурусов, корпусов текстов и других источников оцифрованной информации, представленного в виде онтологии SjD. *Онтология* SciLibRu реализует семантическую модель данных, в основе которой лежит тезаурус предметной области. Онтология проецируется на KG, поскольку реализуется через схему RDF. Основным контент SciLibRu составляют математические SjD, приложения и смежные области из прикладных и междисциплинарных журналов.

Знания трактуются как структурированные *данные*. Извлечение знаний состоит в получении ответа на информационный запрос. Ответ может быть (не)релевантным теме запроса и (не)пертинентным, то есть (не)удовлетворяющим информационную потребность пользователя, (не)соответствующим тезаурусу адресата [8].

Семантическая модель данных представляет собой структурированные данные, где есть объект, субъект и отношения между ними. Именно отношения составляют смысл этой тройки данных. В KG отношения – это связи графа, а объекты и субъекты – вершины графа.

Онтология LibMeta [1] – трехуровневая, содержит:

- универсальные *понятия* онтологии;
- описания *объектов* прикладной области;
- *метаданные прикладной области* как таковые.

Основными элементами онтологии на OWL являются описания классов, их свойств, отношений между классами и представителями (индивидов) классов (их свойств и отношений). Для этих описаний OWL последовательно использует бинарные отношения своего словаря, а также словарей RDF и RDFS (<https://www.w3.org/TR/owl-semantics/>).

SjD LibMeta, $DLib = \{Per, Pub, Th, Enc, Jour\}$:

- данные персон и их свойств, множество $\{Per\}$;
- данные публикаций и их свойств, множество $\{Pub\}$;
- данные тезаурусов, множество $\{Th\}$;
- данные энциклопедий, множество $\{Enc\}$;
- данные журналов, множество $\{Jour\}$.

Данные *SjD SciLibRu DSci* содержат множество *DLib* и данные множества $\{KG\}$, то есть $DSci = \{Per, Pub, Th, Enc, Jour, KG\}$.

На рис. 1 показано, что к трехуровневой модели онтологии *LibMeta* добавился уровень данных *KG*.

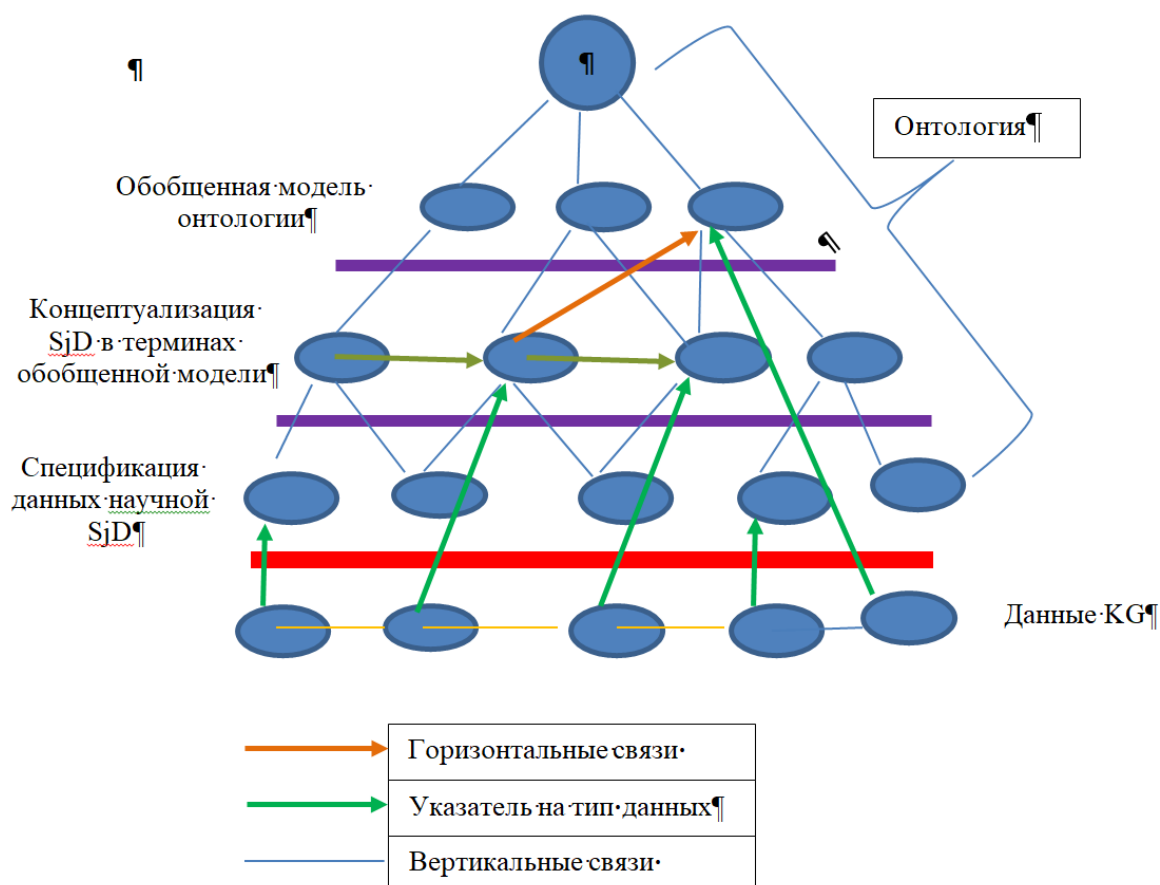


Рис. 1. Архитектура данных в *SciLibRu*.

1.2. Анализ качества KG *SciLibRu*

Оценка качества данных стала одним из этапов представления знаний, поскольку внедрение LLM «перекладывает» вопросы достоверности с экспертов научных областей на создателей информационных систем, которые организуют интеграцию данных и навигацию по ним. Известные недостатки применения LLM в поисковых запросах, которые приводят к «галлюцинациям», связаны как с логикой модели, так и с массивом данных (датасет), на которых проводилось обучение языковой модели [14].

Для библиотеки SciLibRu оценка данных заключается в проверке связей в онтологии и, соответственно, в KG, как в результирующем множестве концептов и связей, который будет использоваться на входе LLM [15].

Для нашего исследования были вычислены метрики KG SciLibRu для вершин классификаторов УДК (<https://teacode.com/online/udc/>) и ГРНТИ (<https://grnti.ru/>) и получены значения метрик для оценки качества графа.

Общее количество узлов, число вершин (узлов) в графе: $|V| = 10813$.

Общее количество связей, число ребер (связей) в графе: $|E| = 178728$.

Распределение узлов KG SciLibRu по типам:

тип 1 – KG математической энциклопедии [16], $|V|_1 = 6263$,

тип 2 – KG энциклопедии математической физики [17], $|V|_2 = 3228$.

Распределение узлов KG SciLibRu по связям показывает, сколько ассоциативных связей (ребер) каждого типа в KG:

$|E|_1 = 155448$ (связь *related*), $|E|_2 = 11805$ (связь *use*), $|E|_3 = 11475$ (связь *see also*).

Плотность – мера *заполненности* графа связями: $\rho = \frac{|E|}{|V|(|V|-1)}$.

Для направленного графа: $|V|(|V|-1)$ – это возможное максимальное количество ребер (каждый узел может иметь ребро к каждому, кроме себя). Плотность графа KG SciLibRu: $\rho = 0.0015$.

Низкое значение плотности графа указывает на его *разряженность*, что типично для графов большого размера (граф энциклопедий) и объясняется тем, что большинство узлов связано только с небольшим подмножеством других узлов преимущественно в рамках тематических взаимосвязей каждого понятия.

В графе KG SciLibRu выявлено 1057 изолированных узлов типа *Concept*, что составляет около 10% от общего количества понятий SJD в графе. Это указы-

вает на отсутствие связей этих понятий с другими сущностями, что может свидетельствовать о недостаточности доступной информации в данной подобласти. Далее были вычислены метрики центральности.

Самыми влиятельными узлами оказались наиболее общие понятия, что говорит о высокой взаимосвязанности и значимости этих понятий в охватываемых SJD. Анализ центральности подтвердил значимость этих понятий, при этом были выявлены дополнительные узлы, которые, несмотря на меньшую распространенность, обладают высокой вероятностной значимостью в графе. Кроме того, в результате анализа были обнаружены узлы, замыкающиеся на себя (пример эго-графа «Функция», рис. 2), то есть их (концепты и связи) необходимо дополнительно анализировать. В поисковой выдаче, вероятно, это приведет к дублированию.

Качественный анализ графа показал, что есть связи, которые не были выявлены в процессе предобработки данных, что тоже, очевидно, влияет на результат поиска. Была установлена корреляция между эмбедингами и классическими метриками.

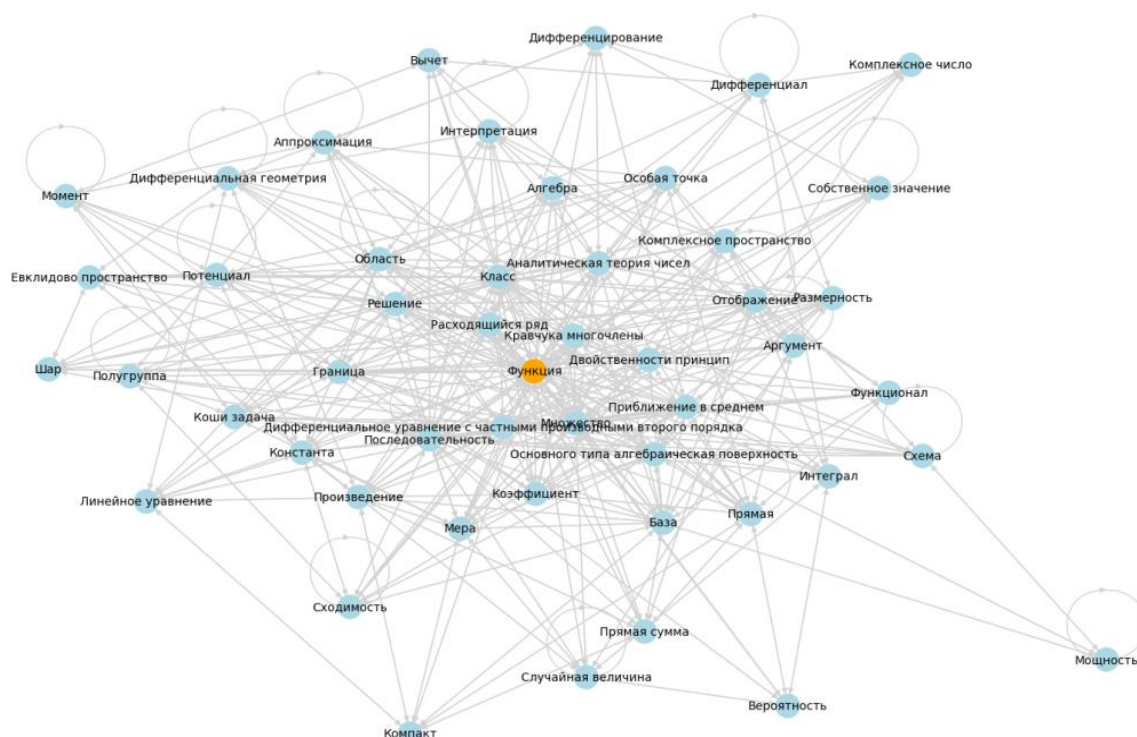


Рис. 2. Эко-граф узла «Функция» (подграф KG SciLibRu).

Качественный анализ привел к достраиванию графа и онтологии библиотеки.

1.3. Навигация по данным SciLibRu

Навигация по данным SciLibRu реализована с помощью KG. Вершины KG (статьи, термины, формулы SjD) – это экземпляры элементов онтологии, связи – это связи тезауруса SjD. Для описания онтологии использованы следующие формальные определения.

Объекты и ресурсы:

$OB = \{O_1, \dots, O_n\}$, where $O_i, i=1, \dots, n$, – множество информационных объектов;

$R = \{r_1, \dots, r_m\}$, где $r_j, j=1, \dots, m$, – множество информационных ресурсов (типы информационных объектов);

$TYPE(O) = r$ – отношение, где каждому информационному объекту O_i соответствует информационный ресурс r_j ;

$IsRe(r_1, r_2)$ – отношение иерархии ресурсов, где r_2 ($r_2 \geq 0, r_2 \subset r_1$) составляет часть ресурса r_1 (подресурс).

Атрибуты:

$A = \{a_1, \dots, a_k\}$ – множество атрибутов информационных ресурса, $Z(a_i)$ – значение атрибута a_i .

Источники модели данных:

$SX = \{(r_x, Ar)\}$, – множество пар, где r_x и Ar – ресурсы и их атрибуты,

$G(r_x, Ar) = (r, A)$, G – отношение и функция сопоставления элементов исходной модели данных с информационными ресурсами информационной системы и их набором атрибутов.

Данные источников: $X = (SX, G)$.

Предметная область тезауруса Th : $Th = (P, T, R)$, где P – множество концептов, T – множество вербальных терминов концептов, R – множество вертикальных и горизонтальных связей терминов и концептов.

F – функция сопоставления каждого информационного объекта из OB с соответствующими терминами из T : $F(o_i) = \{t\}, t \in T, o_i \in OB$.

Тезаурус Th определен для онтологии семантической библиотеки на языке OWL.

KG определяется на основе RDF-схемы как триплет (s, p, o) .

Каждая тройка представляет собой упорядоченный набор терминов RDF:

субъект $s \in U \cup B$, предикат $p \in U$ и объект $o \in U \cup B \cup L$.

Кроме того, термин RDF – это URI, $u \in U$, «пустой» узел $b \in B$ или литерал $l \in L$.
Значит, KG лучше всего представляется на основе схем RDF, но не каждое представление данных RDF рассматривается как KG .

Основные связи, которые реализованы между элементами KG :

- объект \leftrightarrow объект;
- концепт \leftrightarrow концепт;
- объект \leftrightarrow концепт \leftrightarrow объект;
- концепт \leftrightarrow объект \leftrightarrow концепт;
- классификатор \leftrightarrow концепт \leftrightarrow классификатор;
- концепт \leftrightarrow классификатор \leftrightarrow концепт;
- объект \leftrightarrow классификатор \leftrightarrow объект;
- классификатор \leftrightarrow объект \leftrightarrow классификатор.

1.4. Применение KG SciLibRu

Множество данных в виде KG имеет большое значение для создания рекомендательных систем с применением LLM для коммуникации. Одну из актуальных проблем LLM можно сформулировать как *ограничение знаний, на которых формируется ответ LLM при обеспечении полноты данных описания SjD*. Вариантом решения этой задачи может быть использование KG SjD в качестве входного множества данных для LLM, ранее предобученной на общих данных.

В нашем исследовании были проведены эксперименты [18] применения LLM к работе с SjD ODE в библиотеке SciLibRu. Разработана методология взаимодействия LLM и KG SjD ODE на основе инструкций, применяемых к описанию SjD в виде KG . На примере SjD ODE мы ограничиваем знания модели KG SjD, не позволяя ей выходить за границы SjD ODE. На выходе получаем ответы на естественном языке, опираясь на знания, представленные в библиотеке в виде KG .

В процессе экспериментов были выявлены следующие ошибки, которые влияют на качество данных в библиотеке SciLibRu.

1. Синтаксические ошибки в запросах. Это ошибки в запросах, генерируемых моделью LLM. Например, лишние «слэши» в предикатах, отсутствие префиксов. Для их исправления добавлена инструкция *промт*.

2. Синтаксические ошибки в онтологии. Они вызывают у модели LLM сложности «в понимании», если классы и свойства некорректно именованы или аннотированы. Для их исправления в онтологию добавлены `rdfs:label` и `rdfs:comment` и учтены в дальнейшем при доработке модели данных.

С учетом этих ошибок были сформулированы инструкции к LLM с использованием модели KG SJD ODE для формирования SPARQL-запросов по запросам пользователя на естественном языке [18, 19].

2. ОБНОВЛЕНИЕ ДАННЫХ И НАПОЛНЕНИЕ БИБЛИОТЕКИ

Обновление данных и дальнейшее наполнение библиотеки, достраивание онтологии и графа знаний реализуются при *интеграции научных статей*, которые проходят предобработку для загрузки в библиотеку. В процессе предобработки данных был разработан *гибридный подход* для интеграции слабоструктурированных данных, содержащих символьную информацию (формулы в LaTeX). Были выявлены проблемы при реализации подхода, которые решались программным путем, созданием скриптов на Python.

Этап предобработки исходных данных журнальных статей для наполнения библиотеки SciLibRu потребовал анализа как самих данных, так и методов конвертации из формата LaTeX в стандартизированный формат XML, соответствующий заданным спецификациям.

2.1. Первичный анализ слабоструктурированных данных

Архивы выпусков выбранных периодических научных изданий хранятся в формате «ГГГГ-Н.rar» (где ГГГГ – год, Н – номер), где обязательно присутствует главный исходный файл – IPI.TEX. С помощью файла IPI.TEX выполняется формирование выпуска, при этом редакциями используются существующие шаблоны, обозначения и авторская стилистика применения LaTeX-макрокоманд.

Для выбора стратегии предобработки разнородных составных архивов были проанализированы некоторые подходы и методы, доступные при заданных (ограниченных по времени и мощности) вычислительных ресурсах.

В качестве стратегии для предобработки рассматривались:

А. Синтаксический анализ (парсинг, parsing) – построение формальной грамматики и использование парсер-генераторов для создания анализатора исходного текста. Этот подход требует тщательной проработки грамматики и сложен в реализации из-за гибкости и расширяемости LaTeX.

Б. Переопределение макрокоманд – модификация стандартных и пользовательских команд LaTeX для генерации XML-разметки вместо типографского вывода. Этот метод позволяет сохранить семантику документа, но требует глубокого понимания системы макросов LaTeX.

В. Использование промежуточных форматов – конвертация через промежуточные форматы (DVI, PDF) с последующим извлечением структуры и содержания. При этом подходе может теряться семантическая информация, но он проще в реализации.

Г. Гибридные решения – комбинация различных подходов, например, использование парсера для базовой структуры документа и переопределение макрокоманд для специфических конструкций.

Д. Использование нейросетевых моделей – для трансформации исходного текста LaTeX в XML. Этот подход позволяет достичь высокой точности конвертации, но требует значительных вычислительных ресурсов и большого объема данных для обучения модели.

Надо отметить, что самый качественный и универсальный подход состоит в использовании нейросетевых моделей, но высокие требования к вычислительным ресурсам и объему данных для обучения модели усложняют его применение в данном случае.

Использование промежуточных форматов позволяет частично упростить обработку, однако возникают проблемы при работе с устаревшими форматами документов. Кроме того, возникают накладные расходы на хранение и обработку промежуточных форматов.

Рассмотрены также методы и инструменты преобразования документов по извлечению структурированной информации [20, 21] из неструктурированных или полуструктурированных текстов, что имеет прямую аналогию с задачей преобразования LaTeX-разметки в семантически обогащенный XML.

Е. Изучены универсальные конвертеры и программатические подходы (programmatic approach). Проведены исследования по созданию *кастомных парсеров* для LaTeX с использованием регулярных выражений, синтаксических анализаторов (например, ANTLR) или специализированных библиотек для работы с текстовыми данными. Эти исследования позволили оценить сложность разработки собственного решения и его преимущества в плане точности и контроля выходного формата.

В результате проведенного анализа методов (А-Е) сделаны выводы, что наиболее простым и надежным подходом представляется *гибридный: комбинация парсинга для базовой структуры документа и переопределения макрок команд для специфических конструкций и пользовательских команд*.

2.2. Разработка архитектуры преобразователя исходного документа и построение соответствующего XML-документа

Разработка системы конвертации LaTeX в XML реализована путем создания лексера (лексического анализатора) и парсера (синтаксического анализатора), авторского гибридного «компилятора» [22–24].

Сначала был проведен эксперимент: была произведена тестовая сборка файлов с помощью существующих систем сборки LaTeX. Далее, несмотря на наличие PDF-файлов, собранных в архиве, было принято решение провести тестовую сборку для проверки возможности использования решения, основанного на генерации промежуточного файла.

2.2.1. Проблемы кодировки

В ходе сборки были выявлены проблемы с кодировкой. Кодировка файлов произведена в CP866 (также известна как IBM866), разработана в паре с основной кодировкой (с которой совпадает по набору символов) в середине 1980-х годов в Вычислительном центре Академии наук СССР [25]. Эта кодировка пользовалась большой популярностью среди советских пользователей IBM PC-совместимых ПК.

Все кириллические символы архивов оказались нечитаемыми в исходном виде, в связи с чем появилась задача перекодировки файлов в кодировку UTF-8, как основную систему кодировки документов для подавляющего большинства повседневных задач для текстов в LaTeX. Для решения проблемы преобразования

файлов был написан скрипт на языке Python, изменяющий кодировку с сохранением исходного файла с расширением «.backup». В ходе работы выяснилось, что существующие механизмы обнаружения кодировок (разные версии uchardet) не определяют CP866 корректно (во всех случаях кодировка определена как турецкий подтип ISO). Это связано с тем, что CP866 – единственная или одна из немногих IBM-образных таблиц, получивших достаточно широкое распространение и при этом не вошедших в спецификацию ISO.

2.2.2. Проблемы специфических библиотек

Некоторые библиотеки (в частности, «acad.sty») не содержатся в основных репозиториях и пакетах поставки LaTeX-систем на Linux-дистрибутивы, а CTAN (архив Comprehensive TeX Archive Network) более не считает пакет активным. В результате эксперимента с тестовой сборкой файлов было принято решение разработки собственного лексера-парсера (компилятора) и последующего траверса AST (Abstract Syntax Tree). Были выбраны существующие генераторы грамматик (ANTLR4 + Python). Разделение на лексер и парсер не осуществлялось, единая грамматика была сформирована с целью упрощения организации правил и управления конфигурацией.

2.3. Реализация прототипа конвертера, удовлетворяющего заявленным требованиям

Для работы был использован пример формата XML, конвертацию в который требовалось осуществить. С целью облегчения и ускорения отладки были составлены *модельные примеры корректных входного и выходного файлов*.

В процессе анализа исходных файлов для построения модельного файла был также решен ряд проблем с пользовательскими макрокомандами и грамматиками.

2.3.1. Исследование пользовательских макрокоманд

Все (или почти все) файлы с «полезным содержимым», то есть декларативным LaTeX-кодом, содержат определения пользовательских макрокоманд. Кроме того, нередко встречается переопределение макрокоманд, при этом не всегда с сохранением смысла. В самом файле IPI.TEX количество и содержание переопределений также нестабильны и изменяются от выпуска к выпуску. Веро-

ятнее всего, это связано с различной тематикой выпусков (например, определение D для быстрого начертания *дисперсии* для выпуска, посвященного теории вероятностей).

Названная проблема существенно осложняет разработку компиляторного решения и в очередной раз указывает на удобство нативного (оптимизированного для конкретного случая) решения (использования существующих систем сборки для генерации промежуточного или конечного файла и последующей обработки такового).

Для устранения несогласованности пользовательских макрокоманд был написан Python-скрипт, который в два прохода по файлам производит раскрытие пользовательских макросов (запись – раскрытие: 2 прохода). При этом пользовательские макросы (которые определены в файлах статей) имеют приоритет над глобальными. Такое действие относится к предобработке данных и выполняется до запуска непосредственного решения, аналогично с программой смены кодировки.

2.3.2. Результат исследований пользовательских макрокоманд

Написанный прототип успешно конвертировал модельный исходный файл в модельный целевой файл, опираясь на очевидные семантические маркеры (например, форматированная строка «Доказательство» для отделения доказательства присутствовала во всех файлах в том или ином виде.)

Грамматика модельного файла занимала 107 строк без учета сопроводительных пометок. В ходе реализации грамматики для «рабочего», а не модельного файла, объем грамматики быстро вырос до ~400 строк, а ANTLR-генерированные Python-программы стали нестабильными. Появились ошибки, например разбор текстового сегмента, как правило, после перевода на новую строку (символ абзаца в LaTeX – «\», при этом LaTeX-правила начинаются с «\»).

2.3.3. ANTLR4 и сложные грамматики

В ходе работы над практической грамматикой установлено, что фреймворк ANTLR4 представляет собой мощный и расширяемый генератор парсеров по заданным правилам. Однако конфигурация ANTLR4 строго формализована и не допускает ручного управления и временного отхода от строгих правил реализации

грамматических правил обработки языка, что осложняет разработку практических лексеров и парсеров. В частности, при реализации вручную рекурсивного спуска можно реализовать «наивный lookahead» путем поиска закрывающего тэга с углублением рекурсии при обнаружении повторного открывающего. С ANTLR4, как и любым другим генератором, это невозможно.

Несмотря на отличную работу ANTLR4-генератора на модельных файлах, попытка применить его на практике столкнулась с проблемой *конфликтующих правил*, а также нетривиальной приоритетной очередности составленных выражений. Когда грамматика для практического файла достигла ~400 строк и ее отладка стала проблематичной, стало очевидно, что использование функционала грамматических генераторов необходимо пересмотреть.

Использование ANTL4 сопряжено с необходимостью считаться с особенностями подхода.

2.4. Оценка эффективности предложенного подхода путем экспериментального тестирования и сравнения результатов с существующими аналогичными инструментами

В исследовании был реализован траверс существующего дерева для формирования AST с применением лексера и парсера LaTeX для более корректного и качественного разбора текста. Были рассмотрены основные популярные приложения: pdftex, xetex, luatex, bibtex, miktex.

В ходе анализа выяснилось, что как такового построения AST они не производят. Основными недостатками оказались:

- неспособность использовать семантические маркеры (не-LaTeX команды) в качестве управляющих символов для правил без вмешательства в систему разбора исходных файлов;
- высокий порог вхождения для корректной настройки шаблонов и собственных макросов;
- исходный файл должен быть представлен в виде отдельного документа, не имеющего зависимостей.

После экспериментов построения конфигурации была реализована рабочая система раскрытия большинства макросов методом двойного прохода по файлу. Идея заключается в следующем: объединить все статьи в единый выходной

LaTeX-файл, в котором раскрыть все макрокоманды с учетом приоритета и области видимости. Это позволило избежать сложностей в виде разрешения зависимостей и раскрытия пользовательских макрокоманд.

Реализация алгоритма состояла из трех этапов: раскрытие макросов, разрешения подключений зависимостей и объединение текста.

ЗАКЛЮЧЕНИЕ

В результате исследования была проведена оценка качества данных библиотеки SciLibRu, выявлены направления для их совершенствования как на этапе загрузки (предобработки данных с помощью гибридного подхода), так и на этапе достраивания онтологии и KG. Было проведено обширное исследование методов и подходов предобработки сложно структурированных файлов и выработан подход для их конвертации и интеграции в семантическую библиотеку. Получен набор структурированных данных, который может быть использован для интеграции LLM с KG SciLibRu для коммуникации на естественном языке.

Дальнейшие исследования направлены на улучшение качества данных SciLibRu с целью формирования датасета для работы LLM с русскоязычными научными текстами.

Благодарности

Работа выполнена в рамках выполнения темы НИР «Математические методы анализа данных и прогнозирования» ФИЦ ИУ РАН.

СПИСОК ЛИТЕРАТУРЫ

1. *Серебряков В.А., Атаева О.М.* Информационная модель открытой персональной семантической библиотеки LibMeta // Научный сервис в сети Интернет: труды XVIII Всероссийской научной конференции (19–24 сентября 2016 г., г. Новороссийск). М.: ИПМ им. М.В. Келдыша, 2016. С. 304–313.
URL: <http://keldysh.ru/abrau/2016/3.pdf>.
2. *Rospocher M., Tonelli S., Serafini L. and Pianta E.* Corpus-based terminological evaluation of ontologies // Applied Ontology. 2012. Vol. 7, No. 4. P. 429–448.
<https://doi.org/10.3233/AO-2012-0114>
3. *Ataeva O., Serebryakov V., Tuchkova N.* Ontological approach to a knowledge graph construction in a semantic library // Lobachevskii J. of Mathematics. 2023. Vol. 44, No. 6. P. 2229–2239. <https://doi.org/10.1134/S1995080223060471>

4. Handbook on Ontologies. Editors: Steffen Staab, Rudi Studer, Springer-Verlag Berlin Heidelberg, 2004. <https://doi.org/10.1007/978-3-540-24750-0>
 5. Атаева О.М., Серебряков В.А., Тучкова Н.П. Подходы к организации математических знаний при формирования предметных тезаурусов различных разделов математики // CEUR Workshop Proceedings. 2018. Vol. 2260. P. 42–54. <https://doi.org/10.20948/abrau-2018-66>
 6. Hlomani H., Stacey D., Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey // Semantic Web Journal. 2014. Vol. 1, No. 5. P. 1–11. <https://www.semantic-web-journal.net/system/files/swj657.pdf>
 7. Lozano-Tello A. and Gómez-Pérez A. Ontometric: A method to choose the appropriate ontology // Journal of Database Management. 2004. Vol. 15, No. 2. P. 1–18. <https://doi.org/10.4018/jdm.2004040101>
 8. Шрейдер Ю.А. Тезаурусы в информатике и теоретической семантике // Научно-техническая информация. Сер. 2. 1971. № 3. С. 21–24.
 9. Лукашевич Н.В. Тезаурусы в задачах информационного поиска. М.: Изд-во МГУ, 2011. 495 с.
 10. Харари Ф. Теория графов. Пер. с англ. и предисл. В.П. Козырева. Под ред. Г.П. Гаврилова. Изд. 2-е. М.: Едиториал УРСС, 2003. 296 с.
 11. Barrasa J., Webber J. Building Knowledge Graphs: A Practitioner's Guide. O'Reilly. 2023. 290 p.
 12. Biswas G., Bezdek J., Oakman R. L. A knowledge-based approach to online document retrieval system design // In Proc. ACM SIGART Int. Symp. Methodol. Intell. Syst. (ISMIS '86) 1986. P. 112–120. <https://doi.org/10.1145/12808.12821>
 13. Гаврилова Т.А., Кудрявцев Д.В., Муромцев Д.И. Инженерия знаний. Модели и методы: Учебник. СПб.: Изд-во «Лань», 2016. 324 с.
 14. Pan S. et al. Unifying Large Language Models and Knowledge Graphs: A Roadmap // in *IEEE Transactions on Knowledge and Data Engineering*. 2024. Vol. 36, No. 7. P. 3580–3599. <https://doi.org/10.1109/TKDE.2024.3352100>
 15. Luo L. et al. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models // arXiv preprint arXiv:2410.13080. 2024. <https://doi.org/10.48550/arXiv.2410.13080>
 16. Виноградов И.М. (Гл. ред.). Математическая энциклопедия (в 5 томах)/ М.: Советская энциклопедия (1977–1985).
-

17. *Фаддеев Л.Д.* (Гл. ред.). Энциклопедия математической физики. Энциклопедия. М.: Большая русская энциклопедия, 1998. 692 с.

18. *Ataeva O.M., Tuchkova N.P.* Adaptation of the language model for mathematical texts in the semantic library // *System Informatics (Системная информатика)*. 2025. No. 27. P. 59–75.

19. *Будзко В.И., Атаева О.М., Тучкова Н.П.* Автоматизация доступа к информации при навигации по данным семантической библиотеки и интеграции графа знаний с языковой моделью // *Системы высокой доступности*. 2025. Т. 21. № 2. С. 5–11. <https://doi.org/10.18127/j20729472-202502-0>

20. *Клюкин А.А., Широков А.А.* Автоматизированная система подготовки слабоструктурированной информации [Электронный ресурс] // *Гаудеамус*. 2014. №2 (24).

URL: <https://cyberleninka.ru/article/n/avtomatizirovannaya-sistema-podgotovki-slabostrukturirovannoy-informatsii> (дата обращения: 01.11.2025).

21. *Куртюкин С.В.* Метод автоматизированного формирования сборников архивных документов [Электронный ресурс] // *Теория и практика современной науки*. 2018. №5 (35).

URL: <https://cyberleninka.ru/article/n/metod-avtomatizirovannogo-formirovaniya-sbornikov-arhivnyh-dokumentov> (дата обращения: 01.11.2025).

22. *Ахо А., Сети Р., Ульман Дж.* Компиляторы: принципы, технологии, инструменты. М.: Вильямс, 2001, 762 с.

23. *Волкова И.А., Вылиток А.А., Руденко Т.В.* Формальные грамматики и языки. Элементы теории трансляции: учебное пособие для студентов II курса М.: Изд-во Моск. гос. ун-та, 2009.

24. *Гладкий А.В.* Формальные грамматики и языки. М.: Наука, Гл. ред. физ.-мат. лит., 1973, 368 с.

25. *Брябрин В.М., Ландау И.Я., Неменман М.Е.* О системе кодирования для персональных ЭВМ // *Микропроцессорные средства и системы*. 1986. № 4. С. 61–64.

Scilibru, The Library Of Scientific Subject Domains

O. M. Ataeva¹ [0000-0003-0367-5575], N. P. Tuchkova² [0000-0001-5357-9640],
K. B. Teymurazov³ [0009-0000-0876-4222], A. Abdyshev⁴ [0009-0000-4971-0378],
M. G. Kobuk⁵ [0009-0002-9834-8218]

^{1, 2, 3}*Federal Research Center "Informatics and Control" of the Russian Academy of Sciences, Moscow, Russia*

^{1, 4, 5}*S. Y. Witte Moscow University, Moscow, Russia*

¹OAtaeva@frccsc.ru, ²NTuchkova@frccsc.ru, ³KTeymurazov@frccsc.ru,
⁴abdysovajdin@gmail.com, ⁵mgkobuk@gmail.com

Abstract

The work is devoted to the problem of data integration for representing scientific subject areas based on their semantic description in the SciLibRu digital library. The LibMeta library's ontology and knowledge graph are used as the data model. SciLibRu is populated by adding data from scientific journals. The paper demonstrates how the stages of processing semi-structured scientific publications for their integration into the library's ontology are implemented. Completing all data preprocessing stages yields a dataset that can be used to train language models for queries in Russian-language scientific subject areas.

Keywords: *applied ontology, knowledge graph, data sources, analysis of semi-structured scientific publications.*

REFERENCES

1. Serebryakov V.A., Ataeva O.M. Informacionnaya model' otkrytoj personal'noj semanticheskoy biblioteki LibMeta // Nauchnyj servis v seti Internet: trudy XVIII Vserossijskoj nauchnoj konferencii (19–24 sentyabrya 2016 g., g. Novorossiysk). M.: IPM im. M.V. Keldysha, 2016. S. 304–313. URL: <http://keldysh.ru/abrau/2016/3.pdf> (In Russ.)
2. Rospocher M., Tonelli S., Serafini L., Pianta E. Corpus-based terminological evaluation of ontologies // Applied Ontology. 2012. Vol. 7, No. 4. P. 429–448. <https://doi.org/10.3233/AO-2012-0114>

3. *Ataeva O., Serebryakov V., Tuchkova N.* Ontological approach to a knowledge graph construction in a semantic library // *Lobachevskii J. of Mathematics*. 2023. Vol. 44, No. 6. P. 2229–2239. <https://doi.org/10.1134/S1995080223060471>
4. *Handbook on Ontologies*. Editors: Steffen Staab, Rudi Studer, Springer-Verlag Berlin Heidelberg, 2004. <https://doi.org/10.1007/978-3-540-24750-0>
5. *Ataeva O., Serebryakov V., Tuchkova N.* Podhody k organizacii matematicheskikh znaniy pri formirovaniya predmetnyh tezaurusov razlichnyh razdelov matematiki // *CEUR Workshop Proceedings*. 2018. Vol. 2260. P. 42–54. <https://doi.org/10.20948/abrau-2018-66> (In Russ.)
6. *Hlomani H., Stacey D.* Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey // *Semantic Web Journal*. 2014. Vol. 1, No. 5. P. 1–11. <https://www.semantic-web-journal.net/system/files/swj657.pdf>
7. *Lozano-Tello A., Gómez-Pérez A.* Ontometric: A method to choose the appropriate ontology // *Journal of Database Management*. 2004. Vol. 15, No. 2. P. 1–18. <https://doi.org/10.4018/jdm.2004040101>
8. *Shrejder Yu.A.* Tezaurusy v informatike i teoreticheskoy semantike // *Nauchno-tekhnicheskaya informaciya*. Ser. 2. 1971. № 3. S. 21–24 (In Russ.).
9. *Lukashevich N.V.* Tezaurusy v zadachah informacionnogo poiska. M.: Izd-vo MGU, 2011. 495 s. (In Russ.).
10. *Harari F.* Teoriya grafov. Per. s angl. i predisl. V.P. Kozyreva. Pod red. G.P. Gavrilova. Izd. 2-e. M.: Editorial URSS, 2003. 296 s
11. *Barrasa J., Webber J.* Building Knowledge Graphs: A Practitioner's Guide. O'Reilly. 2023. 290 p.
12. *Biswas G., Bezdek J., Oakman R.L.* A knowledge-based approach to online document retrieval system design // *In Proc. ACM SIGART Int. Symp. Methodol. Intell. Syst. (ISMIS '86)*, 1986. P. 112–120. <https://doi.org/10.1145/12808.12821>
13. *Gavrilova T.A., Kudryavcev D.V., Muromcev D.I.* Inzheneriya znaniy. Modeli i metody: Uchebnik. SPb.: Izdatel'stvo «Lan'», 2016. 324 s. (In Russ.).
14. *Pan S. et al.* Unifying Large Language Models and Knowledge Graphs: A Roadmap // in *IEEE Transactions on Knowledge and Data Engineering*. 2024. Vol. 36, No. 7. P. 3580–3599. <https://doi.org/10.1109/TKDE.2024.3352100>
15. *Luo L. et al.* Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models // *arXiv preprint arXiv:2410.13080*. 2024.

<https://doi.org/10.48550/arXiv.2410.13080>

16. *Vinogradov I.M.* (Gl. red.). *Matematicheskaya enciklopediya* (v 5 tomah) M.: Sovetskaya enciklopediya (1977–1985) (In Russ.).

17. *Faddeev L.D.* (Gl. red.). *Enciklopediya matematicheskoy fiziki*. Enciklopediya. M.: Bol'shaya russkaya enciklopediya. 1998. 692 s. (In Russ.).

18. *Ataeva O.M., Tuchkova N.P.* Adaptation of the language model for mathematical texts in the semantic library // *System Informatics*. 2025. No. 27. P. 59–75.

19. *Budzko V.I., Ataeva O.M., Tuchkova N.P.* Access automation to information for navigating through semantic library data and integrating the knowledge graph with the language model // *Highly Available Systems*. 2025. V. 21. No. 2. P. 5–11.

<https://doi.org/10.18127/j20729472-202502-0>. (In Russ.).

20. *Klyukin A.A., Shirokov A.A.* Avtomatizirovannaya sistema podgotovki slabostrukturirovannoy informacii. [Elektronnyj resurs] // *Gaudeamus*. 2014. Vol. 24, No. 2. URL: <https://cyberleninka.ru/article/n/avtomatizirovannaya-sistema-podgotovki-slabostrukturirovannoy-informatsii> (date of access: 01.11.2025) (In Russ.).

21. *Kurtyukin S.V.* Metod avtomatizirovannogo formirovaniya sbornikov arhivnyh dokumentov [Elektronnyj resurs] // *Teoriya i praktika sovremennoj nauki*. 2018. №5 (35). URL: <https://cyberleninka.ru/article/n/metod-avtomatizirovannogo-formirovaniya-sbornikov-arhivnyh-dokumentov> (data obrashcheniya: 01.11.2025).

22. *Aho A., Seti R., Ul'man Dzh.* Kompilyatory: principy, tekhnologii, instrument. M.: Vil'yams, 2001, 762 s. (In Russ.).

23. *Volkova I.A., Vylitok A.A., Rudenko T.V.* Formal'nye grammatiki i yazyki. Elementy teorii translyacii : uchebnoe posobie dlya studentov II kursa M.: Izd-vo Mosk. gos. un-ta, 2009. (In Russ.).

24. *Gladkij A.V.* Formal'nye grammatiki i yazyki. M.: Nauka, Gl. red. fiz.-mat. lit., 1973, 368 s. (In Russ.).

25. *Bryabrin V.M., Landau I.Ya., Nemenman M.E.* O sisteme kodirovaniya dlya personal'nyh EVM // *Mikroprocessornye sredstva i sistemy*. 1986. № 4. S. 61–64 (In Russ.).

СВЕДЕНИЯ ОБ АВТОРАХ



АТАЕВА Ольга Муратовна – старший научный сотрудник Федерального исследовательского центра «Информатика и управление» Российской академии наук (ФИЦ ИУ РАН), кандидат техн. наук, специалист в области системного программирования и баз данных.

Olga Muratovna ATAeva – senior researcher at FRC SCS RAS, PhD, expert in the field of system programming and databases.

email: OAtaeva@frccsc.ru

ORCID: 0000-0003-0367-5575



ТУЧКОВА Наталия Павловна – старший научный сотрудник ФИЦ ИУ РАН, кандидат физ.-мат. наук. Специалист в области алгоритмических языков и информационных технологий.

Natalia Pavlovna TUCHKOVA – senior researcher at FRC SCS RAS, PhD in physics with a math degree. The expert in the field of algorithmic languages and information technologies.

email: NTuchkova@frccsc.ru

ORCID: 0000-0001-5357-9640



ТЕЙМУРАЗОВ Кирилл Борисович – ведущий программист ФИЦ ИУ РАН, специалист в области информационных технологий и баз данных.

Kirill Borisovich TEYMURAZOV – lead programmer at FRC SCS RAS, expert in the field of computer sciences and databases.

email: KTeymurazov@frccsc.ru

ORCID: 0009-0000-0876-4222



АБДЫШОВ Айдин – студент бакалавриата Московского университета имени С.Ю. Витте, область интересов: анализ данных и визуализация.

Aidin ABDYSHOV – undergraduate student at the S. Y. Witte Moscow University, area of interest: data analysis and visualization.

email: abdysovajdin@gmail.com

ORCID: 0009-0000-4971-0378



КОБУК Михаил Геннадьевич – студент бакалавриата Московского Университет имени С.Ю. Витте, область интересов NLP, анализ данных, системное программирование.

Mikhail Gennadievich KOBUK – undergraduate student at the S. Y. Witte Moscow University, area of interest: NLP, data analysis, systems programming.

email: mikhail.kobuk@mail.ru

ORCID: 0009-0002-9834-8218

Материал поступил в редакцию 10 ноября 2025 года

ИНСТРУМЕНТ ДЛЯ ОПЕРАТИВНОЙ ДИАГНОСТИКИ ПАМЯТИ НЕЙРОСЕТЕВЫХ АРХИТЕКТУР ЯЗЫКОВЫХ МОДЕЛЕЙ

П. А. Гавриков¹ [0009-0009-6175-2883], А. К. Усманов² [0009-0006-2535-1052],
Д. Реваев³ [0009-0008-1630-1912], С. Н. Бузыканов⁴ [0009-0005-3405-0705]

¹⁻⁴Московский физико-технический институт, г. Долгопрудный,
Московская обл., Россия

¹gavrikov.pa@phystech.edu, ²usmanov.ak@phystech.edu, ³revaev.d@phystech.edu,
⁴bsn1977@mail.ru

Аннотация

Большие языковые модели (Large Language Models, LLM) прошли путь от простых N-граммных систем до современных универсальных архитектур, однако ключевым ограничением остается квадратичная сложность механизма самовнимания по длине входной последовательности. Это существенно увеличивает потребление памяти и вычислительных ресурсов, а с появлением задач, требующих рекордно длинных контекстов, создает необходимость разработки новых архитектурных решений. Поскольку для исследования предлагаемой архитектуры требуется длительное и дорогостоящее обучение полновесной сети, необходимо разработать инструмент, который позволял бы быстро дать предварительную оценку архитектуре с точки зрения внутренней памяти.

В настоящей работе предложен метод количественной оценки внутренней памяти нейросетевых архитектур на основе синтетических тестов, не требующих больших корпусов данных. Под внутренней памятью понимается объем информации, который модель способна воспроизвести без обращения к исходным входам.

Для верификации подхода разработан программный комплекс, апробированный на архитектурах GPT-2 и Mamba. Используются задачи копирования, инверсии и извлечения значения по ключу. Проведенное сравнение по точности предсказаний, распределению ошибок и вычислительным

затратам позволяет оперативно оценивать эффективность и перспективность архитектур LLM.

Ключевые слова: *большие языковые модели, архитектура нейросетей, внутренняя память, долговременное хранение информации, обработка последовательностей, измерение функциональной памяти, сравнение архитектур.*

ВВЕДЕНИЕ

В последние годы искусственный интеллект достиг значительных успехов в области обработки естественного языка. Особенно примечательным стало развитие больших языковых моделей (Large Language Models, LLM), способных не только продолжать текст, но и выполнять сложные когнитивные задачи: от ответа на вопросы и перевода до программирования и логического рассуждения. Эти модели, построенные на нейросетевых архитектурах, стали основой для создания новых поколений интеллектуальных систем, способных взаимодействовать с человеком на естественном языке. Их широкое распространение стало возможным благодаря доступности больших объемов текстовых данных, увеличению вычислительных мощностей и совершенствованию алгоритмических решений в глубоком обучении [1].

Одним из ключевых факторов, определяющих эффективность языковых моделей, является длина контекста, с которым модель способна работать. Под контекстом в данном случае понимается последовательность исходных текстовых единиц (токенов), из которых модель «понимает» смысл сообщения, определяет стиль ответа и извлекает релевантную информацию. Чем длиннее контекст, тем больше информации может быть учтено при генерации следующего выходного элемента. Однако с увеличением длины входа возрастают и вычислительные затраты, что накладывает ограничения на используемые архитектурные решения. В частности, в большинстве современных моделей на практике ограничение длины контекста составляет несколько тысяч текстовых единиц [2]. Это становится серьезным препятствием при решении задач, связанных с анализом длинных документов, поддержанием диалога на

протяжении нескольких обменов репликами или извлечением информации из больших массивов текста [3].

Задача работы с длинными контекстами приобрела актуальность в последние годы в результате совокупного влияния технологических и прикладных факторов. Ранние языковые модели, такие как GPT-2 [4], имели сравнительно небольшие окна контекста (512–1024 токена) и обучались на фрагментах текстов соответствующего размера. Основной задачей было научиться предсказывать следующую единицу текста и улучшить качество генерации. Многочисленные улучшения достигались за счет увеличения числа параметров и размера обучающих выборок. Длина окна внимания при этом оставалась второстепенной характеристикой, поскольку типичные задачи (продолжение текста, классификация, перевод) не требовали обработки больших объемов входной информации.

Ситуация изменилась в 2022–2023 г., когда появились новые классы задач, требующих долгосрочного хранения и использования информации. Среди них – многошаговые диалоги в чат-ботах, анализ длинных юридических и технических документов, понимание и редактирование исходного кода программ, а также цепочки логических рассуждений. Эти задачи подразумевают, что модель должна оперировать не десятками, а тысячами и десятками тысяч единиц текста за один раз [5]. Окна контекста длиной 2048 или 4096 перестали быть достаточными.

На фоне этих требований возникла техническая проблема: архитектура трансформера [6], лежащая в основе большинства современных языковых моделей, в существующем виде оказывается плохо масштабируемой при увеличении длины контекста. Это накладывает серьезные ограничения на использование модели в задачах, требующих анализа длинных входных последовательностей [1]. В результате возникает необходимость в оптимизации существующих архитектур или разработке новых подходов, способных эффективно справляться с большими объемами входных данных.

Один из наиболее очевидных и формально корректных способов оценки памяти модели – это полноценное обучение на больших объемах реальных данных (например корпусах новостных или разговорных текстов) и последующий анализ поведения модели при варьировании длины входного контекста. Такой

подход, несмотря на свою корректность, имеет целый ряд серьезных недостатков. Обучение современных языковых моделей даже среднего размера требует сотен GPU-часов и масштабной инфраструктуры [1]. В условиях разработки новых архитектур, когда нужно протестировать множество гипотез и модификаций,

такой подход становится крайне неэффективным: стоимость одной неудачной архитектурной итерации становится неоправданно высокой. Кроме того, из-за большого числа параметров (слои, головы (heads), размеры внутренних представлений и т. д.) обучение каждой новой конфигурации становится дорогим по времени и требует длительного инженерного цикла [7]. Таким образом, метод «обучения и тестирования на полном корпусе» тормозит исследовательскую активность и ограничивает возможности итеративной разработки.

Поскольку на сегодняшний день отсутствует инструмент, который бы позволял не только быстро оценить способность модели к запоминанию и воспроизведению длинных контекстов данных, но и не требовал бы при этом огромных корпусов натуральных данных, возникает необходимость создания инструмента для быстрой оценки перспективных архитектур с точки зрения памяти.

ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

Существует несколько классов методов, так или иначе позволяющих оценить архитектуру языковой модели на разных этапах ее жизненного цикла: от предварительного анализа архитектурных решений до профилирования при постобучении. Ниже приведен обзор наиболее репрезентативных подходов и средств к обучению, а также анализ их применимости к задаче оценки памяти и способности модели к работе с длинным контекстом.

Одним из наиболее компактных и при этом широко обсуждаемых датасетов в сообществе разработчиков языковых моделей является TinyStories – синтетический корпус, представленный в 2023 г. исследователями из Microsoft Research [8]. Датасет был создан в ответ на потребность в специализированных тренировочных данных для обучения малых языковых моделей объемом от 10 до 100 млн параметров. В отличие от масштабных корпусов, таких как The Pile [9] или

Common Crawl [10], TinyStories ориентирован на минимализм, простоту синтаксиса и ограниченность лексического разнообразия, что делает его пригодным для быстрого обучения и интерпретируемого анализа поведения моделей.

Однако в контексте оценки способности языковой модели к обработке длинных контекстов датасет TinyStories оказывается ограниченно применимым. Поскольку длина каждой истории составляет в среднем менее 300 текстовых единиц, модель, обучающаяся на этом корпусе, практически не сталкивается с необходимостью оперировать удаленными по позиции фрагментами текста. Отсутствие длинных зависимостей и компактная структура входа означают, что модель не развивает способности к долгосрочному контекстному запоминанию и ее механизмы внимания не подвергаются реальному стресс-тесту. Соответственно, TinyStories не позволяет протестировать или сравнить архитектуры по признакам, важным для задач с удлинённым контекстом: устойчивости представлений при увеличении длины входа, способности сохранять релевантную информацию на больших расстояниях, точности адресации при ассоциативном поиске и др.

Одним из наиболее масштабных и широко используемых источников текстовых данных для обучения больших языковых моделей является датасет The Pile [9], разработанный исследовательским коллективом EleutherAI в 2020 г. Этот корпус представляет собой специально составленный набор текстов объемом порядка 825 гигабайт, предназначенный для обучения авторегрессионных языковых моделей.

Объем и разнообразие данных делают The Pile одним из наиболее полных и репрезентативных наборов текстов в открытом доступе. Однако его применение сопряжено с рядом технических ограничений. В первую очередь, обучение на полном корпусе требует значительных вычислительных ресурсов [1]. Даже при использовании оптимизированных программных решений и современных графических ускорителей обучение модели среднего масштаба может занимать несколько недель. Это делает невозможным широкое применение набора The Pile в условиях быстрой прототипизации или исследования множества архитектурных гипотез. Другой проблемой корпуса

является неоднородность качества исходных данных. Некоторые подкорпуса, например, GitHub или Common Crawl (Pile-CC), включают в себя тексты со значительным уровнем шума или нерелевантных фрагментов.

При всей своей ценности как обучающего корпуса, The Pile не подходит для решения задач быстрой оценки архитектурных особенностей моделей. Его масштаб и связанная вычислительная нагрузка делают набор неприменимым в сценариях, где требуется быстрая обратная связь, например при тестировании памяти модели или устойчивости к длинным зависимостям. В подобных случаях более уместны компактные, синтетические задачи, позволяющие получить интерпретируемые показатели в кратчайшие сроки и без необходимости полного обучения.

FlashAttention – это специализированный алгоритм эффективного вычисления механизма внимания, предложенный Трием Фонгом с соавторами в 2022 г. [11]. Его ключевая цель – устранение избыточных обращений к памяти при вычислении операций внимания (attention) в трансформерах. Это достигается путем переработки порядка выполнения операций и управления памятью таким образом, чтобы данные не записывались и не считывались из памяти GPU больше одного раза. Алгоритм FlashAttention позволяет добиться ускорения в 2–4 раза по сравнению с классической реализацией внимания softmax, особенно на больших длинах последовательностей.

Основное нововведение метода заключается в использовании блочного (tile-based) подхода к вычислению матриц внимания, при котором входная последовательность делится на небольшие блоки. Каждая операция выполняется в пределах локального хранилища, что позволяет держать промежуточные значения в регистровой памяти или кэшах высокого уровня. Это существенно сокращает пропускную способность ввода-вывода и минимизирует объем трафика между GPU DRAM и ядрами.

С практической точки зрения алгоритм FlashAttention особенно полезен при работе с длинными контекстами, где классическая реализация механизма внимания становится узким местом как по времени, так и по памяти. В задачах обработки длинных текстов (1000–64000 токенов и более) FlashAttention обеспечивает как ускорение, так и снижение потребления видеопамати. Однако

важно понимать, что FlashAttention не является архитектурным улучшением, а представляет собой инженерную оптимизацию одного из компонентов стандартного трансформера. Поэтому его использование не расширяет функциональные возможности модели, а лишь делает существующие механизмы внимания быстрее и дешевле в использовании. Более того, он сохраняет квадратичную сложность по длине контекста, хотя и с меньшим коэффициентом. Это означает, что даже с использованием FlashAttention трансформер остается плохо масштабируемым при экстремально длинных последовательностях (например, 100000 единиц текста и более), что ограничивает его применимость в задачах долговременной памяти.

В контексте задач быстрой оценки новых архитектур механизм FlashAttention может быть полезен в тех случаях, когда исследователь планирует использовать классические блоки внимания в трансформере и ищет способ ускорить обучение на упрощенных корпусах. Тем не менее он по-прежнему требует обучения модели хотя бы на части данных и не решает проблему эффективной диагностики архитектуры без дорогостоящего предобучения. Более того, архитектуры, не основанные на механизме самовнимания (например, Mamba [12], S4 [7]), не могут использовать алгоритм FlashAttention напрямую. Это ограничивает область применения и подчеркивает, что этот механизм является способом повышения эффективности существующего архитектурного шаблона трансформера, нежели универсальным средством для оценки или разработки новых языковых моделей.

Одной из ключевых оптимизаций, применяемых в больших языковых моделях при генерации текста, является использование так называемого KV-кэша (Key-Value Cache) [13]. Этот механизм позволяет значительно сократить вычислительные затраты при автогенерации последовательности, устраняя необходимость многократного пересчета коэффициентов внимания ко всем предыдущим позициям текста на каждом новом шаге вывода. Он особенно важен для трансформеров в режиме авторегрессионной генерации, где текст генерируется по одному токену за раз.

Суть KV-кэширования заключается в размене части времени вычисления ответа обученной модели во время генерации на память: матрицы K и V для

каждого слоя и каждой головы трансформера сохраняются в кэше и переиспользуются, а не пересчитываются на каждом шаге, что позволяет добиться линейного времени генерации относительно длины входной последовательности.

Тем не менее, как и алгоритм FlashAttention, KV-кэш не является архитектурной инновацией. Это инженерная оптимизация, направленная на повышение производительности стандартного механизма самовнимания. Кроме того, он применим только к авторегрессионным моделям с декодером и не используется, например, в энкодерных архитектурах (BERT) или в задачах без автогенерации. В контексте задач оценки новых архитектур по способности к работе с длинными последовательностями KV-кэш не предоставляет полезной информации о внутренней памяти модели. Он не влияет на архитектурную глубину или когнитивные способности, а только ускоряет выполнение уже обученной модели. Это делает его ценным инструментом для развертывания, но ограниченно применимым в задачах архитектурного анализа и ранней диагностики.

ОПИСАНИЕ МЕТОДА

Предлагаемое решение включает в себя количественное упрощение модели, использование конкретных синтетических задач и сбор показателей в процессе и по итогам обучения.

Количественное упрощение модели

Поскольку целью применения описываемого инструмента является оценка именно характеристик памяти, предполагается, что упомянутые характеристики можно оценить без обучения полноразмерной модели. Используются конфигурации сетей, количество слоев, голов внимания и размер внутреннего представления в которых были сокращены до предела, при котором модели еще сохраняют свои свойства выразительности, но уже могут обучаться достаточно быстро для практической применимости метода. Предполагается, что архитектура, показавшая себя лучше других при небольшом числе параметров, покажет себя лучше и при полноразмерной реализации.

Нужно отметить, что описываемый метод не предполагает ни количественной оценки «достаточного» числа параметров или слоев, которое необходимо сохранить для выразительности архитектуры, ни алгоритма получения такой оценки. Поскольку при использовании описываемого метода время тестирования одной архитектуры измеряется в часах, а не в днях и неделях, можно подобрать минимально выразительную конфигурацию эмпирически. Тем не менее это требует дополнительного времени исследователя и остается актуальным вопросом при большом количестве тестируемых архитектур.

Синтетические диагностические задачи

Ключевой составляющей метода является обучение модели на наборе из трех синтетических задач, отобранных по следующим критериям.

1. Простота генерации: каждая задача должна быть легко воспроизводиться, не должна требовать хранения больших корпусов данных или знания конкретного естественного языка, предметной области или какой-либо характеристики словаря. Такое требование позволяет оценивать модель безотносительно корпусов предобучения и сконцентрироваться на характеристиках памяти.
2. Диагностическая специфичность: каждая задача должна оценивать конкретный аспект работы внутренней памяти модели, позволяя делать интерпретируемые выводы.
3. Сложность решения: задачи должны последовательно увеличивать нагрузку на память, от базового воспроизведения до извлечения структурированной информации, позволяя начинать с оценки базовой способности воспроизвести информацию и заканчивая пониманием структуры входа и взаимосвязи частей этой структуры.

В соответствии с этими критериями были выбраны три синтетические задачи, представленные ранее в работе [15]:

- 1) задача копирования (Copy Task) – модель должна воспроизвести входную последовательность без изменений. Эта задача является базовой и позволяет оценить способность модели к удержанию линейной последовательности на всем протяжении контекста. Она особенно

чувствительна к затуханию градиентов и забыванию на больших расстояниях;

- 2) задача переворота (Reverse Task) – модель должна вывести ту же последовательность, но в обратном порядке. Это требует от модели не только хранения, но и перестройки последовательности во внутреннем представлении, что отражает способность к манипулированию содержимым памяти;
- 3) задача ассоциативного извлечения (Associative Retrieval Task) – во входную последовательность включаются пары «ключ – значение», а также отдельный запрос по ключу. Модель должна вернуть соответствующее значение. Эта задача проверяет наличие структурного представления во внутреннем состоянии модели, близкого к ассоциативной памяти.

Порядок задач соответствует увеличению когнитивной сложности. Если в первом случае достаточно механической передачи информации, то в третьей задаче модель должна интерпретировать структуру входа и избирательно реагировать на заданный шаблон.

Метрики и статистики

Для всесторонней оценки качества модели и диагностики ее поведения применялся набор аналитических показателей, получаемых в процессе обучения и непосредственного применения сети. Все они рассчитаны на анализ в пределах одной порции обрабатываемых данных, что позволяет осуществлять интерактивный контроль за ходом эксперимента.

Собирались следующие статистики.

1. Распределение точности предсказания по номеру токена внутри одного примера входа. Этот показатель отражает, какие части последовательности воспроизводятся хуже, и позволяет выявить позиционную деградацию памяти (например, хуже воспроизводится вторая половина текста).
2. Объем используемой оперативной памяти во время обучения. Этот показатель позволяет сопоставить объем памяти, необходимой для

вычисления ответа, с параметрами модели и задачей, давая косвенную оценку аппаратной стоимости архитектуры.

3. Время обучения одной эпохи. Этот параметр является важным показателем инженерной эффективности, он позволяет сравнивать архитектуры по затратам на обучение при прочих равных условиях.

4. Распределение вклада токенов в функцию потерь. Показывает, какие участки входа дали наибольший вклад в ошибку модели и, следовательно, были плохо предсказаны. Он дополняет показатель точности и позволяет глубже интерпретировать поведение сети.

Основным критерием оценки является максимальная длина последовательности, при которой модель сохраняет приемлемую среднюю точность ответа (Assurasy). Порог допустимого падения точности задавался эмпирически. При снижении качества ниже установленного порога считалось, что модель утратила способность к удержанию информации на данной длине контекста. Таким образом, определение «внутренней памяти» сводится к выявлению границы длины, при которой архитектура сохраняет функциональность.

ЭКСПЕРИМЕНТЫ

Механика подачи данных

Поскольку некоторые модели имеют строго ограниченную длину входа, в течение всех экспериментов данные подавались единым образом – посредством скользящего окна. Это позволяет проводить эксперименты независимо от архитектуры сети. В таком случае в каждом эксперименте появляются две интересующие нас переменные – длина окна, подававшегося на каждом шаге эксперимента на вход, и текущая длина последовательности, которая может превышать длину окна контекста или быть меньше этой длины.

Архитектуры

В качестве исследуемых моделей были выбраны три архитектуры: GPT-2, Mamba и GPT-X. Первые две хорошо изучены.

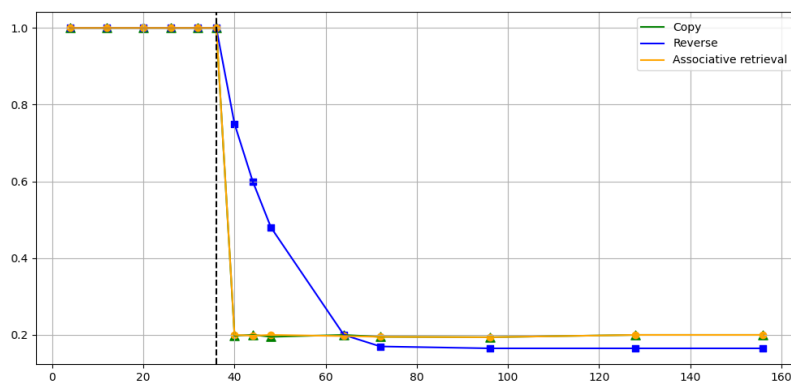
GPT-2 представляет собой классический трансформер с авторегрессионным вниманием.

Mamba представляет собой линейно масштабируемую по памяти и времени архитектуру, основанную на идеях рекуррентной обработки входа. В отличие от трансформеров, она не использует механизм самовнимания и сохраняет внутреннее состояние между шагами, что делает ее перспективной для задач с длинными зависимостями. Однако из-за последовательной природы вычислений Mamba не позволяет эффективно использовать параллелизм на видеокартах, что делает ее обучение медленным и ресурсоемким по сравнению с моделями, основанными на механизме внимания.

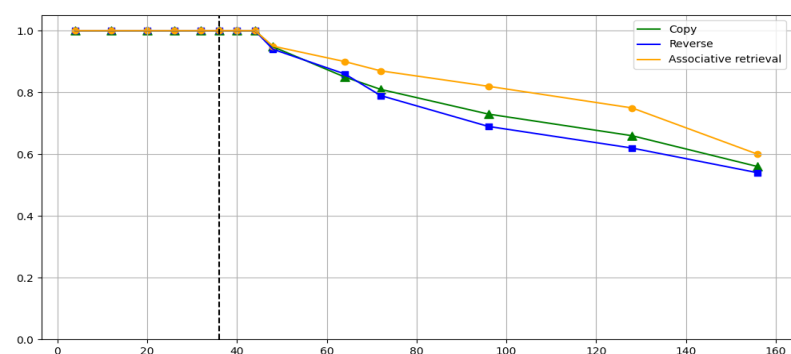
Модель GPT-X включена в исследование как объект с заранее неизвестным поведением, что делает ее тестирование особенно показательным. Результаты ее тестирования позволяют оценить пригодность предложенного инструмента для реального применения.

Результаты экспериментов

На рис. 1 видно, что с достижением предела окна контекста модель GPT-2 резко теряет в качестве предсказания, в то время как Mamba продолжает отвечать на сравнительно высоком уровне точности. Как известно, классическая архитектура трансформера не предполагает внутренних структур памяти, из-за чего такая модель не имеет возможности сохранять где-либо во внутреннем представлении участки входа, которые она видела ранее вне текущего окна. Поэтому при выходе первого токена за пределы окна, как при длине 36, модель не может его воспроизвести, как и последующие участки входа при дальнейшем сдвиге окна контекста. Аналогичная ситуация наблюдается и в двух других примерах. В табл. 1 можно проследить предметно, что точность GPT-2 практически не изменяется с увеличением длины последовательности свыше 72 токенов.



(a)



(b)

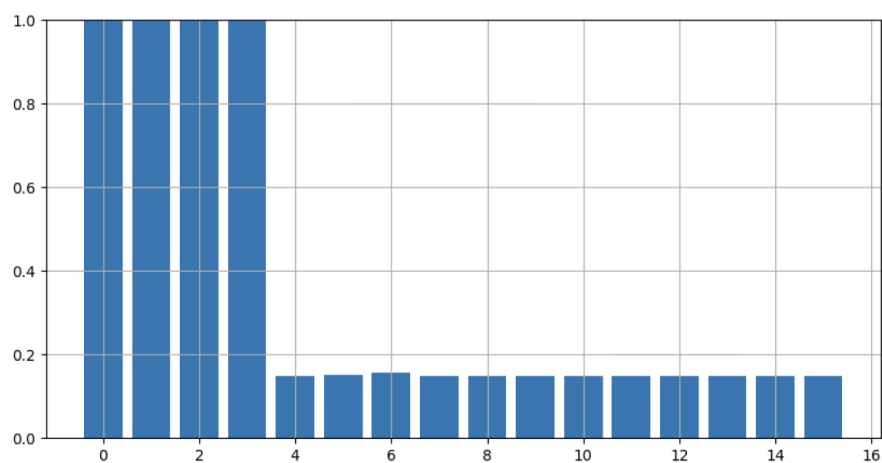
Рис. 1. График зависимости точности от длины последовательности для моделей GPT-2 (a) и Mamba (b) на задачах копирования, переворота и ассоциативного извлечения при размере скользящего окна в 36 токенов.

Табл. 1. Результаты моделей GPT-2 и Mamba на задачах копирования (copy), перевота (reverse) и ассоциативного извлечения (AR)

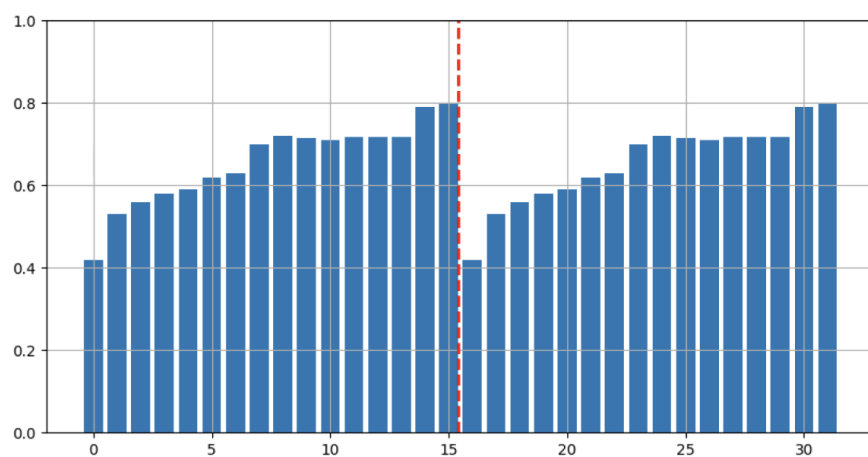
Длина	4	12	26	32	36	40	44	48	56	72	96	128	156
GPT-2, Copy	1	1	1	1	0.20	0.20	0.20	0.20	0.20	0.19	0.20	0.20	0.20
Mamba, Copy	1	1	1	1	1	1	1	0.95	0.85	0.81	0.73	0.66	0.56
GPT-2, Reverse	1	1	1	1	1	0.75	0.60	0.48	0.20	0.17	0.17	0.17	0.17
Mamba, Reverse	1	1	1	1	1	1	1	0.94	0.86	0.79	0.69	0.62	0.54

GPT-2, Associative retrieval	1	1	1	1	0.20	0.19	0.19	0.19	0.20	0.20	0.20	0.20	0.20
Mamba, Associative retrieval	1	1	1	1	1	1	0.95	0.93	0.90	0.87	0.82	0.75	0.60

Модель Mamba за счет своей рекуррентной архитектуры потенциально способна обрабатывать бесконечно длинные зависимости и сохранять их в своем внутреннем представлении, благодаря чему точность ее предсказаний почти не падает с ростом длины.



(a)



(b)

Рис. 2. График зависимости точности от номера токена в последовательности для модели GPT-2 (a) и GPT-X (b) на задаче переворота при размере скользящего окна в 8 токенов (a) и 32 токена (b).

На рис. 2 представлены графики распределения точности предсказания в задаче переворота для двух моделей: классической трансформерной архитектуры GPT-2 и экспериментальной модели GPT-X неизвестной структуры. Как и ожидалось, модель GPT-2 продемонстрировала характерное поведение: при выходе токенов за пределы окна внимания точность предсказаний снижается до 15%, в соответствии с табл. 2. Это подтверждает ограниченность памяти трансформера фиксированной длиной окна контекста.

В случае модели GPT-X ситуация иная: точность остается высокой на всем протяжении последовательности, включая те токены, которые по идее должны выходить за пределы обозреваемого окна. На графике красной пунктирной линией отмечена граница, после которой поведение модели изменяется – это может указывать на наличие в ее архитектуре механизма обработки входа блоками с сохранением промежуточного состояния. Такой эффект может быть достигнут за счет встроенной памяти или оптимизации скрытых представлений, как это реализовано, например, в некоторых вариантах рекуррентных трансформеров [14] или state-space моделей.

Поведение GPT-X указывает на наличие у нее внутренней памяти, превышающей по емкости окно стандартного самовнимания. Это делает модель особенно интересной для дальнейшего анализа: требуется исследовать характер хранения информации, ее объем, механизм извлечения и устойчивость к увеличению длины входа. Такие особенности могут иметь важное значение при разработке архитектур с расширенными возможностями долговременного контекста.

Табл. 2. Результаты моделей GPT-2 и GPT-X на задаче переворота (reverse) при длине контекстного окна в 8 и 32 токена соответственно.

Позиция токена во входной посл-	Точность, GPT-X	Точность, GPT-2	Позиция токена во входной посл-ти	Точность, GPT-X

ти				
0	0.42	1	16	0.42
1	0.53	1	17	0.53
2	0.56	1	18	0.56
3	0.58	1	19	0.58
4	0.59	0.15	20	0.59
5	0.62	0.151	21	0.62
6	0.63	0.157	22	0.63
7	0.7	0.15	23	0.7
8	0.72	0.15	24	0.72
9	0.715	0.15	25	0.715
10	0.71	0.15	26	0.71
11	0.717	0.15	27	0.717
12	0.717	0.15	28	0.717
13	0.718	0.15	29	0.718
14	0.79	0.15	30	0.79
15	0.8	0.15	31	0.8

ЗАКЛЮЧЕНИЕ

Представлен инструмент для быстрой диагностики архитектур больших языковых моделей в аспекте их способности обрабатывать длинные контексты. Актуальность задачи связана с тем, что эффективность современных больших языковых моделей определяется не только качеством генерации, но и глубиной внутренней памяти. Полноценное обучение каждой новой архитектуры требует

значительных ресурсов, поэтому необходимы методы раннего и дешевого тестирования.

Предложен диагностический инструмент на основе синтетических задач – задачи копирования, задачи переворота и задачи ассоциативного извлечения. Эти тесты позволяют оценить объем внутренней памяти модели и проследить, на какой длине последовательности сохраняется приемлемая точность. Каждая задача вводит свой уровень сложности: копирование проверяет удержание последовательности в памяти, переворот – способность к перестановке участков входа, а ассоциативное извлечение – выборочное обращение к элементам памяти.

Проведено сравнение архитектур GPT-2, Mamba и GPT-X в сопоставимых условиях. Mamba продемонстрировала наилучшие результаты на всех задачах, устойчиво работая с последовательностями, длина которых превышает размер контекстного окна. GPT-2 показала ожидаемые результаты, быстро теряя точность при увеличении длины входа. GPT-X заняла промежуточное положение, что может быть связано с внутренней блочной обработкой данных в этой сети.

Предложенный подход продемонстрировал эффективность при сравнении архитектур по их способности к долговременному хранению информации и может быть использован как инструмент предварительной оценки и выбора наиболее перспективных из них.

СПИСОК ЛИТЕРАТУРЫ

1. *Kaplan J., McCandlish S., Henighan T., et al.* Scaling Laws for Neural Language Models // arXiv preprint arXiv:2001.08361. 2020.
<https://doi.org/10.48550/arXiv.2001.08361>
2. *Brown T., Mann B., Ryder N., et al.* Language Models are Few- Shot Learners // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 1877-1901. <https://doi.org/10.5555/3495724.3495883>
3. *Beltagy I., Peters M. E., Cohan A.* Longformer: The Long- Document Transformer // arXiv preprint arXiv:2004.05150. 2020.
<https://doi.org/10.48550/arXiv.2004.05150>
4. *Radford A., Wu J., Child R., Luan D., Amodei D., Sutskever I.* Language Models are Unsupervised Multitask Learners // OpenAI. 2019.

5. Common Crawl Foundation. Common Crawl dataset.
<https://commoncrawl.org>
 6. Gu A., Goel K., Ré C. Efficiently Modeling Long Sequences with Structured State Spaces // International Conference on Learning Representations (ICLR). 2022.
 7. Gao L., Biderman S., Black S., et al. The Pile: An 800 GB Dataset of Diverse Text for Language Modeling // arXiv preprint arXiv:2101.00027. 2020.
<https://doi.org/10.48550/arXiv.2101.00027>
 8. Eldan R., Li Y. TinyStories: How Small Can Language Models Be and Still Speak Coherent English? // arXiv preprint arXiv:2305.07759. 2023.
<https://doi.org/10.48550/arXiv.2305.07759>
 9. Dao T. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness // Advances in Neural Information Processing Systems (NeurIPS). 2022. Vol. 35. P. 16344-16359. <https://doi.org/10.48550/arXiv.2205.14135>
 10. Gu A., Goel K., Dao T., et al. Mamba: Linear-Time Sequence Modeling with Selective State Spaces // International Conference on Learning Representations (ICLR). 2024.
 11. Kwon W., Lee S., Li S., Zaharia M., Zhang H., Stoica I., Sheng Y., Crichton W., Xie S., Gonzalez J. Efficient Memory Management for Large Language Model Inference with KV-Caching // arXiv preprint arXiv:2309.06180. 2023.
<https://doi.org/10.48550/arXiv.2309.06180>
 12. Vaswani A., Shazeer N., Parmar N., et al. Attention Is All You Need // Advances in Neural Information Processing Systems (NIPS). 2017. Vol. 30. P. 5998–6008.
<https://doi.org/10.5555/3295222.3295349>
 13. Tay Y., Bahri D., Metzler D., et al. Long Range Arena: A Benchmark for Efficient Transformers // arXiv preprint arXiv:2011.04006. 2020.
<https://doi.org/10.48550/arXiv.2011.04006>
 14. Bulatov A., Kuratov Y., Burtsev M. Recurrent Memory Transformer // Advances in Neural Information Processing Systems. 2022. Vol. 35. P. 11079-11091.
<https://doi.org/10.48550/arXiv.2207.06881>
-

A TOOL FOR RAPID DIAGNOSTICS OF MEMORY IN NEURAL NETWORK ARCHITECTURES OF LANGUAGE MODELS

P. A. Gavrikov¹ [0009-0009-6175-2883], A. K. Usmanov² [0009-0006-2535-1052],
D. Revaev³ [0009-0008-1630-1912], S. N. Buzykanov⁴ [0009-0005-3405-0705]

¹⁻⁴*Moscow Institute of Physics and Technology, Moscow, Russia*

¹gavrikov.pa@phystech.edu, ²usmanov.ak@phystech.edu,

³revaev.d@phystech.edu, ⁴bsn1977@mail.ru

Abstract

Large Language Models (LLMs) have evolved from simple n-gram systems to modern universal architectures; however, a key limitation remains the quadratic complexity of the self-attention mechanism with respect to input sequence length. This significantly increases memory consumption and computational costs, and with the emergence of tasks requiring extremely long contexts, creates the need for new architectural solutions. Since evaluating a proposed architecture typically requires long and expensive full-scale training, it is necessary to develop a tool that allows for a rapid preliminary assessment of a model's internal memory capacity.

This paper presents a method for quantitative evaluation of the internal memory of neural network architectures based on synthetic tests that do not require large data corpora. Internal memory is defined as the amount of information a model can reproduce without direct access to its original inputs.

To validate the approach, a software framework was developed and tested on the GPT-2 and Mamba architectures. The experiments employed copy, inversion, and associative retrieval tasks. Comparison of prediction accuracy, error distribution, and computational cost enables a fast assessment of the efficiency and potential of various LLM architectures.

Keywords: *large language models; neural network architecture; internal memory; long-term information retention; sequence processing; functional memory measurement; architecture comparison.*

REFERENCES

1. Kaplan J., McCandlish S., Henighan T., et al. Scaling Laws for Neural Language Models // arXiv preprint arXiv:2001.08361. 2020.
<https://doi.org/10.48550/arXiv.2001.08361>
2. Brown T., Mann B., Ryder N., et al. Language Models are Few- Shot Learners // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 1877-1901. <https://doi.org/10.5555/3495724.3495883>
3. Beltagy I., Peters M. E., Cohan A. Longformer: The Long- Document Transformer // arXiv preprint arXiv:2004.05150. 2020.
<https://doi.org/10.48550/arXiv.2004.05150>
4. Radford A., Wu J., Child R., Luan D., Amodei D., Sutskever I. Language Models are Unsupervised Multitask Learners // OpenAI. 2019.
5. Common Crawl Foundation. Common Crawl dataset.
<https://commoncrawl.org>
6. Gu A., Goel K., Ré C. Efficiently Modeling Long Sequences with Structured State Spaces // International Conference on Learning Representations (ICLR). 2022.
7. Gao L., Biderman S., Black S., et al. The Pile: An 800 GB Dataset of Diverse Text for Language Modeling // arXiv preprint arXiv:2101.00027. 2020.
<https://doi.org/10.48550/arXiv.2101.00027>
8. Eldan R., Li Y. TinyStories: How Small Can Language Models Be and Still Speak Coherent English? // arXiv preprint arXiv:2305.07759. 2023.
<https://doi.org/10.48550/arXiv.2305.07759>
9. Dao T. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness // Advances in Neural Information Processing Systems (NeurIPS). 2022. Vol. 35. P. 16344-16359. <https://doi.org/10.48550/arXiv.2205.14135>
10. Gu A., Goel K., Dao T., et al. Mamba: Linear-Time Sequence Modeling with Selective State Spaces // International Conference on Learning Representations (ICLR). 2024.
11. Kwon W., Lee S., Li S., Zaharia M., Zhang H., Stoica I., Sheng Y., Crichton W., Xie S., Gonzalez J. Efficient Memory Management for Large Language Model Inference with KV-Caching // arXiv preprint arXiv:2309.06180. 2023.
<https://doi.org/10.48550/arXiv.2309.06180>

12. Vaswani A., Shazeer N., Parmar N., et al. Attention Is All You Need // Advances in Neural Information Processing Systems (NIPS). 2017. Vol. 30. P. 5998–6008.
<https://doi.org/10.5555/3295222.3295349>
13. Tay Y., Bahri D., Metzler D., et al. Long Range Arena: A Benchmark for Efficient Transformers // arXiv preprint arXiv:2011.04006. 2020.
<https://doi.org/10.48550/arXiv.2011.04006>
14. Bulatov A., Kuratov Y., Burtsev M. Recurrent Memory Transformer // Advances in Neural Information Processing Systems. 2022. Vol. 35. P. 11079-11091.
<https://doi.org/10.48550/arXiv.2207.06881>
-

СВЕДЕНИЯ ОБ АВТОРАХ



ГАВРИКОВ Павел Андреевич – студент 1-го курса магистратуры ФРКТ МФТИ. Область научных интересов: Большие языковые модели, генеративные нейронные сети, теория компиляторов. Число научных публикаций: 2.

Pavel Andreevich GAVRIKOV – First-year Master's student, Department of Radio Engineering and Computer Technology (DREC), Moscow Institute of Physics and Technology (MIPT). Research interests: large language models, generative neural networks, compiler theory. Number of scientific publications: 2.

email: gavrikov.pa@phystech.edu

ORCID: 0009-0009-6175-2883



УСМАНОВ Азамат Комилжон угли – студент 1-го курса магистратуры ФПМИ МФТИ. Область научных интересов: обработка естественного языка, авторегрессионные модели, оптимизация больших языковых моделей. Число научных публикаций: 1.

Azamat Komiljon ugli USMANOV – First-year Master's student, Department of Applied Mathematics and Computer Science (DAPCS), Moscow Institute of Physics and Technology (MIPT). Research interests: natural language processing, autoregressive models, optimization of large language models. Number of scientific publications: 1.

email: usmanov.ak@phystech.edu

ORCID: 0009-0006-2535-1052



РЕВАЕВ Дмитрий аспирант 1-го года, Кафедра Интеллектуальных Систем ФПМИ МФТИ. Область научных интересов: большие языковые модели, обучение с подкреплением, теория информации.

Dmitriy REVAYEV – First-year PhD student, The Intelligent Systems Department, The Phystech School of Applied Mathematics and Informatics, Moscow Institute of Physics and Technology (MIPT). Research Interests: Large Language Models, Reinforcement Learning, Information Theory.

email: revaev.d@phystech.edu

ORCID: 0009-0008-1630-1912



БУЗЫКАНОВ Сергей Николаевич – д. т. н., профессор кафедры радиотехнических систем НИУ «МЭИ», доцент кафедры МТТК МФТИ, ведущий инженер Huawei. Область научных интересов: цифровая обработка сигналов, системы обработки информации, системы искусственного интеллекта и большие языковые модели. Число научных публикаций – более 70.

Sergey Nikolaevich BUZYKANOV – Doctor of Engineering, Professor of the Radio Engineering Systems Department at National Research University "MPEI," Associate Professor of the MTTC Department at MIPT, and a Huawei Lead Engineer. Research interests: digital signal processing, data processing systems, artificial intelligence systems and large language models. Over 70 scientific publications.

email: bsn1977@mail.ru

ORCID: 0009-0005-3405-0705

Материал поступил в редакцию 9 ноября 2025 года

УДК 004.932.75'1

СИСТЕМА ТЕСТИРОВАНИЯ КОНТРОЛЛЕРОВ, ОСНОВАННАЯ НА РАСПОЗНАВАНИИ ТЕКСТА НА ЭКРАНЕ

А. А. Докукин^[0009-0001-9928-7851]

*Федеральный исследовательский центр «Информатика и управление»
Российской академии наук, г. Москва, Россия*

adokukin@frccsc.ru

Аннотация

Описано решение задачи тестирования контроллеров на основе чтения информации с их экрана. Для этого разработана программно-аппаратная система, состоящая из камеры и программных модулей, реализующих необходимые алгоритмы и методы: модуля предобработки изображения; модуля определения типа меню; модуля обработки символов шрифта; модуль чтения текста, в том числе, написанного различными шрифтами; собственно модуля тестирования. Система реализована для контроллеров определенного типа с монохромным дисплеем 128 x 64 точек. Все методы реализованы на языке Python с использованием популярных библиотек. Система внедрена в эксплуатацию и на данный момент осуществляет автоматизацию нескольких наиболее трудоемких тестов. Поддерживается расширение их набора в виде плагинов.

Ключевые слова: *компьютерное зрение, распознавание текста, тестирование контроллеров.*

ВВЕДЕНИЕ

При разработке различных контроллеров, таких как контроллеры «умного дома» или контроллеры отопления, возникает задача тестирования встраиваемого в них программного обеспечения (ПО). Контроллерами принято называть универсальные устройства, реализованные на базе слабого процессора, с очень ограниченными другими ресурсами, и способные при этом непосредственно управлять исполнительными механизмами, такими как насосы, сервоприводы и т. п. При этом контроллеры реализуют достаточно сложную логику. Так, например, контроллеры отопления Smartweb [1] работают в составе распределенной

сети, каждый контроллер в которой реализует часть общей функциональности, при этом отправляя мониторинговые данные на облачный сервер.

При автоматизации тестирования контроллеров возникают различные проблемы. Процедуры отслеживания результатов обработки длинных сценариев невозможно разместить на контроллере вместе с его собственным ПО из-за ограниченности ресурсов: оперативной и постоянной памяти. Частичная эмуляция работы контроллера возможна на более мощных устройства, таких как персональный компьютер. Но в этом случае исключено взаимодействие с реальным экраном устройства, его файловой системой, исполнительными устройствами. Поэтому при проверке ПО реальных контроллеров значительные усилия ложатся на плечи людей-тестировщиков. От них требуются наблюдение и вмешательство в длительные сценарии, исполняемые непосредственно в сети контроллеров. При многократном исполнении тестовых сценариев внимание людей притупляется, поэтому возникает задача автоматизации рутинных операций такого типа более сложными методами, например, на основе машинного распознавания информации с экрана контроллера.

В настоящем исследовании рассмотрены реальные контроллеры серии Smartweb K [1] с дисплеем, его целью является поиск комбинации технических средств и алгоритмов для чтения информации с экрана контроллера и построение системы тестирования на основе полученных данных [2, 3].

Устройства обладают монохромным экраном размером 128 x 64 пикселя, изображение на котором формируется по определенным правилам. Тип меню, выбранного на контроллере, определяет состав отображаемой информации: положение строк текста и графических элементов. Задача естественным образом разбивается на две подзадачи: определение текущего типа меню и распознавание графических элементов в соответствии с шаблоном, в том числе чтение текста.

Такой подход обладает своими ограничениями, т. к. не позволяет обнаружить наличие символов за пределами полей шаблона, что легко дается человеку. Тем не менее он покрывает подавляющее число реальных сценариев тестирования.

ОПРЕДЕЛЕНИЕ ТИПА МЕНЮ

Контроллеры серии Smartweb K обладают достаточно простым монохромным дисплеем размером 128 на 64 пикселей, поэтому чтение информации с него представляется несложным. Однако эта простота не отменяет необходимости получать изображение экрана. В ходе настоящего исследования используемые для этого технические средства менялись, первый стенд был собран на основе бытовой веб-камеры из высшего ценового сегмента. Установка и получаемое изображение показаны на рис. 1. Как видно, несовершенство установки порождает дополнительную задачу – определение реального положения экрана на изображении камеры. Центровка и масштабирование изображения с помощью штатива и ПО веб-камеры не представляются возможным.



Рис. 1. Тестовый стенд на основе веб-камеры и получаемое изображение.

На рис. 2 представлены изображения некоторых из 49 различных типов меню, отображаемых контроллером. Как видно, они достаточно сильно различаются, но эти различия трудно описать, поэтому их классификация является идеальной задачей для нейронной сети. Единственным препятствием для ее применения является необходимость сбора обучающей выборки. Поскольку вручную переключать экраны контроллера и фотографировать их нет возможности, была написана программа генерации правдоподобных изображений различных меню со случайным заполнением. Это оказалось наиболее трудоемкой операцией данного этапа. Часть полученных изображений и представлена на рис. 2.



Рис. 2. Примеры различных типов меню контроллера из обучающей выборки.

Для классификации типов меню была подобрана трехслойная сверточная нейронная сеть [4] следующей конфигурации: (1 -> 32, 3 x 3), (32 -> 16, 3 x 3), (16 -> 49, 60 x 124). Здесь первая пара чисел – это число входных и выходных слоев, а вторая – размер ядра. Первые слои достаточно стандартные, особого внимания заслуживает последний. Число выходных слоев в нем равно числу классов, т. е. типов меню, а размер ядра приводит к тому, что результатом свертки изображения экрана становится одно число. Сеть была реализована с помощью библиотеки pytorch [5], очень быстро обучается (десятки секунд на ноутбуке без дискретной видеокарты) и еще быстрее работает. При обучении на 100 экземплярах изображений каждого класса точность распознавания тестовой выборки составляет 100%. Для теста были использованы по 10 изображений каждого класса.

Кроме того, построенная архитектура позволяет определять положение экрана контроллера на более широком поле. Для второго теста сгенерированные изображения располагались на более широком фоне (256 x 128) в случайной позиции. На выходе нейросети получается тензор размера 49 x 129 x 257, максимум значений которого соответствует классу и смещению экрана (использована нотация библиотек PIL и pytorch, в которых первая пространственная координата соответствует x, если речь идет про изображения, и y, если про тензоры). Тесты при том же размере выборок также показали 100%-ную точность (рис. 3).

Отметим также, что, эта архитектура сети, примененная к нефильтрованному изображению с камеры, дала отличный результат по стабильности опреде-

ления координат на разных типах экранов и скорости обработки потокового видео.

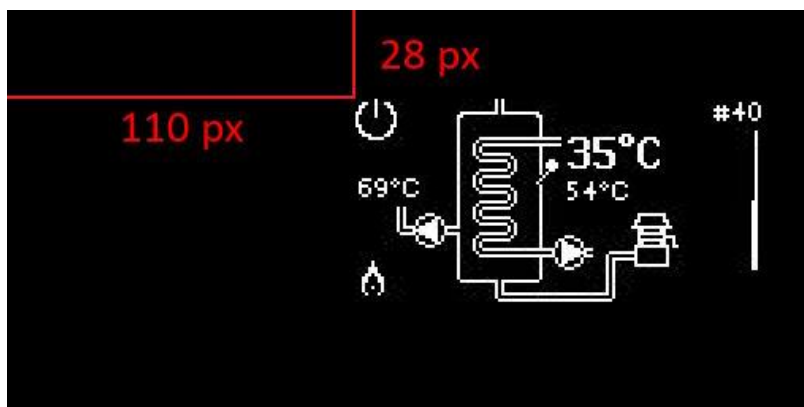


Рис. 3. Совместное определение класса меню и смещения экрана.

Наконец, эту же архитектуру можно использовать для определения масштаба изображения. Из-за того, что необходимо перебирать некоторый диапазон масштабов, эта процедура становится слишком трудоемкой для использования в реальном времени, но масштаб остается неизменным в течение тестовой сессии, поэтому такую калибровку можно производить один раз при запуске системы.

ОПРЕДЕЛЕНИЕ ОТДЕЛЬНЫХ СИМВОЛОВ

Задание типа меню однозначно определяет расположение на экране и содержимое полей. Это могут быть надписи, выполненные одним из трех шрифтов, либо иконки. Иконки, в свою очередь, могут быть статическими или динамическими, но последние представляют собой сменяющие друг друга статические. Таким образом, определение содержимого полей сводится к определению символов одного из четырех перечисленных алфавитов. Задача классификации таких символов, с одной стороны, упрощается по сравнению с задачей определения типа меню, т. к. классифицируемые изображения значительно уменьшаются. Вместо целых экранов размером 128 x 64 точек рассматриваются символы, размер которых имеет порядок 10 x 15. С другой стороны, задача усложняется, потому что значительно увеличивается число классов, а символы разных классов могут отличаться буквально на одну точку, например, «i», «l» и «!». Но самое главное: части символов могут совпадать с другими значками, и совместная класси-

фикация и определение их расположения становятся слишком затруднительными. Поэтому было решено использовать отдельно четыре нейронные сети: три шрифтовые и одну для иконок. Текстовые поля при этом ограничиваются по высоте размерами соответствующего алфавита. Ниже описаны исследования по обработке одного из шрифтов – 8-пиксельного.

Были рассмотрены две архитектуры: классическая сверточная нейронная сеть и нейросеть без обучения, разработанная специально для этой задачи. В качестве сверточной рассмотрена нейронная сеть, схожая с сетью, описанной в предыдущем разделе, с поправкой на размеры характерных изображений и число классов. Для ее обучения были использованы символы алфавита, по одному примеру для символа, и они же использовались для теста. Даже в такой простой постановке сеть не дала необходимого стопроцентного качества распознавания, причина чего становится понятной после исследования областей интереса на отдельных символах (рис. 4).

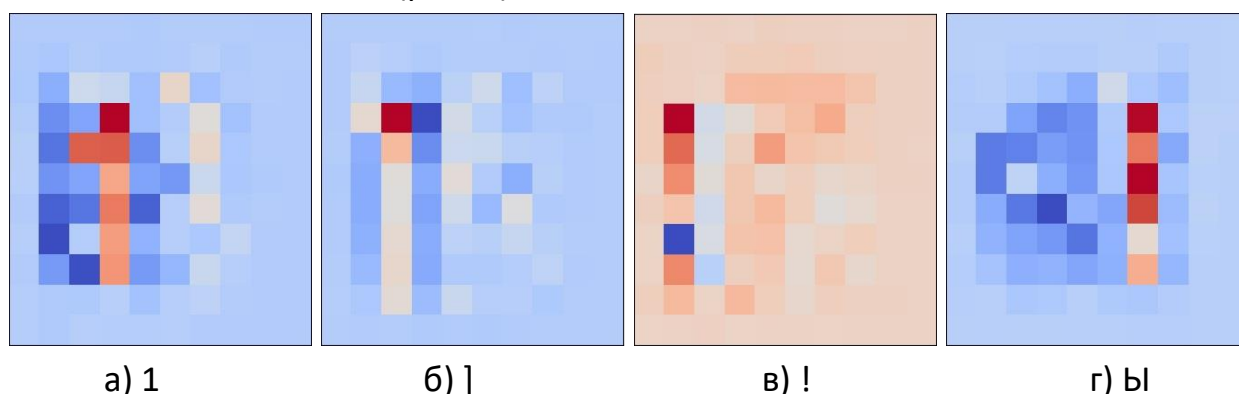


Рис. 4. Визуализация признаков классификации некоторых символов.

Полученные результаты показывают, что размеры выборок недостаточны для определения действительно характерных сочетаний в результате обучения сети. Для ее использования представляется необходимым разнообразить выборку с помощью различных аугментаций, таких как маскирование определенных областей. Но это также связано с трудностями по причине значительной похожести отдельных символов.

В то же время простота символов позволяет проводить непосредственное поточечное их сравнение. В целом использование обучения представляется из-

быточным для этой задачи. Однако удобство средств, предоставляемых библиотекой pytorch для работы со сверточными сетями, и скорость обработки изображений с их помощью являются важными практическими преимуществами. Это легло в основу схемы нейросетей без обучения, т. е. нейросетей, в которых веса вычисляются аналитически, минуя стадию обучения.

Для рассматриваемого шрифта была предложена однослойная схема (1 -> 143, 11 x 10), где 143 – число различных используемых символов, 8 x 11 – размер самой большой буквы, и 2 точки – пробелы вокруг. Работа сверточного слоя была модифицирована для работы по формуле $(2X - 1) \otimes T$, где X – изображение, T – маска буквы (1, если точка принадлежит букве, -1 – фон и отступы, 0 – за пределами ширины буквы), нормированная на ширину. Примеры таких масок для символов представлены на рис. 5 (можно сравнить их с результатами обучения на рис. 4).

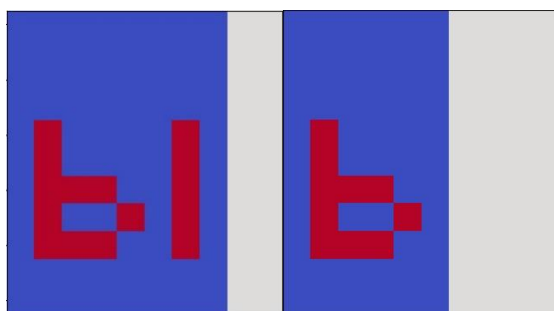


Рис. 5. Результаты вычисления ядер свертки для различных символов.

В отличие от схемы с обучением, данная схема дала 100%-ное качество при определении единственного символа. В неизвестной позиции в строке, т. е. в изображении символа, дополненного пустым фоном слева и справа, точность составила 99.3%, т. к. путаницу «l» и вертикальной черта «l» преодолеть не удалось. При определении же пяти случайных букв подряд точность составила около 97%. Ошибка с вклиниванием несуществующей буквы в последовательность дает сдвиг остальным и портит их. Например, в последовательности «ВмыЧ,» уверенно определилась буква «ь», что дало «ммыЧ,» в качестве результата. Исправление этого эффекта рассмотрено ниже.

Тесты с использованием реальных изображений текста, полученных с камеры, выявили еще одну принципиальную сложность. Невозможность точного

позиционирования камеры по нескольким осям приводит к едва заметным перспективным искажениям, которые влияют на точность: изображения текста из некоторых областей экрана перестают распознаваться, поскольку характеристики символов не оставляют никакой толерантности к искажениям. В следующем разделе описаны алгоритмы предобработки изображения, позволяющие избежать этого эффекта.

ПРЕДОБРАБОТКА ИЗОБРАЖЕНИЯ

Классификация типов меню оказалась достаточно устойчивой к мелким перспективным искажениям экрана за счет сложности и размера изображения. Это не относится к распознаванию отдельных символов, поскольку несовпадение даже одной точки может привести к полному изменению результата классификации. Для устранения искажений был предложен простой алгоритм: матрица перспективного преобразования восстанавливается по четырем угловым точкам экрана, на ее основе рассчитывается расположение всех точек раstra и затем выбирается ближайший к расчетному положению пиксель. Пример искажения, гипертрофированного для наглядности, и восстановленного раstra приведен на рис. 6.

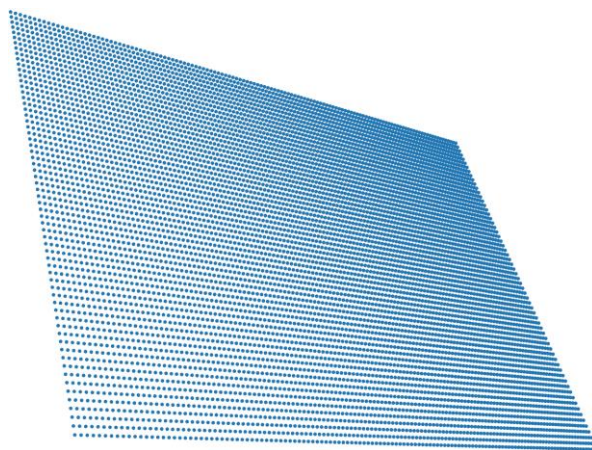


Рис. 6. Пример восстановленного раstra.

Восстановление матрицы перспективного преобразования $A = \|a_{ij}\|_{3 \times 3}$ производится с помощью решения системы линейных уравнений:

$$\begin{aligned}a_{11}x_i + a_{12}y_i + a_{13} &= r_i x'_i, \\ a_{21}x_i + a_{22}y_i + a_{23} &= r_i y'_i,\end{aligned}$$

$$a_{31}x_i + a_{32}y_i + 1 = r_i,$$

где (x_i, y_i) – координаты точки растра, (x'_i, y'_i) – координаты ее проекции на изображение. Для четырех точек, $i = \overline{1,4}$, система имеет единственное решение.

Еще одной проблемой является засветка соседних точек, особенно при увеличении выдержки для борьбы с мерцанием. Один из худших примеров такой засветки – это круги в символе процента (рис. 1), где черная точка находится в окружении 7 светящихся, и ее яркость значительно отличается от других точек фона. Для преобразования в бинарный вид картинки, получаемой с камеры, с устранением этой проблемы мы использовали метод Оцу [6], реализованный в библиотеке OpenCV для языка питон [7]. Заметим, что эта библиотека интенсивно используется в системе и заслуживает отдельного упоминания.

Более серьезную проблему представляло мерцание камеры, источником которого, по всей видимости, были даже не ее физические характеристики, а внутренние алгоритмы коррекции изображения, которые в сочетании с мерцанием экрана приводили к постоянной его деградации, вплоть до невозможности обработки через полчаса съемки. Эту проблему удалось решить с помощью замены камеры на промышленную камеру стандарта GigE Vision. На рис. 7 показаны внешний вид усовершенствованного стенда и полученное с камеры изображение меню, аналогичное рис. 1.

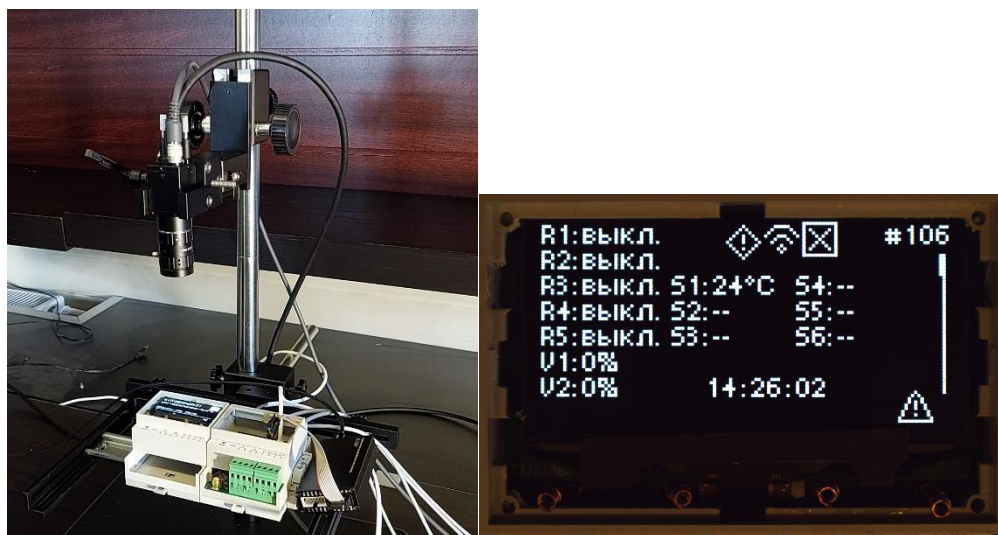


Рис. 7. Тестовый стенд на основе промышленной камеры и пример изображения.

Смена камеры решила проблемы с мерцанием – она работает независимо

от компьютера, и получаемое изображение не меняется месяцами. Кроме того, увеличилась четкость изображения и пропали проблемы с засветкой. Но новый штатив снял проблему позиционирования лишь частично, модуль устранения перспективных искажений был оставлен в системе. Устранения более сложных дисторсий, таких как бочкообразная, не потребовалось даже для веб-камеры. Бинаризация методом Оцу также была включена в последнюю версию системы.

РАСПОЗНАВАНИЕ СМЕСИ ШРИФТОВ

Методы, описанные выше, позволяют надежно определять изображенные на экране иконки и «читать» текст с экрана контроллера. Для этого используются некоторые простые эвристики, устраняющие неоднозначности шрифтов. Прежде всего, полное включение узкого кандидата в более широкий («b» в «bl») трактуется в пользу более широкого. Похожие допущения используются при покрытии кандидата парой соседних. Особенно это актуально при подсчете числа пробелов между символами.

Однако такой подход работает только с полями, содержащими единственный шрифт. В реальности же на контроллере используются надписи из смеси шрифтов, например, числовой индекс меню и его название имеют разный размер (рис. 2, слева внизу). В этой ситуации граница между шрифтами заранее не известна. Более того, длинные названия могут прокручиваться горизонтально, и индекс с подписью могут поменяться местами или даже образовать две границы. Поэтому необходимо иметь возможность распознавать такие тексты в общем виде. Это порождает дополнительные сложности. Прежде всего отменяется сделанное ранее предположение, что все надписи содержатся в строках высотой с размер шрифта, что возрождает проблему схожести частей одних символов на другие. Похожая неприятность возникает из-за схожести символов разных шрифтов. Так, прописная «Р» маленького шрифта не отличается от строчной «р» большого. Единственное различие – это высота строки, на которой определяется символ. Ножка строчной «р» свешивается ниже базовой линии. Это различие использовалось при разборе текста. Предполагалось, что внутри полей находятся

строки символов и соседние элементы должны поддерживать выбор альтернативы. Описанная идея представлена на рис. 8. Красным цветом показана потенциальная строка, если в голубой рамке определяется двоеточие, синим цветом – точка. Для наглядности каждая строка показана только с одной стороны от рамки.

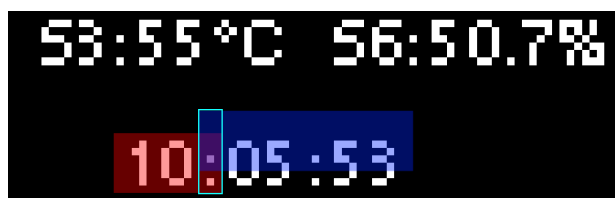


Рис. 8. Влияние соседних символов на выбор альтернативных гипотез.

Эту идею можно реализовать разными способами. В рамках проведенного исследования были осуществлены и проверены на практике два варианта ее реализации. Первый из них продолжает эксплуатировать концепцию нейросетей без обучения, использованную ранее для распознавания отдельных символов. Рассмотрим матрицу изображения $P_{H \times W}$, где H – его высота, W – ширина. «Шрифтовая» нейросеть F для шрифта высотой h и с числом символов n преобразует эту матрицу в тензор $T_{n \times H-h \times W} = F(P)$, в котором значение $T(k, y, x)$ соответствует оценке наличия символа с номером k в строке y и позиции x (положение определяется верхним левым пикселем). Аналогично можно работать со смесью шрифтов, для чего нужно объединить по первой координате тензоры $T_i = F_i(P)$ для нескольких шрифтов, дополнив пространственные измерения нулями, где необходимо. Для простоты выкладок опишем случай одного шрифта.

К полученному тензору T применим трехмерный [5] сверточный слой U с одним входным каналом, $n(F)$ выходными и ядром размера $n(F) \times 2w + 1$, где $n(F)$ – число символов в шрифте. Выходные каналы соответствуют символам шрифта и содержат поощрение этих символов от их соседей. В i -м канале j -я строка ядра содержит $2w - 1$ нулей и две единицы в позициях $w + 1 - w_j$ и $w + 1 + w_i$, поскольку поддержка левого символа зависит от его ширины, а правого – от ширины самого рассматриваемого символа. Тензор T далее заменим тензором $T + U(I(T))$, где I зануляет все позиции T , содержащие не единицы. Дальнейшая обработка достаточно очевидна и основана на тех же эвристиках.

Второй вариант из упомянутых выше реализует ту же идею, но как некую вариацию метода ветвей и границ. Столбцы тензора T перебираются последовательно и величина поощрений рассчитывается только для максимальных элементов, т. е. элементов, чья оценка равна 1. Поддержка выражается в умножении оценки символа на оценку его левого соседа, после чего немаксимальные ветки отбрасываются.

Оба варианта были реализованы практически и показали одинаковое, стопроцентное качество распознавания тестовых примеров. Различие проявилось только в скорости обработки. Нейросетевой вариант оказался примерно на порядок медленнее из-за большего числа лишних вычислений. Так, распознавание текста из 26 символов на изображении размером 148 на 20 пикселей с помощью дополнительной свертки заняло 179 мс. Эффективность разогретой сети увеличивается – десять подобных изображений подряд обрабатываются 936 мс. При этом второй метод обрабатывает за 11 мс одно, а за 222 мс – десять изображений. Возможно, при использовании графических ускорителей соотношение изменится, но в текущих условиях для практического применения был выбран второй подход. Сочетание описанных компонентов позволяет проводить обработку изображения экрана на персональном компьютере без графического ускорения в реальном времени с частотой порядка 10 кадров в секунду.

ВСПОМОГАТЕЛЬНЫЕ СРЕДСТВА

Стоит отметить, что полученная система чтения содержимого экрана еще не равносильна системе тестирования контроллеров. Для воспроизведения сценариев пользовательского взаимодействия и проверки их результатов необходимо осуществлять осмысленную навигацию по меню. В распоряжении системы имеется несколько команд, соответствующих кнопкам контроллера: «вверх», «вниз», «внутри» и «наружу». Но написание тестов на языке «15 раз нажать вверх, потом внутрь, ...» попросту невозможно, даже если смириться с неудобством такого описания. Состав меню рассматриваемых контроллеров, а именно набор и порядок элементов, может изменяться динамически. Кроме того, активный пункт при входе в меню может выбираться непредсказуемо. Поэтому для навигации был

написан высокоуровневый язык, позволяющий управлять ей в терминах «перейди в меню калибровки датчика № 1» и «установи значение 10». Перемещение в меню при этом осуществляется системой автоматически на основе информации, отображаемой на экране.

Дополнительно реализован механизм подключения новых тестов к системе в виде плагинов. Детали реализации оставим за пределами настоящей статьи, поскольку к теме они относятся лишь косвенно.

ЗАКЛЮЧЕНИЕ

Представлена программно-аппаратная система для тестирования контроллеров на основе распознавания информации, отображаемой на их экране. Система содержит как общеизвестные, так и специально разработанные для нее алгоритмы и методы, выбор которых подтвержден результатами исследований.

В основе системы лежит обработка в реальном времени отображаемой на экране информации. После выделения экрана, устранения перспективных искажений и бинаризации изображения методом Оцу выполняется классификация типа меню, затем разбор соответствующих этому меню элементов изображения – отдельных символов и строк текста.

Созданная система введена в эксплуатацию и уже выполняет обработку наиболее трудоемких из ручных тестов. Разработан инструментарий их написания, дальнейшее масштабирование системы является чисто программистской задачей.

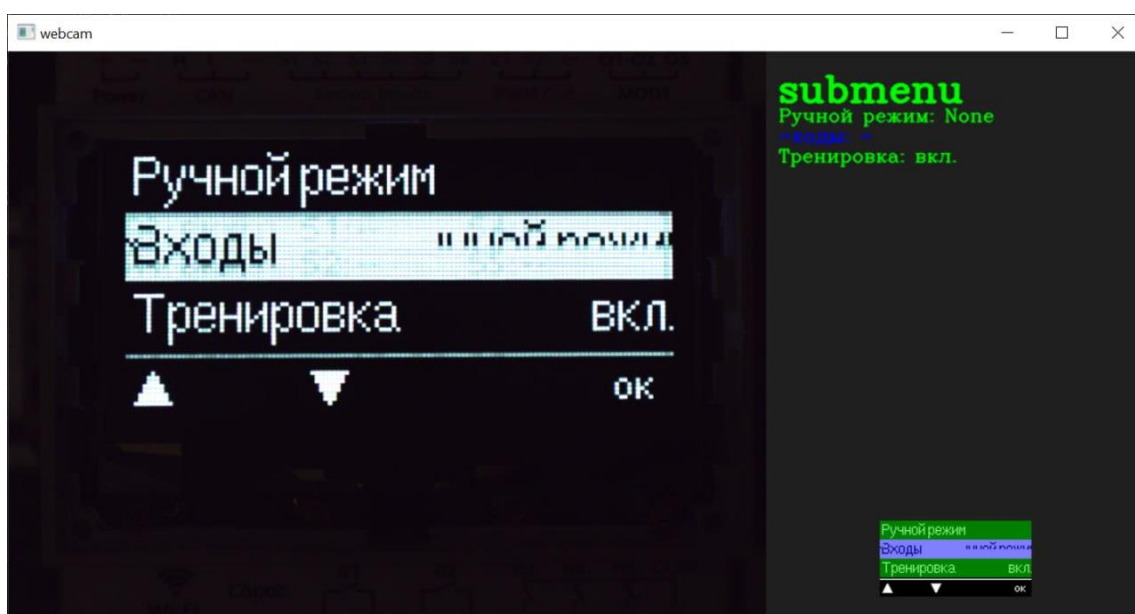


Рис. 9. Пример сбоя с появлением неожиданных символов и результат его обработки.

Открытым принципиальным вопросом остается поиск неожиданных артефактов. Система справляется с проверкой содержимого заданных полей на соответствие известному шаблону. Появление символов в неожиданных местах или неизвестных системе символов остается незамеченным при таком подходе. Обработка нештатных ситуаций такого типа возможна лишь косвенно. Реальный пример показан на рис. 9.

На рисунке представлены реальная фотография экрана, полученная камерой в системе, и результат ее обработки. Как видно, система не смогла прочесть букву «В» в слове «Входы» и полностью проигнорировала мусорные символы справа. Ситуацию удалось обнаружить и исправить благодаря сбою навигации и привлечению оператора.

Вместе с тем для человека сбой достаточно очевиден, и даже в какой-то степени понятна его причина – эхо верхней строчки отобразилось ниже. При некотором уточнении эта картинка даже позволяет с большой надежностью исключить аппаратный сбой. Дальнейшие исследования предполагается направить на автоматизацию обработки нештатных ситуаций такого типа и, в первую очередь, на отслеживание появления графических элементов, которые можно трактовать как символы из неизвестного алфавита, или корректных символов за пределами полей шаблона.

СПИСОК ЛИТЕРАТУРЫ

1. SmartWEB-K // [Электронный ресурс]
URL: <https://www.teplostart.ru/download/booklets/SmartWEB-K.pdf> (дата обращения: 06.10.2025)
2. Докукин А.А. О построении системы считывания информации с экрана контроллера // Интеллектуализация обработки информации (ИОИ-2024): Тез. докл. 15-й междунар. конф. (Гродно, 23–27 сент. 2024 г.) 2024. С. 92–93.
3. Докукин А.А. Развитие системы считывания информации с экрана контроллера // Математические методы распознавания образов (ММРО-2025): Тез. докл. 22-й Всеросс. конф. с междунар. участ. (Муром, 22–26 сент. 2025 г.) 2025.

C.139–140.

4. Le Cun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D. Handwritten Digit Recognition with a Back-Propagation Network // NIPS'89: Proc. 2nd Int. Conf. Neural Information Processing Systems. 1989. P. 396–404.

URL: [https://proceedings.neurips.cc/pa-](https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf)

[per/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf](https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf)

5. Ansel J., Yang E., He H. et al. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation // 29th ACM Int. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '24). 2024. Vol. 2. <https://doi.org/10.1145/3620665.3640366>

6. Otsu N. A Threshold Selection Method from Gray-Level Histograms // IEEE Transact. on Systems, Man, and Cybernetics 9.1. 1979. P. 62–66.

<https://doi.org/10.1109/TSMC.1979.4310076>

7. Bradski G. The OpenCV Library // Dr. Dobb's J. Software Tools. 2000. Vol. 120. P. 122–125.

A SYSTEM FOR TESTING CONTROLLERS BASED ON ON-SCREEN TEXT RECOGNITION

A. A. Dokukin^[0009-0001-9928-7851]

Federal Research Center “Informatics and Control” of the Russian Academy of Sciences, Moscow, Russia

adokukin@frccsc.ru

Abstract

A solution for the problem of testing controllers based on reading information from their screens is described. A hardware and software system has been developed for this purpose, consisting of a camera and software modules implementing the necessary algorithms and methods: an image preprocessing module; a menu type detection module; a font character processing module; a text reading module, including one written in various fonts; and the testing module itself. The system has been developed

for a specific type of controller with a monochrome 128x64 pixel display. All methods are implemented in Python using popular libraries. The system has been launched into test operation and currently automates several of the most labor-intensive tests. The test set can be expanded using plugins.

Keywords: *computer vision, text recognition, controller testing.*

REFERENCES

1. SmartWEB-K (In Russian) // [Electronic resource]
<https://www.teplostart.ru/download/booklets/SmartWEB-K.pdf> (access date: 06.10.2025)
2. Dokukin A.A. *O postroenii sistemy schityvaniya informacii s ekrana kontrollera* (in Russian) // Intellectual Data Processing (IDP-2024): Book of Abstracts 15th Int. Conf. (Grodno, September 23–27, 2024) 2024. P. 92–93.
3. Dokukin A.A. *Razvitie sistemy schityvaniya informacii s ekrana kontrollera* (In Russian) // Mathematical Methods for Pattern Recognition (MMPR-2025): Book of Abstracts 22nd Russian Conf. with Int. Participation (Murom, September 22–26, 2025) 2025. P. 139–140.
4. Le Cun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D. Handwritten Digit Recognition with a Back-Propagation Network // NIPS'89: Proc. 2nd Int. Conf. Neural Information Processing Systems. 1989. P. 396–404.
<https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf>
5. Ansel J., Yang E., He H. et al. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation // 29th ACM Int. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '24). 2024. Vol. 2. <https://doi.org/10.1145/3620665.3640366>
6. Otsu N. A Threshold Selection Method from Gray-Level Histograms // IEEE Transact. on Systems, Man, and Cybernetics 9.1. 1979. P. 62–66.
<https://doi.org/10.1109/TSMC.1979.4310076>
7. Bradski G. The OpenCV Library // Dr. Dobb's J. Software Tools. 2000. Vol. 120. P. 122–125.

СВЕДЕНИЯ ОБ АВТОРЕ



ДОКУКИН Александр Александрович – 1980 года рождения, старший научный сотрудник ФИЦ ИУ РАН, кандидат физико-математических наук. Закончил с отличием факультет ВМК Московского государственного университета им. М.В. Ломоносова в 2002 году. В 2005-м закончил аспирантуру того же факультета. В 2008-м защитил диссертацию на соискание степени кандидата физ.-мат. наук по теме «Синтез полиномов над экстремальными алгоритмами вычисления оценок». С 2000-го года по настоящее время работает в ВЦ РАН (затем ФИЦ ИУ РАН). Область научных интересов – распознавание образов, анализ данных. Опубликовал лично и в соавторстве 139 научных работ.

Aleksandr Aleksandrovich DOKUKIN graduated with honors from the Faculty of Computational Mathematics and Cybernetics at Lomonosov Moscow State University in 2002. In 2005, he completed his postgraduate studies at the same faculty. In 2008, he defended his dissertation for the degree of Candidate of Physical and Mathematical Sciences on the topic "Synthesis of Polynomials over Extremal Estimation Algorithms." From 2000 to the present, he has worked at the Computing Center of the Russian Academy of Sciences (later the Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences). His research interests include pattern recognition and data analysis. He has published and co-authored 139 scientific papers.

email: adokukin@frccsc.ru

ORCID: 0009-0001-9928-7851

Материал поступил в редакцию 3 ноября 2025 года

ПОСТ-КОРРЕКЦИЯ СЛАБОЙ РАСШИФРОВКИ БОЛЬШИМИ ЯЗЫКОВЫМИ МОДЕЛЯМИ В ИТЕРАЦИОННОМ ПРОЦЕССЕ РАСПОЗНАВАНИЯ РУКОПИСЕЙ

В. П. Зыков¹ [0009-0007-8935-9288], **Л. М. Местецкий**² [0000-0001-6387-167X]

^{1, 2}*Московский государственный университет имени М. В. Ломоносова,
г. Москва, Россия*

²*НИУ Высшая школа экономики, г. Москва, Россия*

¹zykovvp@my.msu.ru, ²mestlm@mail.ru

Аннотация

Рассмотрена задача ускорения построения точной редакторской разметки рукописных архивных текстов в рамках инкрементного цикла обучения на основе слабой расшифровки. В отличие от ранее опубликованных результатов, основное внимание уделено интеграции автоматической посткоррекции слабой расшифровки с помощью больших языковых моделей (Large Language Models, LLM). Предложен и реализован протокол применения LLM на уровне строк в режиме обучения на нескольких примерах с тщательно сконструированными промптами и контролем формата вывода (сохранение дореформенной орфографии, защита имен и числительных, запрет на изменение структуры строк). Эксперименты проведены на корпусе дневников А. В. Сухова-Кобылина. В качестве базовой модели распознавания использована строчная версия модели Vertical Attention Network. Результаты показали, что LLM-коррекция на примере сервиса ChatGPT-4o заметно улучшает читабельность слабой разметки и существенно снижает процент ошибок в словах (в нашем опыте – порядка –12 процентных пунктов), при этом не внося ухудшения в проценте ошибок в буквах. Другой исследуемый сервис – DeepSeek-R1 – показал менее стабильное поведение. Рассмотрены практические настройки промптов, ограничения (контекстные лимиты, риск «галлюцинаций») и даны рекомендации по безопасной интеграции LLM-коррекции в итерационный пайплайн разметки с целью сокращения трудозатрат эксперта-ассессора и ускорения оцифровки исторических архивов.

Ключевые слова: *распознавание рукописного текста, слабая разметка, Vertical Attention Network (VAN), большие языковые модели (LLM), посткоррекция, итерационное дообучение.*

ВВЕДЕНИЕ

Автоматическое распознавание рукописных текстов (handwritten text recognition, HTR) остается важной задачей цифровой гуманитаристики: расшифровка архивных дневников открывает доступ к уникальным историческим материалам и позволяет применять инструменты поиска и анализа текста в гуманитарных исследованиях. Одновременно исторические архивы имеют специфические трудности: встречаются дореформенная орфография, частые зачеркивания, многоязычные вставки, значительные различия в качестве сканирования и нелинейная кривизна строк, что делает прямое применение стандартных HTR-конвейеров малоэффективным без адаптации к конкретному корпусу.

В настоящей работе рассмотрен корпус рукописных дневников А. В. Сухова-Кобылина – важного исторического деятеля XIX в. [1]. Этот архив содержит порядка 10000 страниц, но экспертная разметка есть лишь для 92 страниц (~2876 полностью размеченных строк), что существенно ограничивает возможности обучения современных нейросетевых моделей. Разметка рукописных текстов XIX в. – это высокоспециализированная, крайне трудоемкая и время затратная работа, доступная только ограниченному кругу исследователей-гуманитариев, обладающих соответствующей филологической подготовкой. Поэтому мы хотим максимально снизить трудозатраты таких высококвалифицированных экспертов на разметку архивных рукописей.

Для подготовки входных данных мы использовали процедуры сегментации и нормализации строк, разработанные в наших предыдущих работах и реализованные в программном комплексе (см. [2, 3]). Эти алгоритмы позволяют выделять базовые линии, корректировать кривизну строк и приводить их к формату, пригодному для строчных моделей распознавания. В качестве базовой архитектуры распознавания бралась строчная версия архитектуры Vertical Attention Network (VAN) [4], т. к. она продемонстрировала конкурентоспособные результаты на открытых наборах

данных при распознавании параграфов и строк на дневнике Ф. П. Литке [5] и использовалась в предыдущей работе [3]. В выборе строчной версии VAN для данного архива ключевую роль сыграли две практические причины: возможность обучаться на отдельно размеченных строках, что важно при частично размеченных страницах; а также устойчивость к сложным нелинейным структурам строк, характерным для нашего корпуса.

Под термином «**слабая расшифровка (разметка)**» мы понимаем автоматическую расшифровку, полученную моделью распознавания на неразмеченных данных и содержащую значительное число ошибок (как на уровне символов, так и на уровне слов). Эксперт-ассессор исправляет ошибки в слабой расшифровке, получая таким образом новые высококачественные обучающие примеры.

Инкрементный (итерационный) подход наращивания размеченной выборки, подробно описанный в [3], состоит в цикле:

- обучение модели на имеющейся размеченной выборке;
- применение модели к неразмеченным строкам и получение слабой разметки;
- экспертная корректировка слабой разметки (получение точной редакторской разметки);
- дообучение модели на расширенной выборке и переход к новой итерации.

Такой подход позволяет снизить трудозатраты эксперта-ассессора по разметке рукописного текста, т. к. ему нужно лишь исправить ошибки в слабой расшифровке, а не размечать текст с нуля.

В настоящей работе основное внимание уделено интеграции автоматической посткоррекции слабой расшифровки с помощью больших языковых моделей (Large Language Models, LLM) в описанный итерационный цикл. Идея применения LLM в режиме обучения на нескольких примерах (few-shot learning) опирается на подход, описанный в [6], где показано, что LLM могут решать новые языковые задачи, получая только несколько примеров в промпте, без дополнительного дообучения. Мы покажем, что LLM в таком режиме при аккуратно сконструированных промптах и строгих правилах вывода (сохранение дореформенной орфографии, защита имен

и числительных, запрет на изменение структуры строк) способны устранять языковые и морфологические артефакты автоматических транскрипций и повышать читабельность слабой разметки. Описан также протокол применения LLM на уровне строк, проведено сравнение нескольких сервисов (ChatGPT-4o и DeepSeek-R1) и показано, что при корректной настройке промптов LLM может существенно снизить процент ошибок в буквах и уменьшить объем ручных правок, требуемых от эксперта (см. табл. 2).

ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

Область оффлайн-распознавания рукописного текста (HTR) эволюционировала от классических статистических и рекуррентных схем к современным архитектурам с механизмами внимания и трансформерам. Значительный вклад в стандартизацию исследований внесли общедоступные корпуса и соревнования (например, IAM и READ), которые служат основой для сопоставления методов и оценки прогресса [7, 8]. Традиционно в прикладных решениях HTR выделяют два подхода. Первый – это строчный (line-level): страницы предварительно сегментируют на строки одной моделью или алгоритмом, и каждая строка распознается другой моделью. Второй – это страничный (page-level), когда модель принимает на вход большие области текста и генерирует последовательность символов без явной построчной разметки. Классические строчные схемы часто строят на комбинации сверточных блоков и рекуррентных нейронных сетей (recurrent neural networks, RNN) [9], обучение которых происходит через CTC (connectionist temporal classification) – это метод обучения и соответствующая функция потерь, позволяющие обучать модели на последовательных данных без явной разметки по позициям символов [10]. Современные страничные архитектуры (SPAN, OrigamiNet, TrOCR, VAN и др.) включают различные способы формирования 2D-представлений и применения трансформеров с предобученными компонентами [4, 11–13].

Для исторических архивов выбор подхода определяется практическими ограничениями: страничные модели облегчают требования к разметке и лучше учитывают контекст, но предъявляют более строгие требования к форматной стабильности изображений и объему размеченных данных; строчные решения,

напротив, оказываются более гибкими при частично размеченных страницах и сложной геометрии строк, что сделало их предпочтительными в ряде прикладных проектов [4, 7, 8]. Практический опыт проектов по цифровизации (включая Digital Peter [14]) подтверждает, что для исторических данных необходима комбинация надежной сегментации, адаптированных архитектур распознавания и продуманной организационной процедуры разметки.

Ключевым предварительным этапом любого HTR-пайплайна остаются сегментация и нормализация строк: ошибки на этом шаге часто «ломают» последующую обработку, поэтому прикладные системы включают как автоматические алгоритмы детекции и выпрямления строк, так и инструменты ручной корректировки сегментации и аннотаций [2, 8]. При ограниченной экспертной разметке («малые данные») классические подходы по сбору больших размеченных корпусов оказываются нереализуемыми, что стимулирует развитие методов слабой разметки, полу- и самообучения, а также практик итеративного (инкрементного) наращивания размеченной выборки: автоматическая расшифровка применяется к новым данным, затем эксперт правит полученную слабую разметку и модель дообучается на расширенной выборке [3, 14].

Недавний рост качества языковых моделей открывает дополнительное перспективное направление: использование LLM для постобработки автоматических расшифровок. LLM в режиме обучения на нескольких примерах могут исправлять языковые и морфологические артефакты, восстанавливать корректные словоформы и делать выводы о вероятных восстановленных фрагментах текста, что особенно полезно, когда ошибки имеют лингвистический, а не оптический характер. При аккуратной инженерии промптов и ограничениях формата вывода LLM способны сохранять историко-филологические особенности (например, дореформенную орфографию) и тем самым сокращать долю ручной правки со стороны эксперта-ассессора [6].

Таким образом, опираясь на наработки по надежной сегментации и организационным процедурам разметки [2, 3], а также на современные достижения в области строчных и страничных архитектур [4, 11–13], целесообразно исследовать практическую интеграцию LLM-коррекции слабых расшифровок в итерационный

пайплайн. Это направление сочетает преимущества автоматического предобработанного вывода компьютерного зрения и языковой постобработки, что потенциально может позволить существенно снизить трудозатраты экспертов и ускорить оцифровку исторических архивов.

ПОСТАНОВКА ЗАДАЧИ

В нашем распоряжении имеется архив рукописных дневников А. В. Сухово-Кобылина, содержащий порядка 10000 сканов страниц. Для части архива доступна экспертная разметка: 92 страницы, что соответствует примерно 2876 полностью размеченным строкам (в текстовых файлах, по одному файлу разметки на страницу). Остальные страницы представлены в виде изображений (рис. 1) и требуют автоматической расшифровки.

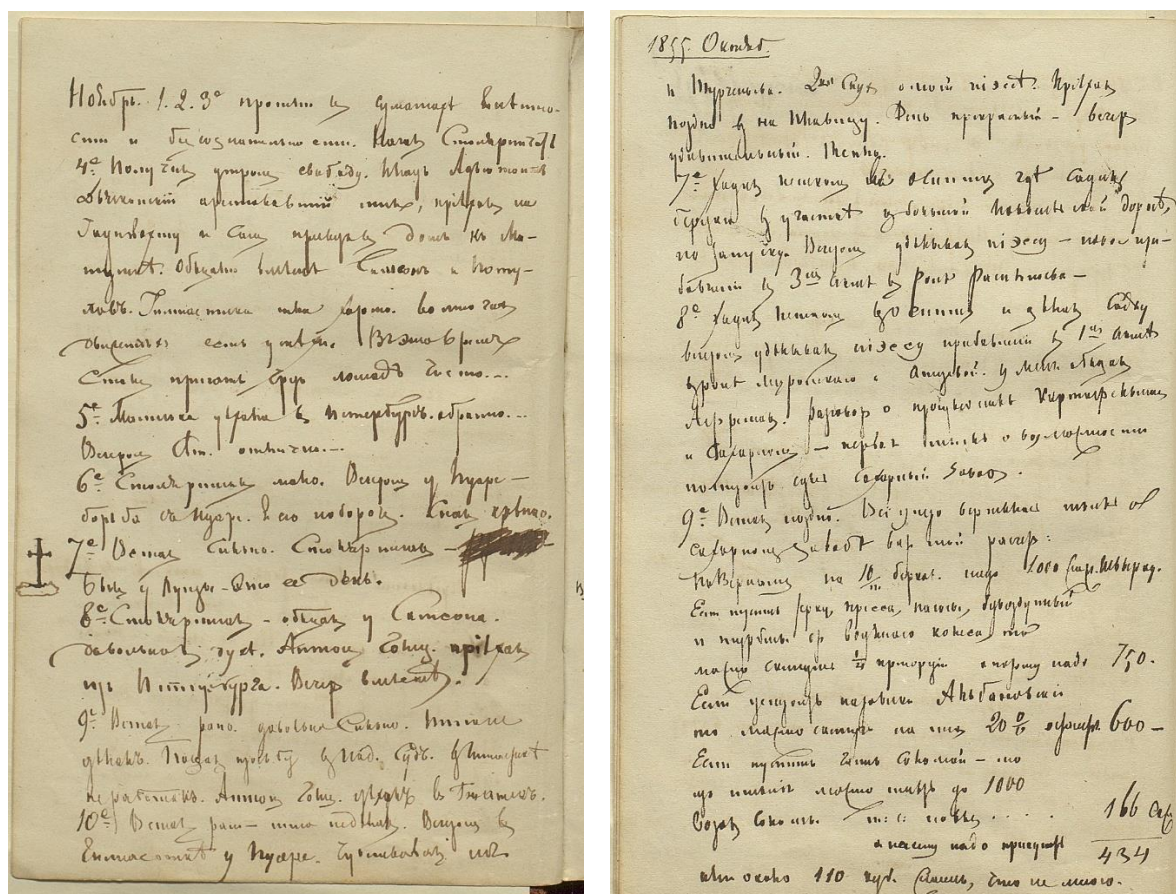


Рис. 1. Примеры страниц дневника А. В. Сухово-Кобылина.

Структура и качество исходных данных характеризуются следующими особенностями, которые определяют практические ограничения задачи:

- сканы различаются по разрешению и качеству, содержат артефакты сканирования;
- тексты написаны дореформенной орфографией XIX в.; присутствуют зачеркивания и вставки на других языках (например на французском);
- строки на странице часто имеют нелинейную (изогнутую) форму; часть строк может быть частично размечена или вовсе не читаема; в случаях явной неразборчивости эксперт отмечает фрагменты специальными маркерами на такие фрагменты и строки мы считаем неразмеченными и excluим из обучения;
- для подготовки строчной подачи используются алгоритмы сегментации и нормализации строк, описанные в [2, 3]; для распознавания в качестве базовой архитектуры используется строчная версия VAN [4].

Необходимо разработать метод и программную процедуру для автоматической расшифровки неразмеченных страниц с учетом следующих требований.

- Метод должен работать в инкрементном режиме: поддерживать цикл «обучение -> применение -> получение слабой разметки -> экспертная корректировка -> дообучение». Детали организации изложены в [3].
- Метод должен поддерживать интеграцию автоматической LLM-коррекции слабой расшифровки перед этапом экспертной правки (промпт-настройки, сохранение дореформенной орфографии и ограничение на изменение структуры строк). Цель – снизить объем ручных правок, требуемых от эксперта.
- Сохранение историко-филологической корректности: метод не должен автоматически «нормализовывать» дореформенную орфографию без явного указания.

В качестве основного критерия качества распознавания мы будем использовать CER (character error rate) – процент ошибок в буквах. Он рассчитывается следующим образом:

$$\text{CER}(y_{\text{true}}, y_{\text{pred}}) = \frac{\rho_{\text{Lev}}(y_{\text{true}}, y_{\text{pred}})}{\text{length}(y_{\text{true}})} \cdot 100\%,$$

где $\rho_{\text{Lev}}(y_{\text{true}}, y_{\text{pred}})$ – расстояние Левенштейна между правильной экспертной разметкой (y_{true}) и расшифровкой модели (y_{pred}), при этом все строки, на которых подсчитывается указанный критерий, считаются одним цельным текстом.

В качестве вспомогательного критерия качества будем использовать WER (word error rate) – процент ошибок в словах:

$$\text{WER}(y_{\text{true}}, y_{\text{pred}}) = \frac{\tilde{\rho}_{\text{Lev}}(y_{\text{true}}, y_{\text{pred}})}{\text{word_length}(y_{\text{true}})} \cdot 100\%,$$

в этой метрике подсчет расстояния Левенштейна ($\tilde{\rho}_{\text{Lev}}$) и длины строки (word_length) происходит на уровне слов, а не отдельных символов.

Необходимо построить и протестировать алгоритм/пайплайн, который при указанных входах и ограничениях минимизирует CER и WER на отложенной тестовой выборке, а также демонстрирует экономию экспертного времени и количества правок при использовании LLM-коррекции.

ПРЕДЛАГАЕМЫЙ ПОДХОД К РАСПОЗНАВАНИЮ

Архитектура сети

Для распознавания рукописного текста нами была использована модель Vertical Attention Network (VAN), описанная в [4]. Эта модель была выбрана, потому что показывает конкурентоспособные результаты на открытых датасетах, хорошо себя зарекомендовала при распознавании дневника Ф. П. Литке [5] и уже использовалась в предыдущей нашей работе [3].

Эта модель имеет две разновидности: страничную и строчную. Страничная модель принимает на вход изображение всей страницы рукописного текста целиком, а строчная работает на уровне отдельных строк.

Строчная архитектура представлена полностью сверточной нейронной сетью, состоящей из кодировщика, содержащего 10 блоков с 3 свертками в каждом

(рис. 2), и сверточного декодировщика, который переводит внутреннее представление модели в набор вероятностей. Архитектура строчной модели схематично представлена на рис. 3.

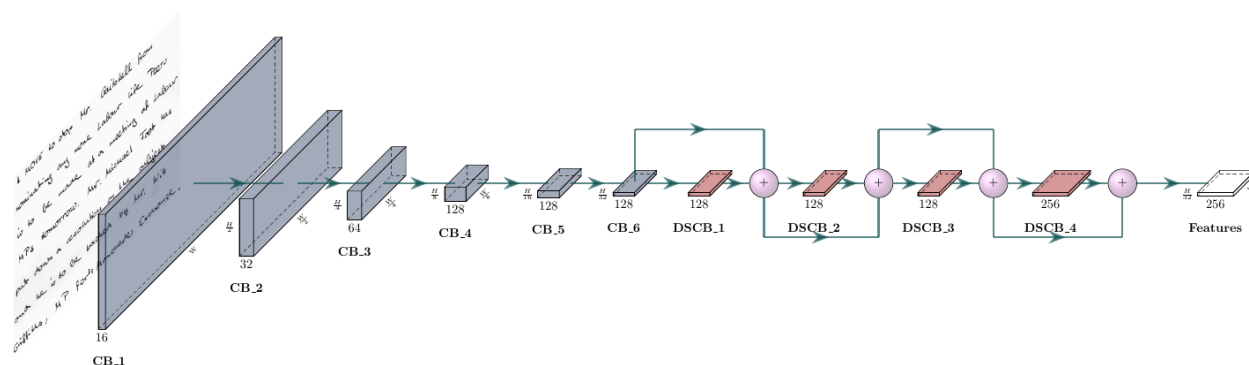


Рис. 2. Кодировщик VAN (одинаковый для страничной и строчной архитектур).

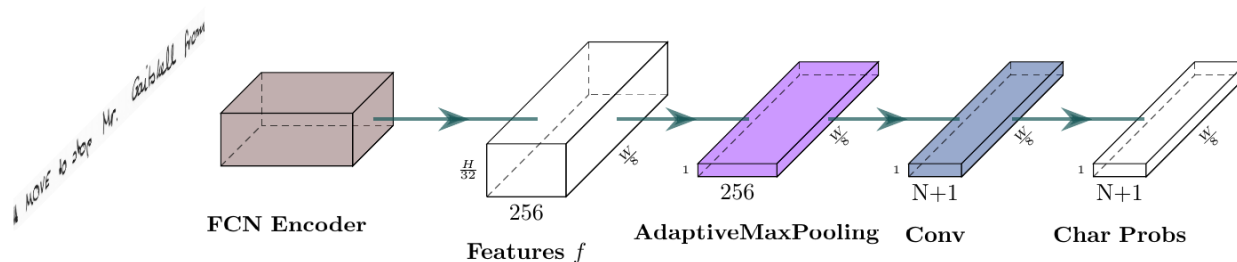


Рис. 3. Строчная архитектура VAN.

Страничная архитектура имеет такой же кодировщик, как и строчная. У страничного варианта модели, в отличие от строчного, есть модуль вертикального внимания (vertical attention), который содержит несколько полносвязных слоев и позволяет рекуррентно собирать представления строк на странице, которые затем приводятся к матрицам вероятностей декодировщиком, содержащим LSTM-слой и сверточный слой 1 x 1. Страничная архитектура изображена на рис. 4. Достоинство страничной модели заключается в том, что она не требует предварительной сегментации строк на странице, но при этом предполагает, что строки текста являются линейными и ровными.

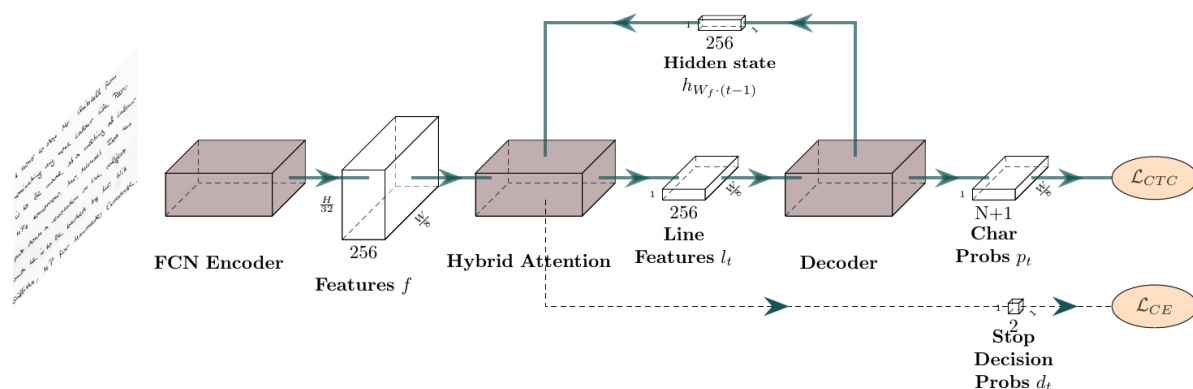


Рис. 4. Страничная архитектура VAN.

Обе модели выдают вероятности символов в строках; расшифровка выполняется жадным декодированием по выходам CTC, а обучение – с использованием функции потерь CTC (Connectionist Temporal Classification) [10]. Таким образом, при обучении решается задача оптимизации

$$\hat{\theta} = \operatorname{argmin}_{\theta} \mathcal{L}_{\text{CTC}}(y, \hat{y}(\theta)),$$

где y – правильная последовательность символов, $\hat{y}(\theta)$ – предсказанная последовательность модели с параметрами θ .

В настоящей работе мы использовали строчный вариант модели VAN. Применительно к рассматриваемой задаче строчная модель имеет несколько преимуществ.

Во-первых, некоторые строки на рукописной странице не удастся разобрать и разметить даже эксперту (или разметка этих строк для него просто не представляет научного интереса), но при этом остальные строки на той же странице хорошо читаются. В этом случае непонятно, как использовать такую не до конца размеченную страницу для обучения страничной модели. Строчная же модель такой проблемы не имеет, так как использует для обучения отдельные размеченные строки.

Во-вторых, страничная модель не всегда справляется с проблемой сегментации строк в том случае, когда они имеют сложную криволинейную структуру. Строчная модель может в этом случае игнорировать плохо сегментированные строки на изображении страницы.

В-третьих, с точки зрения эксперта-ассессора страница является слишком большим фрагментом текста, эксперт в процессе получения подстрочного перевода оперирует понятиями строк и отдельных слов.

Изначально параметры кодировщика будем инициализировать предобученными весами на англоязычном датасете IAM [7].

Сегментация и нормализация строк

Для используемой нами строчной модели необходима предварительная сегментация изображений страниц на отдельные строки. Эта процедура выполняется с помощью программы «Подстрочник». Описание программы и используемого в ней метода представлены в [2, 3].

Алгоритм сегментации включает следующие шаги.

- Изображение страницы переводится в полутоновое (серое) изображение.
- Для каждой строки изображения (на уровне пикселей) вычисляется сумма яркостей пикселей в этой строке. Получаем функцию, зависящую от ординаты. Эта функция имеет форму синусоиды, минимумы которой соответствуют позициям текстовых строк.
- Страницу можно разделить на сегменты вертикальными линиями и для каждого сегмента выполнить данную операцию, чтобы учесть нелинейность строк.
- Полученные базовые линии строк на разных сегментах объединяются в виде ломаных линий, описывающих положение строк на изображении.
- Далее от базовых линий отступают на некоторое количество пикселей вверх и вниз, чтобы выделить ограничивающие прямоугольники для каждой текстовой строки.
- После этого выполняется нормализация строк – ломаные линии выпрямляются с помощью сдвига значений в столбцах изображения, порождая выпрямленные строки для подачи в строчную модель.

На рис. 5 показаны пример исходного изображения страницы и результат выделения базовых линий строк с использованием программы «Подстрочник».

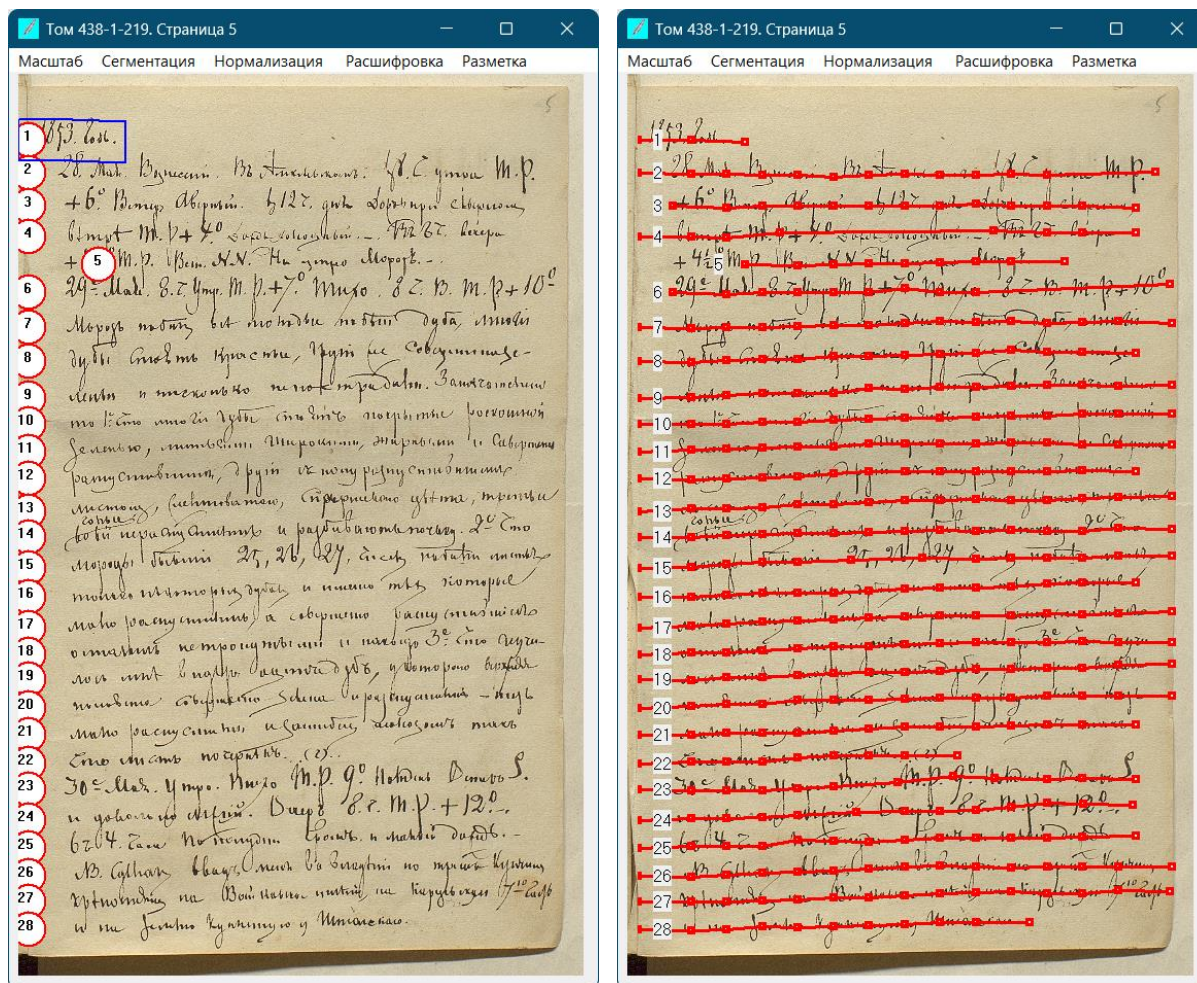


Рис. 5. Исходное изображение страницы, поданное в программу (слева), и изображение с выделенными базовыми линиями строк (справа).

После выделения линий строк выполняется нормализация, в ходе которой строки выпрямляются. На рис. 6 показаны исходная строка и ее нормализованный вид. После этого обработанные строки можно подавать в строчную модель.

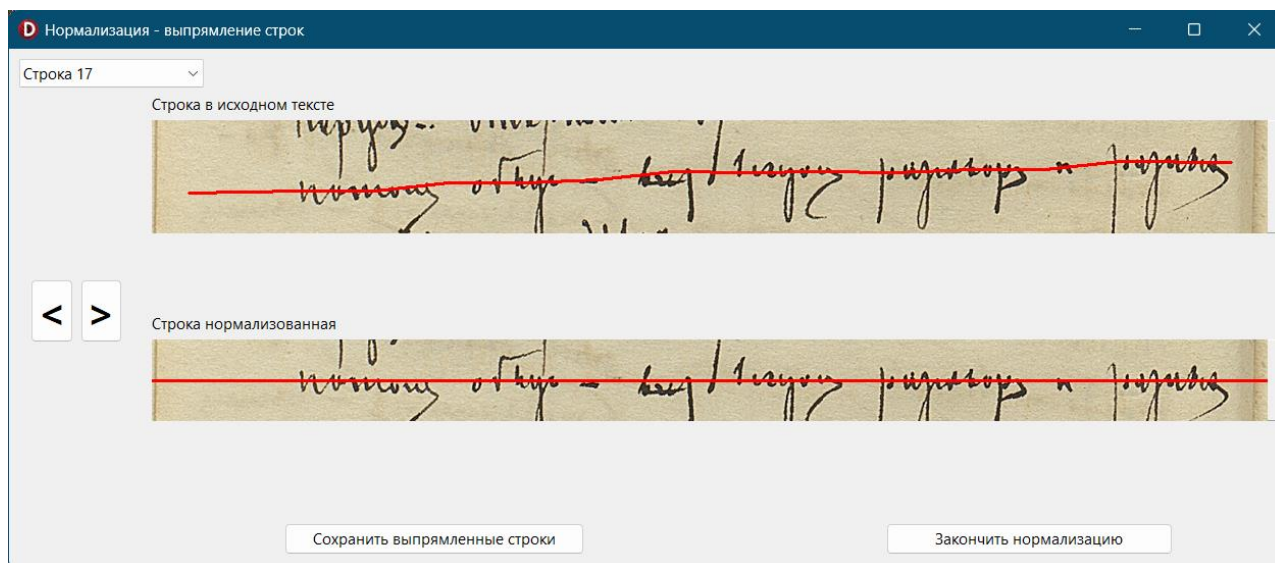


Рис. 6. Нормализация строки: исходная и выпрямленная строка.

Итерационный процесс наращивания обучающей выборки и дообучения модели

Разметка рукописных текстов на русском языке XIX в. является сложной задачей, которую может выполнить только небольшой круг экспертов. Вследствие этого изначально мы имеем довольно небольшое число полностью размеченных строк для обучения нейронной сети.

Предполагается, что увеличить качество распознавания можно путем наращивания обучающей выборки.

Чтобы облегчить эксперту работу по разметке (для увеличения обучающей выборки), мы применили обученную модель на новых, еще неразмеченных рукописных текстах. Полученную после распознавания расшифровку мы называем **слабой разметкой** (или **слабой расшифровкой**). Слабая разметка содержит довольно большое число ошибок. Далее эксперту нужно лишь исправить эти ошибки, чтобы получить редакторскую разметку. Это сделать существенно проще, чем с нуля разметить рукописный текст.

После получения новой порции размеченных данных происходит обучение модели на увеличенной выборке. Это, по предположению, должно увеличить качество распознавания обученного алгоритма. Далее улучшенный алгоритм

можно опять применить на новых данных, после чего эксперту нужно будет исправить меньшее число ошибок по сравнению с предыдущей итерацией. Таким образом, получаем итерационный процесс наращивания обучающей выборки и дообучения модели с увеличением качества распознавания на каждой такой итерации. Этот процесс схематично показан на рис. 7.

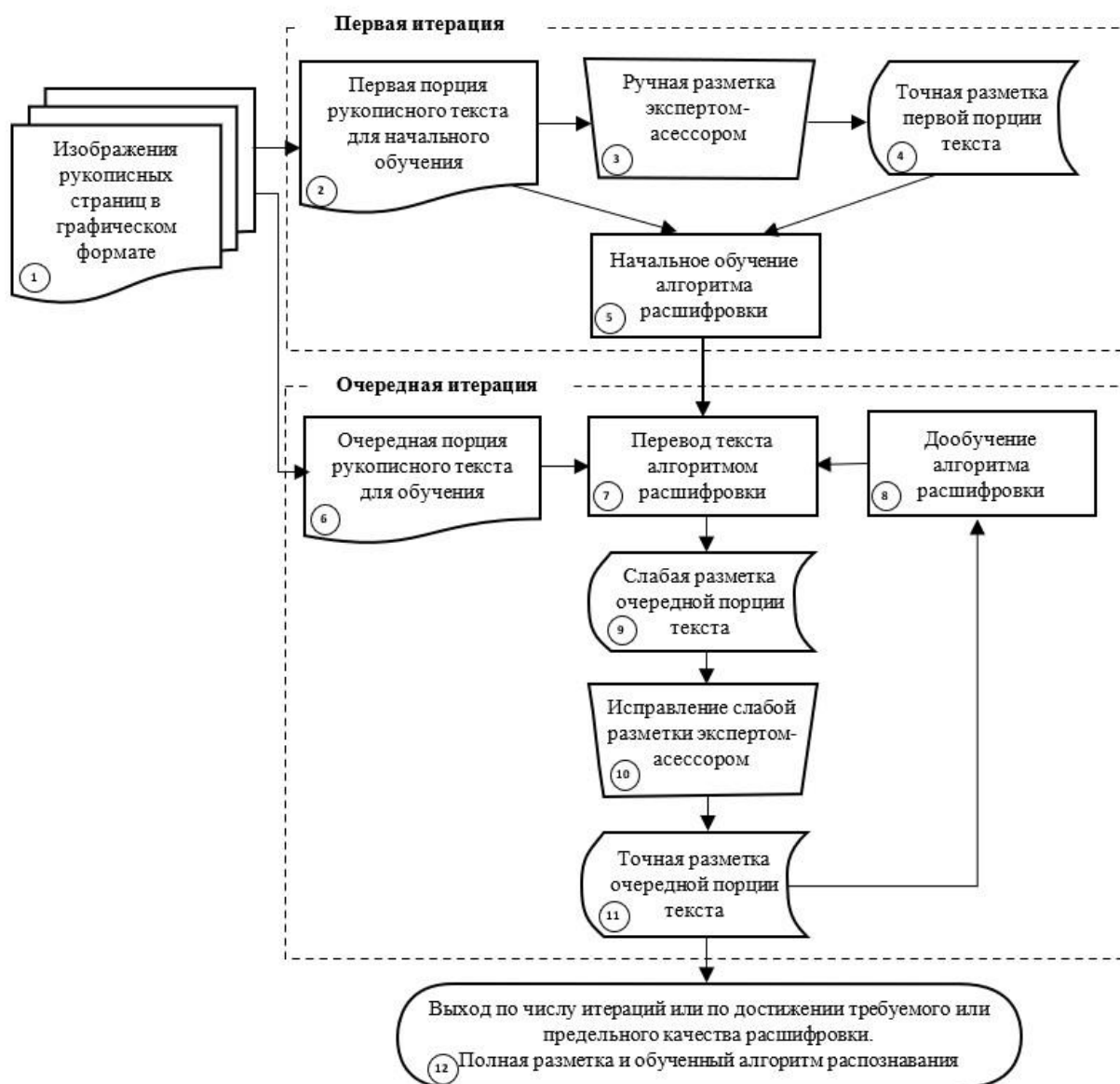


Рис. 7. Схема итерационного процесса распознавания.

Кроме того, работу эксперта упрощает программа «Подстрочник» с графическим интерфейсом. Она позволяет выбрать интересующую эксперта страницу дневника (рис. 8), сделать автоматическую сегментацию строк (рис. 5), исправить

полученную сегментацию в случае обнаружения в ней ошибок, нормализовать строки (рис. 6), применить заранее обученную нейронную сеть для получения автоматической расшифровки модели и отредактировать слабую расшифровку, создав тем самым размеченную для обучения строку (рис. 9).

Более подробное описание программы «Подстрочник», данного итерационного процесса и его экспериментальной корректности дано в [3].

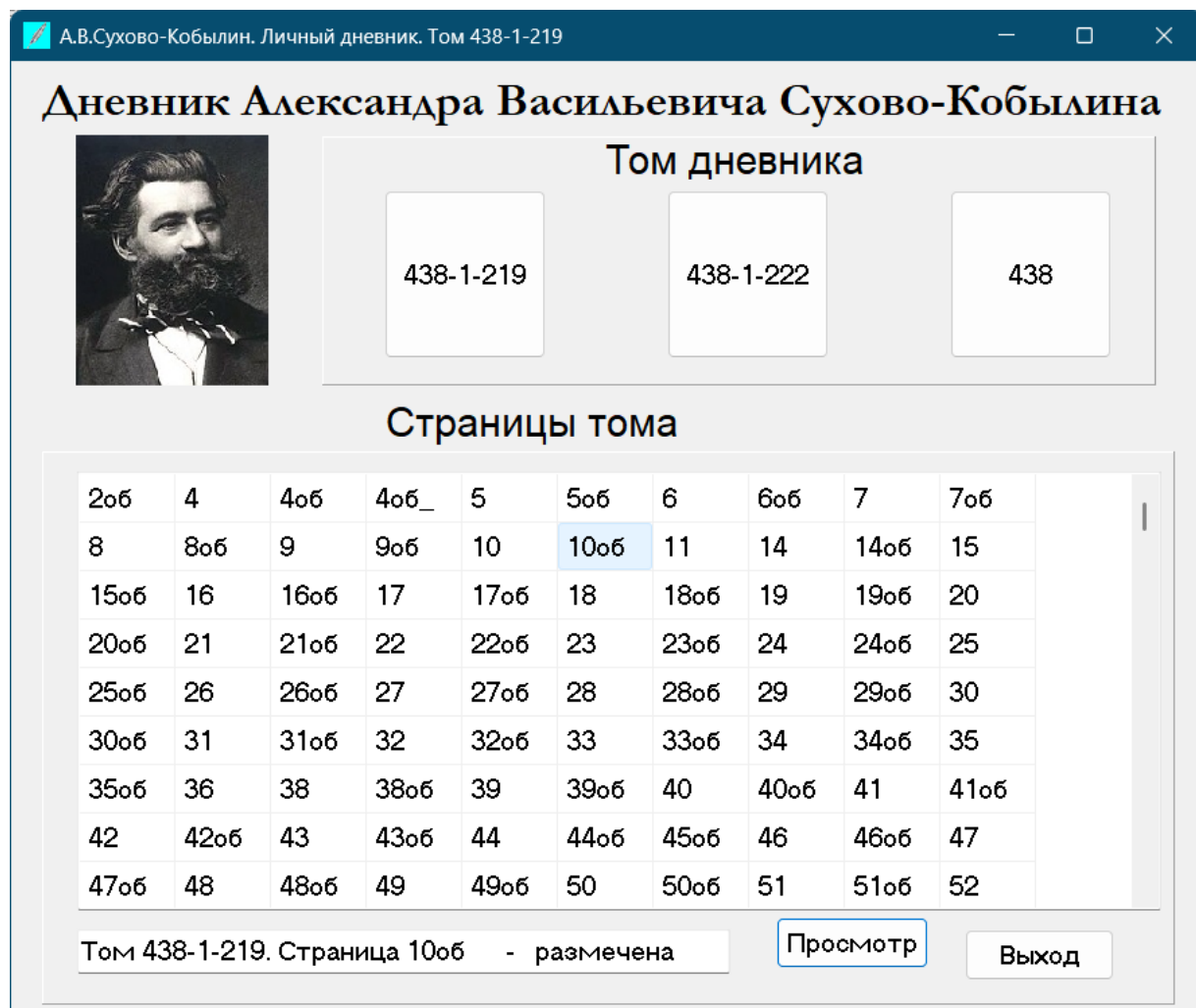


Рис. 8. Программа «Подстрочник»: выбор страницы.

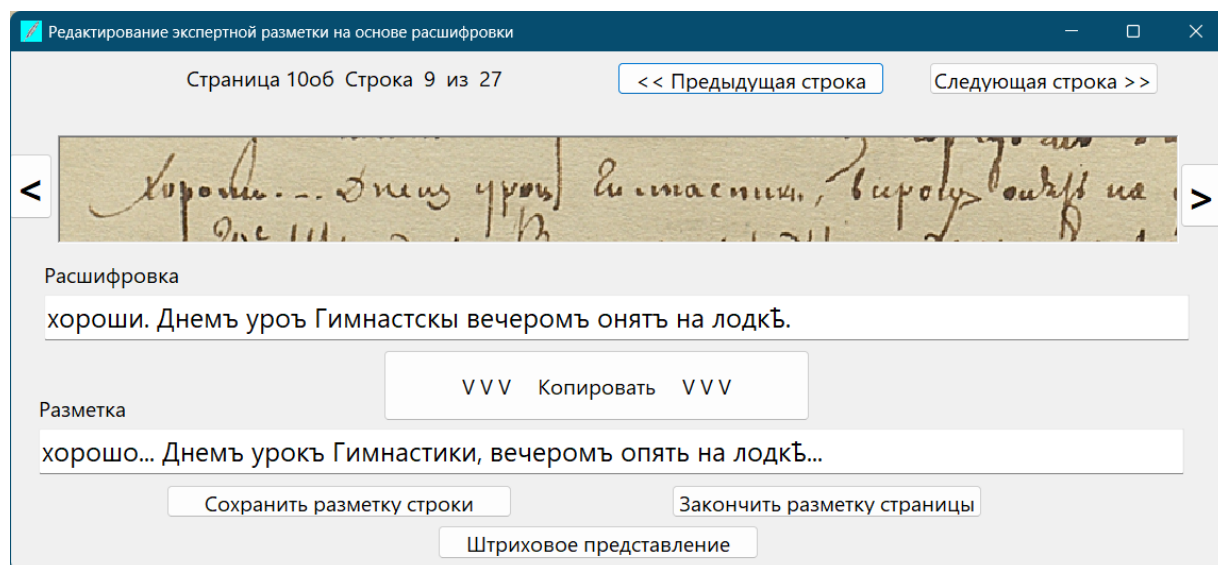


Рис. 9. Программа «Подстрочник»: редактирование расшифровки модели.

ЭКСПЕРИМЕНТЫ

Исходное обучение на архиве А. В. Сухово-Кобылина

В этом эксперименте происходит обучение модели на данных дневника А. В. Сухово-Кобылина. Тренировочная выборка составлена из 1599 строк, валидационная – из 119 строк, тестовая – из 74 строк. Остальные 1084 строки (из указанных в постановке задачи 2876 строк) были не размечены в момент проведения эксперимента.

Для ускорения сходимости модель инициализируется предобученными весами. Для инициализации весов кодировщика использовались веса, полученные при обучении на датасете IAM, а декодировщик обучался с нуля, поскольку алфавит у этих датасетов отличается (в разных датасетах используются разные множества символов).

Применялись аугментации, предложенные в статье про VAN [4]. Был использован `batch_size = 128`. Остальные гиперпараметры обучения остались такими же, как в [4]. Модель обучалась в течение 8152 эпох, что на видеокарте NVIDIA A100 80GB заняло 1 сутки. Лучшее качество на валидации наблюдалось на 7802-й эпохе.

Оценки качества на эпохе (7802), лучшей с точки зрения CER на валидации, представлены в табл. 1.

Табл. 1. Полученные метрики при исходном обучении

Выборка	CER (%)	WER (%)
Train	3.52	10.80
Valid.	17.74	53.73
Test	15.93	50.64

Примеры работы модели на тестовой выборке показаны на рис. 10. Видно, что даже при текущем уровне ошибок расшифрованный текст оказывается вполне читаемым и по нему можно понять смысл написанного.

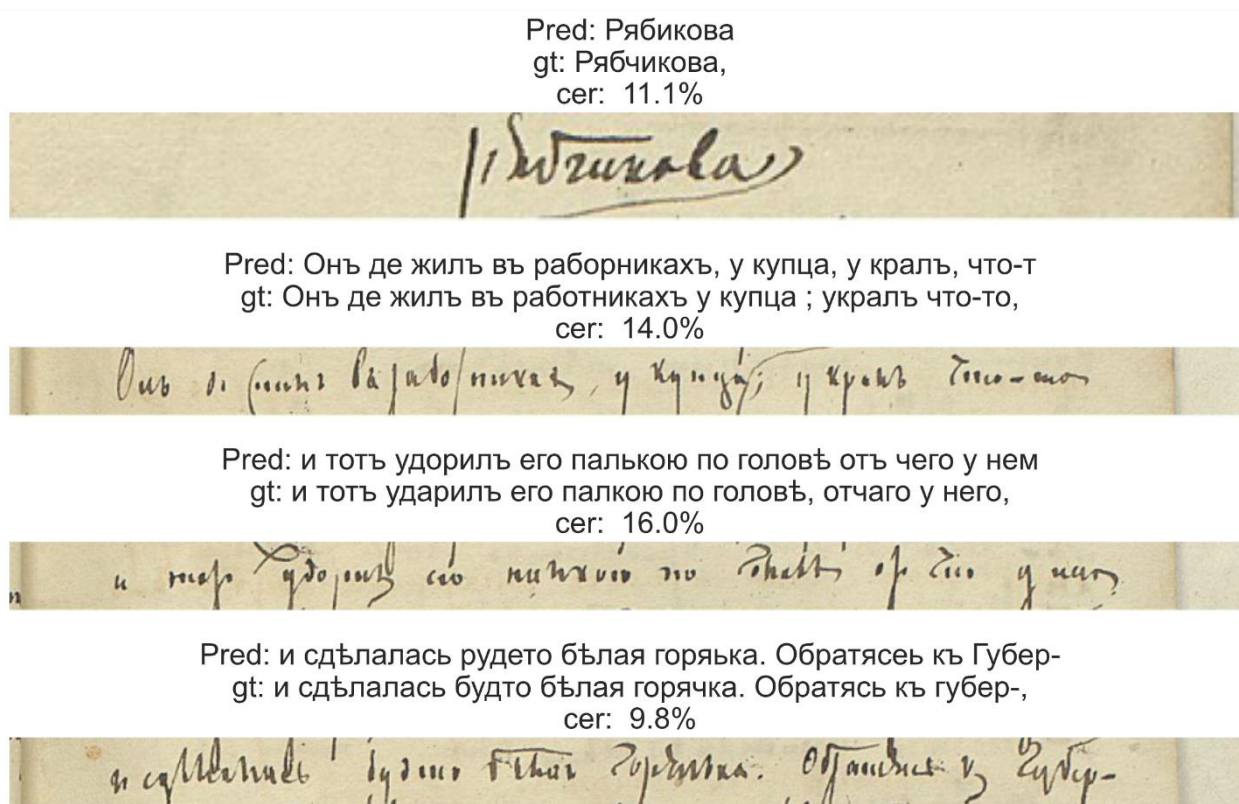


Рис. 10. Примеры работы модели на тестовых данных: Pred – предсказание модели, gt – разметка эксперта.

Однако на тестовых данных ошибки все-таки оказываются довольно большими: CER = 15.93%, WER = 50.64%. Число этих ошибок может уменьшить их исправление с помощью больших языковых моделей, о чем речь пойдет далее.

Исправление ошибок с помощью больших языковых моделей

Цель проведенного эксперимента – проверить гипотезу о том, что большие языковые модели способны повысить качество автоматических расшифровок строк, получаемых строчной моделью VAN, и тем самым снизить объем ручных правок, требующихся от эксперт-ассессора в рамках итерационного цикла.

Исправление выполнялось на уровне отдельных строк (коррекция целых строк), чтобы не изменять количество строк на странице, это важно для интеграции в рабочий интерфейс и последующего дообучения. В эксперименте использовались две LLM: ChatGPT-4o и DeepSeek-R1. В качестве подхода применялся режим обучения на нескольких примерах: в системный промпт включались примеры пар «исходная строка -> исправление» из валидационной выборки (119 примеров). Для ChatGPT дополнительно были поданы 250 примеров из тренировочной выборки; у DeepSeek-R1 технически не удалось разместить столь большой контекст примеров. Ключевые инструкции промпта были такие: сохранять исходную последовательность слов и структуру строк, не выполнять нежелательную нормализацию дореформенной орфографии, не изменять имена собственные и числительные, исправлять только явные опечатки и ошибки распознавания. Пример системного промпта:

«Твоя задача – корректировать входной текст, исправляя в нем ошибки.

Это текст, распознанный моделью компьютерного зрения с рукописей. Рукописи написаны на русском языке 19-го века (также иногда присутствуют другие языки).

Цель – получить максимально близкий к рукописи вариант расшифровки.

Модель допускает ошибки в распознавании символов.

Исправляй только самые явные и понятные места. Если фрагмент текста сложно разобрать, то сохраняй его в том же виде, в котором и получил.

Сохраняй имена собственные и числительные как есть. Сохраняй исходную последовательность слов.

Тебе будут подаваться строки, распознанные моделью компьютерного зрения, к которым в конце добавлены символы ``->". Тебе нужно вернуть исправленную строку.

Примеры:

1856. Мартъ -> 1856. Мартъ

3-е. Въ часовъ прїѣхалъ въ Канугу. Дядѣ принялъ меня -> 3е. Въ 11 часовъ прїѣхалъ въ Калугу. Дядя принялъ меня

лучше. Стотрѣли иланхъ моего завода – онъ далъ мнѣ -> лучше. Смотрѣли планы моего Завода – онъ далъ мнѣ

Отдалъ перепиывать піэсу въ Печать. -> Отдалъ переписывать піэсу въ печать.

<Остальные примеры ...>»

Был использован режим сохранения истории: при последовательной правке строк предыдущие исправления и ответы ассистента оставались в контексте, чтобы LLM получала дополнительный локальный контекст и лучше понимала смысл соседних строк.

На рис. 11 и 12 представлены примеры исправлений с помощью указанных больших языковых моделей.

В табл. 2 приведены показатели CER и WER на тестовых данных после исправления большими языковыми моделями, а также изменения метрик по сравнению с отсутствием исправлений (предыдущий эксперимент). Из таблицы видно, что ChatGPT оставляет CER примерно таким же, но при этом значительно снижает процент ошибок в словах ($\Delta WER = -12.27\%$). DeepSeek же работает в этом случае хуже, он часто выдает больше одной строки, поэтому он увеличивает CER, что является неприемлемым; WER он уменьшает не так сильно, как ChatGPT. Из-за ограничения на размер промпта DeepSeek-R1 принял только 119 примеров исправлений из валидационной выборки, тогда как для ChatGPT-4o в системный промпт дополнительно было включено ещё 250 примеров из тренировочной выборки. Таким образом, использование ChatGPT позволило значительно снизить ошибки модели компьютерного зрения (строчной VAN), а использование DeepSeek не принесло положительных результатов.

Model: Самояло ва презнательные А.В. Сухово.– Ко- (ser = 15.38%)
ChatGPT: Самойлова признательные А. В. Сухово-Ко- (ser = 2.56%)
Human: Самойлова признательные А.В. Сухово-Ко-

Model: 28-е. Сборъ на Выксу. Проицанные съ Самсономъ. (ser = 11.63%)
ChatGPT: 28е. Сборъ на Выксу. Прощанье съ Самсономъ. (ser = 4.65%)
Human: 28е. Сборы на Выксу. Прощаніе съ Самсономъ.

Model: продалъ d пудъ патоки по 180 к. сереб. и п.р. ереб. (ser = 27.59%)
ChatGPT: продалъ 2 пуда патоки по 180 к. сереб. и 1 р. сереб. (ser = 22.41%)
Human: Продалъ 1 т. пудов патоки по 1.80 к. сер. и 1 т. р. сереб.

Model: вшесь Рертеру въ упламу за аппараты Къ сахар (ser = 14.89%)
ChatGPT: внёсь Рейтеру въ уплату за аппараты къ сахару (ser = 12.77%)
Human: внесь Ферстеру въ уплату за аппараты къ сахаръ-

Рис. 11. Примеры работы ChatGPT: исходные строки модели VAN (Model), исправленные ChatGPT строки (ChatGPT) и разметка эксперта (Human).

Model: Самояло ва презнательные А.В. Сухово.– Ко- (ser = 15.38%)
DeepSeek: Самоялову. Признательные А. В. Сухово. – Ко- (ser = 23.08%)
Human: Самойлова признательные А.В. Сухово-Ко-

Model: 28-е. Сборъ на Выксу. Проицанные съ Самсономъ. (ser = 11.63%)
DeepSeek: 28-е. Сборъ на Выксу. Прощаніе съ Самсономъ. (ser = 4.65%)
Human: 28е. Сборы на Выксу. Прощаніе съ Самсономъ.

Model: продалъ d пудъ патоки по 180 к. сереб. и п.р. ереб. (ser = 27.59%)
DeepSeek: продалъ 10 пудъ патоки по 180 к. сереб. и 5 р. сереб. (ser = 22.41%)
Human: Продалъ 1 т. пудов патоки по 1.80 к. сер. и 1 т. р. сереб.

Model: вшесь Рертеру въ упламу за аппараты Къ сахар (ser = 14.89%)
DeepSeek: въ счетъ Ретерту въ уплату за аппараты. Къ сахар- (ser = 25.53%)
Human: внесь Ферстеру въ уплату за аппараты къ сахаръ-

Рис. 12. Примеры работы DeepSeek: исходные строки модели VAN (Model), исправленные DeepSeek строки (DeepSeek) и разметка эксперта (Human).

Табл. 2. Сравнение метрик CER и WER после корректировки расшифровки тестовых данных с помощью ChatGPT-4o и DeepSeek-R1

Модель	CER (%)	Δ CER (%)	WER (%)	Δ WER (%)
ChatGPT-4o	15.87	−0.06	38.37	−12.27
DeepSeek-R1	17.39	+1.46	42.08	−8.56

Вероятно, LLM не специализированы на рукописи русского языка XIX в. и вряд ли в явном виде обучались на больших корпусах дореформенной орфографии; тем не менее при соответствующем промптинге ChatGPT в эксперименте продемонстрировал способность корректно обрабатывать такие формы и в большинстве случаев сохранять дореформенную орфографию (то есть не нормализовать текст к современному правописанию). Это указывает на то, что аккуратная инженерия промпта и ограничение правок (жесткие правила) позволяют эффективно применять LLM для посткоррекции исторических транскриптов.

Полученные результаты показали, что LLM-коррекция может быть полезным встроенным этапом итерационного пайплайна: автоматическая посткоррекция повышает читабельность слабой разметки и уменьшает долю очевидных ошибок, которые эксперту требуется исправить вручную. В частности, значительное снижение WER у ChatGPT подразумевает сокращение числа словесных правок и, как следствие, потенциальную экономию времени эксперта при создании точной редакторской разметки. При этом важно сохранять контроль поведения LLM (четкие промпты, лимиты выдачи, постфилтрация), поскольку некоторые модели могут изменять структуру строки или «предполагать» недостающие фрагменты – поведение, неприемлемое для автоматического включения в обучающий пул без ручной проверки.

В текущей реализации наблюдаются следующие ограничения: размер контекстного промпта влияет на качество – возможность подать больше примеров (как у ChatGPT) дает преимущество; некоторые LLM склонны к перестроению строки (изменению длины/формата), что плохо сочетается со строчным пайплай-

ном; риск «галлюцинаций» и нежелательной нормализации требует строгих правил промпта и постфильтров. Исходя из этого, для практической интеграции LLM в итерационный цикл следует: применять LLM-коррекцию на уровне строк с жестким контролем формата вывода; использовать исторический контекст для повышения согласованности исправлений; добавлять этап автоматической валидации результатов LLM (проверка длины, допустимых символов, сохранение имен и чисел) перед передачей строк эксперту.

В целом эксперимент подтвердил, что LLM-коррекция может быть эффективно использована в пайплайне инкрементной разметки: при аккуратной инженерии промптов и фильтрации выводов она повышает читабельность слабой разметки и убирает часть ошибок, которые в противном случае потребовали бы ручной правки.

ЗАКЛЮЧЕНИЕ

Предложен и экспериментально исследован практический подход к ускорению построения точной редакторской разметки исторических рукописей в рамках инкрементного (итерационного) цикла: основная новация – это интеграция автоматической посткоррекции получаемой «слабой» расшифровки с помощью больших языковых моделей. В качестве базового распознавателя использовалась строчная версия Vertical Attention Network (VAN), обученная на размеченных строках корпуса дневников А. В. Сухова-Кобылина; исходные метрики на тестовой выборке составили CER $\approx 15.93\%$ и WER $\approx 50.64\%$.

Экспериментальная оценка показала, что LLM-коррекция при аккуратной инженерии промптов и контроле формата вывода действительно уменьшает долю ошибок на уровне слов. Наиболее стабильный выигрыш продемонстрировал ChatGPT-4o: WER на тесте снизился примерно на 12.3 процентных пункта (до 38.37%) при практически неизменном CER ($\approx 15.87\%$). Другой протестированный сервис (DeepSeek-R1) показал менее приемлемое поведение: снижение WER сопровождалось ростом CER и неоднократным «перестроением» строк, что делает автоматическое включение таких исправлений в обучающий пул рискованным без ручной валидации.

Практическая значимость полученных результатов состоит в следующем. Автоматическая LLM-коррекция повышает читабельность слабой разметки и уменьшает объем явных ошибок в словах, которые эксперт-ассессор должен исправлять вручную; это потенциально приводит к заметной экономии времени эксперта в процессе итерационного наращивания размеченной выборки. Для безопасной и воспроизводимой интеграции LLM в пайплайн необходимо соблюдать ряд мер предосторожности: жесткую спецификацию промптов (с сохранением дореформенной орфографии, защитой имен и числительных и запретом на изменение структуры строк), использование исторического контекста для повышения согласованности, автоматическая поствалидация вывода (проверки длины / набора символов / сохранения ключевых токенов).

Ограничения проведенной работы также очевидны и важны для интерпретации результатов: зависимость качества от выбора и размера примеров в промпте (context length), стоимость и вопросы приватности при использовании коммерческих LLM, риск «галлюцинаций» и перестроения строки в неконтролируемых моделях, а также то, что эксперименты выполнены на одном корпусе с конкретными свойствами (дореформенная орфография, разное качество сканов, зачеркивания). Эти факторы накладывают практические ограничения на немедленную массовую автоматизацию.

На основе проведенной работы сформулированы практические рекомендации для внедрения LLM-коррекции в пайплайны распознавания исторических рукописей:

- применять LLM-коррекцию на уровне строк с жесткими ограничениями на формат вывода;
- конструировать промпты с явными примерами (в рамках подхода обучения на нескольких примерах) и контролировать историю исправлений для согласованности;
- внедрять автоматическую поствалидацию и фильтрацию результатов LLM перед передачей эксперту;
- учитывать затраты и приватность при выборе LLM.

В будущем мы планируем расширить и углубить исследование в нескольких ключевых направлениях. В частности, предстоит исследовать стратегии селекции данных для разметки. Одно из направлений – это оценка показателей ансамблевой неопределенности (по аналогии с [15]) как одного из сигналов приоритизации строк. Предварительный анализ показал, что простая реализация неопределенности может указывать на «трудные» строки, однако ее прямое применение к отбору обучающих примеров не гарантирует устойчивого улучшения. Поэтому необходима более тщательная проработка (калибровка ансамбля, предфильтрация по качеству сегментации, комбинирование сигналов неопределенности и представительности). Кроме того, планируются изучение альтернативных мер неопределенности, методов агрегации ансамблей, тестирование различных LLM и промпт-стратегий, а также масштабирование подхода на другие корпуса исторических рукописей.

Таким образом, работа показала практическую ценность использования LLM для посткоррекции автоматических расшифровок в рамках итерационного пайплайна – это реальный путь к сокращению ручной работы экспертов при оцифровке архивов. Одновременно установлены важные ограничения и направления доработки, реализация которых сделает предложенный подход более надежным, масштабируемым и пригодным для промышленного применения.

Благодарность

Работа поддержана грантом РНФ №22-68-00066 «Культурное наследие России: интеллектуальный анализ и тематическое моделирование корпуса рукописных текстов».

СПИСОК ЛИТЕРАТУРЫ

1. *Пенская Е.Н., Купцова О.Н.* Невидимая величина. А.В. Сухово-Кобылин: театр, литература, жизнь. М.: Изд. дом ВШЭ, 2024. 472 с.
2. *Местецкий Л.М., Смирнова В.С.* Сегментация строк в изображениях рукописных документов // Материалы Международной конференции по компьютерной графике и зрению (Графикон-2025). Поволжский государственный технологический университет, Йошкар-Ола, Россия, 2025.

3. Местецкий Л.М., Зыков В.П. Инкрементная разметка рукописных архивных дневников XIX века // Программные продукты и системы. 2025. Т. 38, № 4. <https://doi.org/10.15827/0236-235X.152>
4. Coquenat D., Chatelain C., Paquet T. End-to-end Handwritten Paragraph Text Recognition Using a Vertical Attention Network // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45, No. 1. P. 508–524. <https://doi.org/10.1109/TPAMI.2022.3144899>
5. Болтунова Е.М., Лаптев А.К. Распознавание рукописного текста и интеллектуальный анализ: возможности нейронных технологий (на примере работы с «Дневником» Ф.П. Литке) // Имагология и компаративистика. 2025. № 23. С. 358–379. <https://doi.org/10.17223/24099554/23/17>
6. Brown T. B., Mann B., Ryder N., Subbiah M. et al. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems (NeurIPS). 2020. Vol. 33. P. 1877–1901.
7. Marti U.-V., Bunke H. The IAM-database: an English sentence database for offline handwriting recognition // International Journal on Document Analysis and Recognition (IJ DAR). 2002. Vol. 5, No. 1. P. 39–46. <https://doi.org/10.1007/s100320200071>
8. Sánchez J., Romero V., Toselli A. H., Vidal E. ICFHR2016 competition on handwritten text recognition on the READ dataset // Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition (ICFHR 2016). 2016. P. 630–635.
9. Shi B., Bai X., Yao C. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 39, No. 11. P. 2298–2304. <https://doi.org/10.1109/TPAMI.2016.2646371>
10. Graves A., Fernández S., Gomez F., Schmidhuber J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks // Proceedings of the 23rd International Conference on Machine Learning (ICML 2006). 2006. P. 369–376. <https://doi.org/10.1145/1143844.1143891>

11. *Coquenat D., Chatelain C., Paquet T.* SPAN: A Simple Predict & Align Network for Handwritten Paragraph Recognition // Document Analysis and Recognition – ICDAR 2021. Lecture Notes in Computer Science, Vol. 12823. Springer, 2021. P. 70–84. https://doi.org/10.1007/978-3-030-86334-0_5
 12. *Yousef M., Bishop T.E.* OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by Learning to Unfold // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020). 2020. P. 14710–14719. <https://doi.org/10.1109/CVPR42600.2020.01472>
 13. *Li M., Lv T., Chen J., Cui L., Lu Y., Florencio D., Zhang C., Li Z., Wei F.* TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models // Proceedings of the AAAI Conference on Artificial Intelligence. 2023. Vol. 37, No. 12. P. 14216–14224.
 14. *Potantin M., Dimitrov D., Shonenkov A., Bataev V., Karachev D., Novopolitsev M., Chertok A.* Digital Peter: New Dataset, Competition and Handwriting Recognition Methods // Proceedings of the 6th International Workshop on Historical Document Imaging and Processing. ACM, 2021. P. 43–48. <https://doi.org/10.1145/3476887.3476892>
 15. *Lakshminarayanan B., Pritzel A., Blundell C.* Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles // Advances in Neural Information Processing Systems (NeurIPS). 2017. Vol. 30. P. 6402–6413.
-

POST-CORRECTION OF WEAK TRANSCRIPTIONS BY LARGE LANGUAGE MODELS IN THE ITERATIVE PROCESS OF HANDWRITTEN TEXT RECOGNITION

V. P. Zykov¹ [0009-0007-8935-9288], L. M. Mestetskiy² [0000-0001-6387-167X]

^{1, 2}*Lomonosov Moscow State University, Moscow, Russia*

²*HSE University, Moscow, Russia*

¹zykovvp@my.msu.ru, ²mestlm@mail.ru

Abstract

This paper addresses the problem of accelerating the construction of accurate editorial annotations for handwritten archival texts within an incremental training cycle based on weak transcription. Unlike our previously published results, the present work focuses on integrating automatic post-correction of weak transcriptions using large language models (LLMs). We propose and implement a protocol for applying LLMs at the line level in a few-shot setup with carefully designed prompts and strict output format control (preservation of pre-reform orthography, protection of proper names and numerals, prohibition of structural changes to lines). Experiments are conducted on the corpus of diaries by A.V. Sukhovo-Kobylin. As the base recognition model, we use the line-level variant of the Vertical Attention Network (VAN). Results show that LLM post-correction—exemplified by the ChatGPT-4o service—substantially improves the readability of weak transcriptions and significantly reduces the word error rate (in our experiments by about –12 percentage points), without degrading the character error rate. Another service tested, DeepSeek-R1, demonstrated less stable behavior. We discuss practical prompt engineering, limitations (context length limits, risk of “hallucinations”), and provide recommendations for the safe integration of LLM post-correction into an iterative annotation pipeline to reduce expert annotators’ workload and speed up the digitization of historical archives.

Keywords: *handwritten text recognition, weak transcription, Vertical Attention Network (VAN), large language models (LLM), post-correction, iterative retraining.*

REFERENCES

1. Penskaya E.N., Kuptsova O.N. (2024) The Invisible Quantity. A.V. Sukhovo-Kobylin: Theater, Literature, Life. Moscow: HSE Publishing House, 2024. 472 p. (In Russ.)
2. Mestetsky L.M., Smirnova V.S. Line segmentation in images of handwritten documents // Proceedings of the International Conference on Computer Graphics and Vision (Grafikon-2025). Yoshkar-Ola: Volga State Technological University, 2025. (In Russ.)
3. Mestetskiy L.M., Zykov V.P. Incremental markup of 19th-century handwritten archival diaries // Software & Systems. 2025. Vol. 38, No. 4. <https://doi.org/10.15827/0236-235X.152>. (In Russ.)
4. Coquenat D., Chatelain C., Paquet T. End-to-end Handwritten Paragraph Text Recognition Using a Vertical Attention Network // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45, No. 1. P. 508–524. <https://doi.org/10.1109/TPAMI.2022.3144899>
5. Boltunova E.M., Laptev A.K. Handwriting recognition and data mining: Possibilities of neural network technologies (based on admiral Fyodor Lutke's diary) // Imagology and Comparative Studies. 2025. No. 23. P. 358–379. <https://doi.org/10.17223/24099554/23/17>. (In Russ.)
6. Brown T.B., Mann B., Ryder N., Subbiah M. et al. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems (NeurIPS). 2020. Vol. 33. P. 1877–1901.
7. Marti U.-V., Bunke H. The IAM-database: an English sentence database for offline handwriting recognition // International Journal on Document Analysis and Recognition (IJ DAR). 2002. Vol. 5, No. 1. P. 39–46. <https://doi.org/10.1007/s100320200071>
8. Sánchez J., Romero V., Toselli A. H., Vidal E. ICFHR2016 competition on handwritten text recognition on the READ dataset // Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition (ICFHR 2016). 2016. P. 630–635.
9. Shi B., Bai X., Yao C. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition // IEEE

Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 39, No. 11. P. 2298–2304. <https://doi.org/10.1109/TPAMI.2016.2646371>

10. Graves A., Fernández S., Gomez F., Schmidhuber J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks // Proceedings of the 23rd International Conference on Machine Learning (ICML 2006). 2006. P. 369–376. <https://doi.org/10.1145/1143844.1143891>

11. Coquenat D., Chatelain C., Paquet T. SPAN: A Simple Predict & Align Network for Handwritten Paragraph Recognition // Document Analysis and Recognition – ICDAR 2021. Lecture Notes in Computer Science, Vol. 12823. Springer, 2021. P. 70–84. https://doi.org/10.1007/978-3-030-86334-0_5

12. Yousef M., Bishop T.E. OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by Learning to Unfold // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020). 2020. P. 14710–14719. <https://doi.org/10.1109/CVPR42600.2020.01472>

13. Li M., Lv T., Chen J., Cui L., Lu Y., Florencio D., Zhang C., Li Z., Wei F. TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models // Proceedings of the AAAI Conference on Artificial Intelligence. 2023. Vol. 37, No. 12. P. 14216–14224.

14. Potanin M., Dimitrov D., Shonenkov A., Bataev V., Karachev D., Novopolitsev M., Chertok A. Digital Peter: New Dataset, Competition and Handwriting Recognition Methods // Proceedings of the 6th International Workshop on Historical Document Imaging and Processing. ACM, 2021. P. 43–48. <https://doi.org/10.1145/3476887.3476892>

15. Lakshminarayanan B., Pritzel A., Blundell C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles // Advances in Neural Information Processing Systems (NeurIPS). 2017. Vol. 30. P. 6402–6413.

СВЕДЕНИЯ ОБ АВТОРАХ



ЗЫКОВ Валерий Павлович – магистрант кафедры «Математические методы прогнозирования» факультета вычислительной математики и кибернетики МГУ имени М.В. Ломоносова. Область научных интересов: машинное обучение, распознавание рукописных текстов, математика.

Valerii Pavlovich ZYKOV – Master's student at the Department "Mathematical Forecasting Methods", Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. Research interests: machine learning, handwriting recognition, mathematics.

email: zkovvp@my.msu.ru

ORCID: 0009-0007-8935-9288



МЕСТЕЦКИЙ Леонид Моисеевич – доктор технических наук, академик РАН, профессор кафедры математических методов прогнозирования МГУ. Научные интересы: вычислительная геометрия, анализ и распознавание изображений.

Leonid Moiseevich MESTETSKIY – Doctor of Engineering Sciences, Academician of the Russian Academy of Natural Sciences, Professor at the Department of Mathematical Forecasting Methods at Moscow State University. His research interests include computational geometry and image analysis and recognition.

email: mestlm@mail.ru

ORCID: 0000-0001-6387-167X

Материал поступил в редакцию 1 ноября 2025 года

НЕКОТОРЫЕ ПОДХОДЫ К ПОВЫШЕНИЮ ТОЧНОСТИ

ПРОГНОЗИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ АНСАМБЛЕВЫХ МЕТОДОВ

С. Ма¹ [0009-0004-0356-9996], О. В. Сенько² [0000-0002-5586-3503]

^{1, 2}Московский государственный университет имени М. В. Ломоносова,
г. Москва, Россия

²Федеральный исследовательский центр «Информатика и управление»
Российской академии наук, г. Москва, Россия

¹xinyuema35@gmail.com, ²OSenko@frccsc.ru

Аннотация

Представлены результаты экспериментального исследования эффективности использования сверхслучайных деревьев в моделях, основанных на градиентном бустинге, а также в новом ансамблевом методе, в котором лес генерируется, исходя из условия повышенной внутренней дивергенции. Исследована эффективность сверхслучайных деревьев при использовании расширенных наборов признаков с включением новых признаков, вычисляемых как расстояния Идо набора описаний опорных объектов из обучающей выборки. Показано, что использование сверхслучайных деревьев в моделях градиентного бустинга и дивергентного леса позволяет улучшить обобщающую способность, а также, что к еще большему росту обобщающей способности приводит использование расширенных наборов признаков.

Ключевые слова: регрессионное моделирование, ансамблевое обучение, метрическое пространство, метод сверхслучайных деревьев.

ВВЕДЕНИЕ

Ансамблевые методы, основанные на использовании комбинаций более простых алгоритмов, получили широкое распространение при решении разнообразных прикладных задач прогнозирования числовых целевых переменных. Высокая эффективность ансамблевых методов, убедительно подтверждаемая результатами многочисленных экспериментов, делает актуальными дальнейшие

исследования по их совершенствованию. Методы ансамблевого обучения основаны на вычислении коллективного прогноза по набору прогнозов, вычисляемых базовыми алгоритмами, вошедшими в ансамбль, что позволяет повысить устойчивость и точность предсказаний. Формально предсказание ансамбля можно записать в виде

$$a(x) = m(b_1(x), \dots, b_n(x)),$$

где $b_i(x)$, $i = 1, \dots, n$, – предсказания отдельных базовых моделей, а $m(x)$ – мета-алгоритм, агрегирующий эти предсказания.

Построение ансамбля состоит из двух этапов: во-первых, обучение нескольких базовых моделей; во-вторых, применение стратегии объединения их выходов для получения финального результата. Однако ансамблевые методы обычно используют фиксированные способы объединения моделей. Теоретическое обоснование целесообразности использования ансамблей восходит к теореме Кондорсе о присяжных [1]. В соответствии с этой теоремой, если каждый голосующий высказывает независимое мнение и в среднем принимает верное решение чаще, чем ошибается, то вероятность правильного вердикта большинства стремится к единице по мере увеличения числа голосующих.

Ансамблевые методы имеют длительную историю. Одними из первых ансамблевых методов были тестовый алгоритм [2] и алгоритм Кора [3], предложенные еще в 1960-е годы. Идея использования ансамблей решающих и регрессионных деревьев возникла в 1993 г. (см. в [4]). В 2001 г. окончательно оформилась идея случайного леса [5], в котором ансамбль регрессионных или решающих деревьев генерируется с использованием метода бэггинга (bagging, [6]) и метода случайных подпространств [7]. В методе бэггинга деревья обучаются по выборкам, которые являются выборками с возвращением из исходной обучающей выборки. В методе случайных подпространств обучение производится по выборкам, получаемым из исходной выборки с помощью случайного выбора подмножества признаков. Необходимо отметить, что при построении случайного леса каждое новое дерево строится независимо от предыдущих деревьев, исходя из условия наилучшей аппроксимации исходной целевой переменной.

В отличие от случайного леса, метод бустинга (boosting, [8–11]) направлен на построение линейной комбинации «слабых» алгоритмов. На каждом шаге

в нее добавляется новое слагаемое согласно условию минимизации ошибки линейной комбинации. Одним из первых представителей данного подхода является алгоритм AdaBoost [12], в котором при обучении используются веса объектов обучающей выборки. На первом шаге веса объектов выбираются равными. На последующих шагах увеличиваются веса объектов, предсказания для которых были ошибочными. Более широкое распространение по сравнению с AdaBoost в последнее время получил градиентный бустинг, в котором минимизация потерь на каждом шаге производится с использованием градиентного спуска, а каждое новое дерево, добавляемое в линейную комбинацию, аппроксимирует антиградиент функции потерь. Подобные методы демонстрируют очевидные преимущества при моделировании сложных нелинейных зависимостей.

С развитием алгоритмов и вычислительных ресурсов современные методы градиентного бустинга были значительно усовершенствованы с точки зрения производительности и эффективности. Широкое распространение получили модификации градиентного бустинга: XGBoost [13], LightGBM [14] и CatBoost [15]. В [16–18] предложен новый вариант регрессионного леса, основанный на анализе разложения квадратичной ошибки выпуклых комбинаций предикторов. Из разложения следует, что ошибка может быть значительно снижена при увеличении взаимного квадратичного отклонения прогнозов алгоритмов, входящих в ансамбль. В связи с этим было предложено при построении нового дерева, включаемого в ансамбль, не только минимизировать квадратичную ошибку прогноза, но и одновременно максимизировать квадратичное отклонение от текущего ансамбля. Эксперименты показали, что такой подход, который носит название дивергентного леса [18], во многих случаях позволяет снижать ошибку ансамбля.

В современных вариантах алгоритмов случайного регрессионного леса и различных вариантов градиентного бустинга обычно используются регрессионные деревья, в которых пороги для признаков выбираются по критериям минимизации ошибки прогноза. В последнее время растет интерес к так называемым сверхслучайным деревьям (Extra Randomized Trees [19]), в которых пороговые значения для признаков выбираются случайно. Эксперименты показали [19], что нередко леса, состоящие из сверхслучайных деревьев, превосходят по обобщаю-

щей способности стандартные регрессионные случайные леса, требуя значительно меньше времени на обучение. В связи с этим возникла идея исследовать эффективность использования сверхслучайных деревьев также в градиентном бустинге и дивергентном лесе.

Другим способом повышения обобщающей способности является трансформация признакового пространства [20]. Поэтому еще одной целью настоящей работы стало исследование эффективности ансамблевых методов с использованием в качестве признаков расстояний до опорных векторных описаний объектов из обучающей выборки. Обучающая выборка состоит из n объектов, каждый из которых описывается m признаками: $x_i = (x_{i1}, x_{i2}, \dots, x_{im}) \in R^m$ и включает соответствующее целевое значение $y_i \in R$. Тогда обучающая выборка записывается как $S = \{(x_i, y_i)\}, i = 1, \dots, n$.

МЕТОДОЛОГИЯ

Опишем методологию исследования, включая характеристики набора данных, общую структуру ансамблевых моделей и принципы их построения.

Описание данных

В проведенном исследовании был использован набор данных, содержащий физико-химические характеристики различных химических соединений и соответствующие значения их температуры плавления. Каждый образец (соединение) описан множеством числовых признаков, соответствующих свойствам структуры, энергии и состава вещества. Целевой переменной является температура плавления соединения T_m , измеряемая в кельвинах (K).

Данные представлены в табличной форме:

- первая колонка содержит наименование соединения (Compound);
- во второй указано значение целевой переменной T_m ;
- остальные столбцы соответствуют числовым признакам, всего представлено m характеристик;
- полный набор данных включает n наблюдений.

Перед началом обучения данные были разделены на обучающую и тестовую выборки в соотношении 80/20 для последующего обучения и проверки модели.

Используемые модели

Перед описанием предлагаемых ансамблей рассмотрим базовый алгоритм, на основе которого они построены, – метод сверхслучайных деревьев.

Краткое сравнение сверхслучайного дерева и стандартного регрессионного дерева приведено в табл. 1.

Метод снижает вычислительные затраты на построение модели и повышает разнообразие деревьев по сравнению с классическим методом [21], что способствует лучшей обобщающей способности ансамбля и снижает риск переобучения.

Табл. 1. Сравнение ансамблей на основе моделей сверхслучайного дерева и стандартного регрессионного дерева (Regression Tree).

Критерий сравнения	Сверхслучайное дерево	Регрессионное дерево
Выбор точки разделения	Случайная точка разделения (<i>random split</i>)	Лучшая точка разделения (<i>best split</i>)
Скорость обучения	Быстрее (нет поиска, выбирается случайно)	Медленнее (нужно искать оптимальное разделение)
Смещение/дисперсия	Более высокое смещение, низкая дисперсия → устойчивость	Низкое смещение, высокая дисперсия → переобучение

В качестве модели регрессии A мы реализовали два ансамблевых подхода на основе метода сверхслучайных деревьев, в которых одиночное дерево используется в качестве базового алгоритма, добавляемого в ансамбль на каждой итерации:

- градиентный бустинг с аппроксимацией на каждом шаге градиента функции потерь;
- дивергентный лес с итеративным обновлением целевой переменной.

Градиентный бустинг

Градиентный бустинг формирует итоговую модель поэтапно, последовательно добавляя новые базовые алгоритмы, каждый из которых нацелен на устранение ошибок предыдущих шагов. В отличие от случайного леса, где деревья строятся независимо, бустинг организует обучение в последовательной форме, направляя внимание на трудные для предсказания наблюдения.

Пусть задана функция потерь $L(y, F(x))$, измеряющая расхождение между истинным значением y и предсказанием модели $F(x)$.

На каждом шаге бустинг добавляет новую базовую модель $h_k(x)$, которая аппроксимирует направление наискорейшего спуска функции потерь – отрицательный градиент.

Алгоритм градиентного бустинга:

- Инициализация

$$F_0(x) = \operatorname{argmin}_c \sum_{i=1}^n L(y_i, c).$$

- Для каждой итерации $k = 1, 2, \dots, K$:

– вычисляются антиградиенты

$$r_i^{(k)} = \frac{-\partial L(y_i, F_{k-1}(x_i))}{\partial F_{k-1}(x_i)};$$

– обучается базовая модель $h_k(x)$ по парам $(r_i^{(k)}, x_i)$;

– обновляется ансамбль:

$$F_k(x) = F_{k-1}(x) + \eta \cdot h_k(x), \text{ где } \eta \in (0, 1] \text{ – шаг обучения (learning rate).}$$

Если функция потерь имеет вид

$$L(y, F) = \frac{1}{2} (y - F)^2,$$

то отрицательный градиент совпадает с обычным остатком:

$$r_i^{(k)} = y_i - F_{k-1}(x).$$

Следовательно, на каждом шаге базовая модель $h_k(x)$ обучается на остатках предыдущего шага.

В рамках настоящего исследования градиентный бустинг был реализован с использованием сверхслучайных деревьев в качестве базовых моделей. В реализации для текущей задачи процесс обучения выглядит следующим образом.

- На первом шаге вычисляются остатки $r_i^{(1)} = y_i - \bar{Y}$, где \bar{Y} – среднее значение Y на обучающей выборке.
- Строится сверхслучайное дерево T_1 по парам $(X_{\text{train}}, r_{\text{train}}^{(1)})$.
- Предсказание вычисляется по формуле

$$F_1(X_{\text{train}}) = \bar{Y} + \eta T_1(X_{\text{train}}).$$

- На шаге k вычисляются остатки
- $$r_i^{(k)} = y_i - F_{k-1}(x_i).$$
- Строится сверхслучайное дерево T_k по парам $(X_{\text{train}}, r_{\text{train}}^{(k)})$.
 - Предсказания обновляются по правилу

$$F_k(X_{\text{train}}) = F_{k-1}(X_{\text{train}}) + \eta \cdot T_k(X_{\text{train}}).$$

Дивергентный лес

В отличие от градиентного бустинга, который минимизирует функцию потерь на каждом шаге, метод дивергентного леса [17, 18] использует альтернативную схему обновления целевой переменной.

Разработанная нами модификация дивергентного леса представляет собой ансамблевый алгоритм, построенный на основе сверхслучайных деревьев. Основная идея метода заключается в итеративном изменении целевых переменных, что способствует увеличению разнообразия базовых моделей и улучшению общей точности предсказаний, такой подход объединяет концепции последовательного обучения, присущего бустингу, и стохастического характера сверхслучайных деревьев.

Каждая итерация ансамбля использует не исходные метки, а адаптированную целевую переменную $S_{\text{train}}^{(k)}$, скорректированную с учетом накопленных предсказаний. Это позволяет каждой новой модели обучаться на обновленных данных, что обеспечивает более устойчивое и сбалансированное поведение ансамбля.

Алгоритм метода можно представить следующим образом.

- Инициализация

$$Z_{\text{train}}^{(0)} = Y_{\text{train}}.$$

- Для каждой итерации k :

- обучается сверхслучайное дерево T_k на текущем векторе $Z_{\text{train}}^{(k-1)}$;
- вычисляется предсказание $P_k = T_k(X_{\text{train}})$;
- обновляется целевая переменная

$$\overline{P}_k = \frac{1}{k} \sum_{j=1}^k P_j, \quad Z_{\text{train}}^{(k-1)} = \frac{Y_{\text{train}} - \mu \overline{P}_k}{1 - \mu};$$

где $\mu \in (0,1)$ – параметр, регулирующий баланс между дивергенцией и точностью аппроксимации.

- Финальное предсказание при количестве итераций r вычисляется по формуле $\frac{1}{r} \sum_{j=1}^r P_j$.

Предложенный подход отличается от традиционного бустинга тем, что вместо оптимизации градиента ошибки применяется явное аналитическое обновление, основанное на сглаженном ансамбле предсказаний. Такой механизм позволяет контролировать вклад новых и предыдущих моделей посредством параметра μ , снижая чувствительность к шуму и нестабильности меток.

Таким образом, дивергентный лес можно рассматривать как обобщенную версию бустинга, ориентированную на повышение устойчивости и стабильности ансамбля при работе с реальными, зашумленными данными.

Метод опорных точек и расстояние Махаланобиса

Одним из важных элементов представленного подхода является использование метода опорных точек для построения нового метрического пространства. Основная идея состоит в том, чтобы расширить исходное пространство признаков новыми координатами, которые отражают расстояния от каждого объекта до специально выбранных репрезентативных (опорных) точек. Предполагается, что такое преобразование позволит точнее выделить структурные связи в данных и облегчит поиск скрытых закономерностей, которые не всегда хорошо улавливаются моделью регрессионных деревьев в исходных координатах при коррелированных признаках.

Пусть имеется обучающая выборка из n объектов, каждый из которых описывается m признаками. Введем множество из k опорных точек – базовых элементов, относительно которых будет формироваться новое пространство признаков. Эти точки можно рассматривать как «ориентиры», позволяющие измерять степень близости других объектов к сложным областям исходного пространства.

Для отбора опорных точек используем предварительно обученную модель A_0 . После обучения для каждого объекта вычислим ошибку предсказания

$$e_i = (A_0(x) - y_i)^2.$$

В качестве опорных выберем объекты, у которых значение ошибки наибольшее. Интуитивно ясно, что это те наблюдения, которые труднее всего аппроксимировать базовой моделью. Если использовать их как эталоны, то можно улучшить способность модели учитывать сложные или слабо представленные области пространства признаков. Такой принцип близок идее адаптивного бустинга, где внимание последующих моделей сосредоточено на наблюдениях, для которых предыдущие модели допускают наибольшую ошибку. Для каждого объекта x_i построим новый вектор признаков z_i , компоненты которого представляют собой расстояния до всех выбранных опорных точек:

$$z_i = (\rho(x_i, x_1), \rho(x_i, x_2), \dots, \rho(x_i, x_k)).$$

В результате получим новое расширенное описание как конкатенацию векторов x_i и z_i , которое далее обозначим как $[x_i, z_i]$.

В качестве меры расстояния используется расстояние Махаланобиса [22], определяемое формулой

$$\rho_M(x_i, x_k) = \sqrt{\sum_{p=1}^m \sum_{q=1}^m (x_{ip} - x_{kp}) \Sigma_{pq}^{-1} (x_{iq} - x_{kq})},$$

где Σ – ковариационная матрица.

В отличие от евклидовой метрики, расстояние Махаланобиса учитывает масштаб признаков и их взаимную корреляцию. Это особенно важно, когда признаки сильно различаются по дисперсии или связаны между собой. В таких случаях евклидово расстояние может искажать реальную структуру данных, тогда как расстояние Махаланобиса дает более корректную оценку близости.

После добавления новых координат преобразуем исходную выборку: $S' = \{([x_i, z_i], y_i)\}, i = 1, \dots, n$. Затем обучим новую модель $A[x_i, z_i] \approx y_i$.

Построение признаков на основе расстояний до опорных точек делает модель более гибкой и чувствительной к структуре данных. Метод особенно полезен при наличии значительной корреляции признаков, а также при сложной

форме распределения данных. Тем не менее чрезмерное увеличение числа новых признаков может привести к переобучению, поэтому количество опорных точек k следует подбирать эмпирически.

В ходе экспериментов было показано, что использование расширенного пространства признаков на основе расстояния Махаланобиса дает более устойчивые результаты по сравнению с другими метриками. Причина заключается в том, что эта метрика устраняет два эффекта:

- масштабный эффект: признаки с высокой дисперсией не доминируют над остальными;
- корреляционный эффект: сильно зависимые признаки не учитываются дважды.

ПРОВЕДЕННЫЕ ЭКСПЕРИМЕНТЫ

Основная цель экспериментов заключалась в проверке эффективности предложенного подхода. Эксперименты были направлены на сравнение точности и устойчивости различных ансамблевых моделей в задаче регрессии.

Сравниваемые модели

В ходе экспериментов были рассмотрены следующие методы регрессии.

- Случайный лес, построенный с использованием сверхслучайных деревьев в качестве базового алгоритма (ET in RF).
- Дивергентный лес, построенный с использованием стандартных регрессионных деревьев в качестве базового алгоритма (RT in DF).
- Дивергентный лес, построенный с использованием сверхслучайных деревьев в качестве базового алгоритма (ET in DF).
- Дивергентный лес, построенный с использованием расширенного описания и сверхслучайных деревьев в качестве базового алгоритма (ET in DF (расш.)).
- Классический градиентный бустинг на деревьях решений (GBRT).
- Градиентный бустинг с использованием сверхслучайных деревьев в качестве базового алгоритма (GBET).

- Градиентный бустинг, построенный с использованием расширенного описания и сверхслучайных деревьев в качестве базового алгоритма (GBET (расш.))
- LightGBM – реализация градиентного бустинга с построением деревьев по принципу leaf-wise и оптимизациями по скорости и памяти.
- CatBoost – реализация бустинга, использующая упорядоченные статистики и устойчивую обработку категориальных признаков.

Параметры оценки качества

Для оценки качества регрессионных моделей был использован коэффициент детерминации R^2 , который показывает, какая доля дисперсии целевой переменной объясняется моделью.

Формально R^2 определяется как

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - f_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2},$$

где y_i – истинные значения целевой переменной, f_i – предсказанные моделью значения, \bar{y} – среднее значение по всем наблюдениям. Числитель $\sum_{i=1}^n (y_i - f_i)^2$ представляет собой остаточную сумму квадратов ошибок, а знаменатель $\sum_{i=1}^n (y_i - \bar{y})^2$ – полную сумму квадратов. Таким образом, R^2 измеряет, насколько модель уменьшает ошибку по сравнению с простейшей моделью, которая всегда предсказывает среднее значение.

В табл. 2 представлены значения коэффициента детерминации R^2 для десяти задач. По строкам указаны задачи, по столбцам – использованные методы, указанные выше; обозначения Rel. diff (%) – относительная разница между наилучшим и наихудшим результатом; N – число объектов; n – число признаков.

Таблица 2. Результаты эксперимента

	RT in DF	ET in RF	ET in DF	ET in DF (расш.)	CatBoost	LightGBM	GBRT	GBET	GBET (расш.)	Rel.diff (%)	Size (N x n)
Задача 1	0.894	0.902	0.904	0.904	0.881	0.845	0.902	0.917	0.917	8.5	439*86
Задача 2	0.906	0.905	0.924	0.939	0.903	0.921	0.912	0.932	0.934	4.1	451*98
Задача 3	0.918	0.94	0.942	0.943	0.935	0.932	0.923	0.945	0.947	3.1	439*86
Задача 4	0.850	0.892	0.914	0.915	0.901	0.917	0.870	0.908	0.920	8.2	196*88
Задача 5	0.851	0.892	0.916	0.919	0.908	0.912	0.870	0.908	0.929	9.2	447*90
Задача 6	0.797	0.860	0.896	0.919	0.890	0.886	0.796	0.897	0.906	15.4	234*98
Задача 7	0.902	0.908	0.927	0.933	0.873	0.927	0.899	0.928	0.928	7.0	231*98
Задача 8	0.902	0.889	0.926	0.935	0.908	0.911	0.919	0.926	0.926	5.2	235*86
Задача 9	0.864	0.833	0.869	0.877	0.848	0.877	0.851	0.870	0.877	5.3	195*88
Задача 10	0.844	0.768	0.863	0.863	0.869	0.841	0.768	0.863	0.879	14.5	173*68

Анализ результатов

Модели, основанные на сверхслучайных деревьях, демонстрируют более высокие результаты по сравнению с моделями на деревьях решений. Их использование в структуре дивергентного леса обеспечивает лучшие показатели, чем в случайном лесе, что подтверждает большую значимость дивергентного ансамблирования для данного базового алгоритма.

Сравнение градиентного бустинга на деревьях решений и градиентного бустинга на сверхслучайных деревьях показало преимущество второго: за счет большего разнообразия базовых моделей и снижения корреляции их ошибок ансамбль на сверхслучайных деревьях демонстрирует более высокую обобщающую способность и меньшую склонность к переобучению на рассмотренных данных.

Применение нового метрического пространства стабильно улучшает результаты. Так, сверхслучайные деревья в дивергентном лесе с новой метрикой дают более высокие значения R^2 по сравнению с исходным пространством во всех задачах. Аналогично, градиентный бустинг на сверхслучайных деревьях в новом метрическом пространстве также демонстрирует улучшения относительно базового варианта.

Наиболее высокие результаты наблюдаются в следующих случаях:

- в задачах 2, 6, 7 и 8 – сверхслучайные деревья в дивергентном лесе с расширенным набором признаков;
- в задачах 1, 3, 4, 5 и 10 – градиентный бустинг на сверхслучайных деревьях с расширенным набором признаков;
- в задаче 9 оба подхода показали сопоставимые и максимально высокие значения R^2 .

Таким образом, использование нового метрического пространства повышает устойчивость и эффективность моделей, однако выбор оптимального метода зависит от специфики конкретной задачи.

ЗАКЛЮЧЕНИЕ

Исследованы различные ансамблевые методы регрессии, включая случайный лес, градиентный бустинг, LightGBM, CatBoost, а также предложена модификация метода дивергентного леса. Особое внимание уделено выбору базовых алгоритмов и анализу влияния метрических методов расширения признакового пространства на качество предсказаний.

Результаты экспериментов показали, что использование метода сверхслучайных деревьев в качестве базового алгоритма обеспечивает наилучшее соотношение между точностью и устойчивостью модели. По сравнению с классическими

деревьями решений, метод сверхслучайных деревьев демонстрирует более высокие значения R^2 во всех вариантах ансамблей – как в случайном лесе, так и в дивергентном лесе и градиентном бустинге.

Метод дивергентного леса продемонстрировал преимущество по сравнению со случайным лесом, особенно при увеличении параметра баланса μ . Это подтверждает эффективность итеративного обновления целевой переменной, которое позволяет модели лучше адаптироваться к сложной структуре данных.

Переход к расширенному признаковому пространству с помощью метода на основе метрики Махаланобиса дал дополнительное улучшение качества для всех моделей. Данная метрика оказалась особенно полезной при наличии коррелированных признаков, так как она учитывает взаимные зависимости и масштаб признаков, формируя более адекватное представление о расстояниях между объектами.

В целом можно сделать следующие выводы.

- Метод сверхслучайных деревьев как базовый алгоритм обеспечивает стабильное и высокое качество прогнозов как в дивергентном лесе, так и в градиентном бустинге.
- Использование расширенного признакового пространства с помощью метода на основе метрики Махаланобиса позволяет улучшить обобщающую способность моделей и повысить точность на сложных данных.

Таким образом, предложен подход, основанный на сочетании двух принципов: использование сверхслучайных деревьев в дивергентном лесе и градиентном бустинге; расширение признакового пространства с использованием метрики Махаланобиса. Показано, что подход повышает эффективность регрессионного моделирования и улучшает адаптацию моделей к структурно сложным наборам данных.

СПИСОК ЛИТЕРАТУРЫ

1. Хабр. Открытый курс машинного обучения. Тема 5. Композиции: бэггинг, случайный лес [Электронный ресурс].

URL: <https://habr.com/ru/companies/ods/articles/324402/> (дата обращения: 06.11.2025).

2. *Дмитриев А.И., Журавлев Ю.И., Кренделев Ф.П.* О математических принципах классификации предметов или явлений // Дискретный анализ. 1967. Вып. 7. С. 3–17.
3. *Вайнцвайг М.Н.* Алгоритм обучения распознаванию образов «Кора» // Алгоритмы обучения распознаванию образов. М.: Сов. радио, 1973. С. 8–12.
4. *Heath D., Kasif S., Salzberg S.* k-DT: A multi-tree learning method // Proceedings of the Second International Workshop on Multistrategy Learning. 1993. P. 138–149. https://doi.org/10.1007/0-387-34296-6_10
5. *Breiman L.* Random Forests // Machine Learning. 2001. Vol. 45, No. 1. P. 5–32. <https://doi.org/10.1023/A:1010933404324>
6. *Breiman L.* Bagging predictors // Machine Learning. 1996. Vol. 24, No. 2. P. 123–140. <https://doi.org/10.1007/BF00058655>
7. *Ho T.K.* The Random Subspace Method for Constructing Decision Forests // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998. Vol. 20, No. 8. P. 832–844. <https://doi.org/10.1109/34.709601>
8. *Freund Y., Schapire R.E.* A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting // Journal of Computer and System Sciences. 1997. Vol. 55. P. 119–139. <https://doi.org/10.1006/jcss.1997.1504>
9. *Friedman J.H.* Stochastic Gradient Boosting // Computational Statistics & Data Analysis. 2002. Vol. 38, No. 4. P. 367–378. [https://doi.org/10.1016/S0167-9473\(01\)00065-2](https://doi.org/10.1016/S0167-9473(01)00065-2)
10. *Zhou Z.H.* Ensemble Methods: Foundations and Algorithms. New York: Chapman and Hall/CRC, 2012. 446 p.
11. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning: Data Mining, Inference and Prediction. New York: Springer, 2009. 745 p. <https://doi.org/10.1007/978-0-387-84858-7>
12. *Beja-Battais P.* Overview of AdaBoost: Reconciling its Views to Better Understand its Dynamics // arXiv preprint arXiv:2310.18323 [cs.LG]. 2023. <https://doi.org/10.48550/arXiv.2310.18323>
13. *Chen T., Guestrin C.* XGBoost: A Scalable Tree Boosting System // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 785–794. <https://doi.org/10.48550/arXiv.1603.02754>

14. Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., Liu T.-Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree // *Advances in Neural Information Processing Systems (NeurIPS)*. 2017. Vol. 30.
 15. Hancock J.T., Khoshgoftaar T.M. CatBoost for big data: an interdisciplinary review // *Journal of Big Data*. 2020. Vol. 7, No. 94. 45 p. <https://doi.org/10.1186/s40537-020-00369-8>
 16. Zhuravlev Yu.I., Senko O.V., Dokukin A.A., Kiselyova N.N., Saenko I.A. Two-Level Regression Method Using Ensembles of Trees with Optimal Divergence // *Doklady Mathematics*. 2021. Vol. 103, No. 1. P. 1–4. <https://doi.org/10.1134/S1064562421040177>
 17. Dokukin A.A., Sen'ko O.V. A New Two-Level Machine Learning Method for Evaluating the Real Characteristics of Objects // *Journal of Computer and Systems Sciences International*. 2023. Vol. 62, No. 4. P. 607–614. <https://doi.org/10.1134/S1064230723040020>
 18. Senko O.V., Dokukin A.A., Kiselyova N.N., Dudarev V.A., Kuznetsova Yu.O. New Two-Level Ensemble Method and Its Application to Chemical Compounds Properties Prediction // *Lobachevskii Journal of Mathematics*. 2023. Vol. 44, No. 1. P. 188–197. <https://doi.org/10.1134/S1995080223010341>
 19. Geurts P., Ernst D., Wehenkel L. Extremely Randomized Trees // *Machine Learning*. 2006. Vol. 63, No. 1. P. 3–42. <https://doi.org/10.1007/s10994-006-6226-1>
 20. López-Iñesta E., Grimaldo F., Arevalillo-Herráez M. Combining feature extraction and expansion to improve classification-based similarity learning // *Pattern Recognition Letters*. 2016. Vol. 85. P. 84–90. <https://doi.org/10.1016/j.patrec.2016.11.005>
 21. Breiman L., Friedman J., Olshen R.A., Stone C.J. *Classification and Regression Trees*. Monterey, CA: Wadsworth & Brooks/Cole, 1984. 358 p. ISBN 978-0-412-04841-8. <https://doi.org/10.1201/9781315139470>
 22. Mahalanobis P.C. On the Generalised Distance in Statistics (reprint of 1936) // *Sankhya A*. 2018. Vol. 80, Suppl. 1. P. 1–7. <https://doi.org/10.1007/s13171-019-00164-5>
-
-

SOME APPROACHES TO IMPROVING PREDICTION ACCURACY USING ENSEMBLE METHODS

X. Ma¹ [0009-0004-0356-9996], O. V. Senko² [0000-0002-5586-3503]

^{1,2}*Lomonosov Moscow State University, Moscow, Russia*

²*Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, Moscow, Russia*

¹xinyuema35@gmail.com, ²OSenko@frccsc.ru

Abstract

This study presents the results of an experimental analysis evaluating the effectiveness of Extra Trees within gradient boosting models, as well as in a newly proposed ensemble framework where the forest is generated under conditions of enhanced internal divergence. Additionally, the paper explores the performance of Extra Trees when applied to novel feature representations computed as distances to a selected set of reference examples. It has been shown that the use of Extra Randomized Trees in gradient boosting and divergent forest models improves generalization ability. The use of expanded feature sets leads to even greater generalization ability.

Keywords: *regression modeling, ensemble learning, metric space, extremely randomized trees method.*

REFERENCES

1. *Habr. Open Machine Learning Course. Topic 5. Ensembles: Bagging, Random Forest.* Available at: <https://habr.com/ru/companies/ods/articles/324402/> (accessed 6 November 2025). (In Russ.).
2. *Dmitriev A.I., Zhuravlev Yu.I., Krendelev F.P. O matematicheskikh printsipakh klassifikatsii predmetov ili yavlenii [On the Mathematical Principles of the Classification of Objects and Phenomena] // Diskretnyi analiz [Discrete Analysis]. 1967. No. 7. P. 3–17 (In Russ.).*
3. *Vaintsvaig M.N. Algoritmy obucheniya raspoznavaniyu obrazov "Kora" [Algorithm for pattern recognition learning "Kora"] // Algoritmy obucheniya raspoznavaniyu obrazov [Algorithms for pattern recognition learning]. Moscow: Sovetskoe radio, 1973. P. 8–12 (In Russ.).*

4. *Heath D., Kasif S., Salzberg S.* k-DT: A multi-tree learning method // Proceedings of the Second International Workshop on Multistrategy Learning. 1993. P. 138–149. https://doi.org/10.1007/0-387-34296-6_10
5. *Breiman L.* Random Forests // Machine Learning. 2001. Vol. 45, No. 1. P. 5–32. <https://doi.org/10.1023/A:1010933404324>
6. *Breiman L.* Bagging predictors // Machine Learning. 1996. Vol. 24, No. 2. P. 123–140. <https://doi.org/10.1007/BF00058655>
7. *Ho T.K.* The Random Subspace Method for Constructing Decision Forests // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998. Vol. 20, No. 8. P. 832–844. <https://doi.org/10.1109/34.709601>
8. *Freund Y., Schapire R.E.* A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting // Journal of Computer and System Sciences. 1997. Vol. 55. P. 119–139. <https://doi.org/10.1006/jcss.1997.1504>
9. *Friedman J.H.* Stochastic Gradient Boosting // Computational Statistics & Data Analysis. 2002. Vol. 38, No. 4. P. 367–378. [https://doi.org/10.1016/S0167-9473\(01\)00065-2](https://doi.org/10.1016/S0167-9473(01)00065-2)
10. *Zhou Z.H.* Ensemble Methods: Foundations and Algorithms. New York: Chapman and Hall/CRC, 2012. 446 p. ISBN 978-1-4398-3003-1.
11. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning: Data Mining, Inference and Prediction. New York: Springer, 2009. 745 p. <https://doi.org/10.1007/978-0-387-84858-7>
12. *Beja-Battais P.* Overview of AdaBoost: Reconciling its Views to Better Understand its Dynamics // arXiv preprint arXiv:2310.18323 [cs.LG]. 2023. <https://doi.org/10.48550/arXiv.2310.18323>
13. *Chen T., Guestrin C.* XGBoost: A Scalable Tree Boosting System // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 785–794. <https://doi.org/10.48550/arXiv.1603.02754>
14. *Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., Liu T.-Y.* LightGBM: A Highly Efficient Gradient Boosting Decision Tree // Advances in Neural Information Processing Systems (NeurIPS). 2017. Vol. 30.

15. *Hancock J.T., Khoshgoftaar T.M.* CatBoost for big data: an interdisciplinary review // *Journal of Big Data*. 2020. Vol. 7, No. 94. 45 p. <https://doi.org/10.1186/s40537-020-00369-8>
16. *Zhuravlev Yu.I., Senko O.V., Dokukin A.A., Kiselyova N.N., Saenko I.A.* Two-Level Regression Method Using Ensembles of Trees with Optimal Divergence // *Doklady Mathematics*. 2021. Vol. 103, No. 1. P. 1–4. <https://doi.org/10.1134/S1064562421040177>
17. *Dokukin A.A., Sen'ko O.V.* A New Two-Level Machine Learning Method for Evaluating the Real Characteristics of Objects // *Journal of Computer and Systems Sciences International*. 2023. Vol. 62, No. 4. P. 607–614. <https://doi.org/10.1134/S1064230723040020>
18. *Senko O.V., Dokukin A.A., Kiselyova N.N., Dudarev V.A., Kuznetsova Yu.O.* New Two-Level Ensemble Method and Its Application to Chemical Compounds Properties Prediction // *Lobachevskii Journal of Mathematics*. 2023. Vol. 44, No. 1. P. 188–197. <https://doi.org/10.1134/S1995080223010341>
19. *Geurts P., Ernst D., Wehenkel L.* Extremely Randomized Trees // *Machine Learning*. 2006. Vol. 63, No. 1. P. 3–42. <https://doi.org/10.1007/s10994-006-6226-1>
20. *López-Iñesta E., Grimaldo F., Arevalillo-Herráez M.* Combining feature extraction and expansion to improve classification-based similarity learning // *Pattern Recognition Letters*. 2016. Vol. 85. P. 84–90. <https://doi.org/10.1016/j.patrec.2016.11.005>
21. *Breiman L., Friedman J., Olshen R.A., Stone C.J.* Classification and Regression Trees. Monterey, CA: Wadsworth & Brooks/Cole, 1984. 358 p. <https://doi.org/10.1201/9781315139470>
22. *Mahalanobis P.C.* On the Generalised Distance in Statistics (reprint of 1936) // *Sankhya A*. 2018. Vol. 80, Suppl. 1. P. 1–7. <https://doi.org/10.1007/s13171-019-00164-5>

СВЕДЕНИЯ ОБ АВТОРАХ



МА Синьюэ – студентка 2 курса магистратуры кафедры «Математические методы прогнозирования» факультета ВМК, МГУ имени М.В. Ломоносова. Область научных интересов – машинное обучение и интеллектуальный анализ данных. Число научных публикаций – 2.

Xinyue MA – second-year Master's student of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. Research interests: machine learning and data analysis. Number of publications – 2.

email: xinyuema35@gmail.com

ORCID: 0009-0004-0356-9996



СЕНЬКО Олег Валентинович – ведущий научный сотрудник Федерального исследовательского центра «Информатика и управление» Российской академии наук. Область научных интересов – машинное обучение и интеллектуальный анализ данных. Число научных публикаций – более 200.

Oleg Valentinovich SEN'KO – leading Researcher at the Federal Research Center "Informatics and Control" of the Russian Academy of Sciences. Research interests: machine learning and data mining. The number of publications – over 200.

email: OSenko@frccsc.ru

ORCID: 0000-0002-5586-3503

Материал поступил в редакцию 2 ноября 2025 года

УДК 004.422+004.9

ПОИСК СЛОВ В РУКОПИСНОМ ТЕКСТЕ НА ОСНОВЕ ШТРИХОВОЙ СЕГМЕНТАЦИИ

И. Д. Морозов¹ [0009-0004-1813-3474], Л. М. Местецкий² [0000-0001-6387-167X]

¹Московский государственный университет имени М. В. Ломоносова,
г. Москва, Россия

²НИУ Высшая школа экономики, г. Москва, Россия

¹morozov-ivan-2003@yandex.ru, ²mestlm@mail.ru

Аннотация

Рукописные архивные документы составляют фундаментальную часть культурного наследия человечества, однако их анализ остается трудоемкой задачей для профессиональных исследователей-историков, филологов и лингвистов. В отличие от коммерческих приложений систем OCR (Optical Character Recognition, оптического распознавания символов), работа с историческими рукописями требует принципиально иного подхода из-за чрезвычайного многообразия почерков, наличия правок и деградации материалов.

Предложен метод поиска в рукописных текстах, основанный на штриховой сегментации. Вместо полного распознавания текста, часто недостижимого для исторических документов, метод позволяет эффективно отвечать на поисковые запросы исследователей. Ключевая идея заключается в декомпозиции текста на элементарные штрихи, формировании семантических векторных представлений с помощью контрастного обучения, последующей кластеризации и классификации для создания адаптивного словаря почерка.

Экспериментально показано, что поиск сравнением кортежей редуцированных последовательностей наиболее информативных штрихов по расстоянию Левенштейна обеспечивает достаточное качество для рассматриваемой задачи. Метод демонстрирует устойчивость к индивидуальным особенностям почерка и вариациям написания, что особенно важно для работы с авторскими архивами и историческими документами.

Предложенный подход открывает новые возможности для ускорения научных исследований в гуманитарной сфере, позволяя сократить время поиска нужной информации с недель до минут, что качественно меняет возможности исследовательской работы с большими архивами рукописных документов.

Ключевые слова: *рукописный текст, поиск, штриховый анализ, сегментация, векторное представление, контрастное обучение, кластеризация.*

ВВЕДЕНИЕ

Автоматизированный поиск в рукописных документах – одна из ключевых задач для исторических архивов. Однако традиционные OCR-системы демонстрируют низкую эффективность из-за нестандартности почерков, деградации носителей и сложной структуры рукописного текста [1, 2].

Перечислим существующие подходы и отметим их ограничения: оптическое распознавание символов (OCR) требует точного определения границ и форм символов, что трудно достижимо в рукописях [3]; поиск по визуальному сходству оперирует целыми фрагментами изображений, обладает высокой вычислительной сложностью и плохо масштабируется [4].

В настоящей работе предложен штриховой подход, который преодолевает указанные ограничения. Его основная идея – это декомпозиция текста на элементарные графические единицы (штрихи) и последующий анализ их устойчивых комбинаций [5–7]. Это позволяет перейти от распознавания символов к выявлению структурных паттернов, специфичных для почерка.

Основные этапы метода:

- Сегментация текста на элементарные штрихи и их предобработка.
- Создание семантических эмбеддингов штрихов с помощью контрастного обучения.
- Кластеризация полученных семантических эмбеддингов для выделения основных типов штрихов и формирования «словаря почерка».
- Классификация штрихов документа, в котором будет проводиться поиск.
- Поиск сравнением кортежей редуцированных последовательностей наиболее информативных штрихов по расстоянию Левенштейна.

ФОРМАЛЬНАЯ ПОСТАНОВКА ЗАДАЧИ

Штрих s задается ломаной линией из k точек (k для каждого штриха разное) в декартовой системе координат:

$$s = \{(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k)\}, \quad (x_m, y_m) \in R^2.$$

Текстовый запрос $q_{\text{text}} = c_1 c_2 \dots c_L$ длины L – это последовательность из L символов алфавита Σ . Здесь Σ – алфавит символов, из которых составляются запросы (например, буквы русского алфавита).

Имеется множество D изображений из N рукописных документов (фрагмент приведен на рис. 1), где каждый документ D_i представлен как кортеж S_i из M_i элементарных штрихов: $S_i = \{s_{i1}, s_{i2}, \dots, s_{iM_i}\}$.

Для многопроцессорных систем с общей оперативной памятью используются более сложные модели организации кэша, которые называются кэш с отслеживанием.

Рис. 1. Фрагмент изображения рукописного документа.

Введем обозначения:

- $C = \bigcup_{i=1}^N S_i$ – все штрихи всех документов;
- $Q_l \subseteq C$ – множество штрихов, соответствующих отдельному символу $c_l \in \Sigma$;
- $P(C)$ – множество всех подмножеств C .

Тогда запрос q , получающийся из текстового запроса q_{text} , представляет собой последовательность $q = (Q_1, Q_2, \dots, Q_L)$.

Для текстового запроса q_{text} необходимо найти такое отображение

$$F: D \times \prod_{l=1}^L P(C) \rightarrow P(C),$$

что

$$\forall D_j \in D \quad F(D_j, q) = \{s \in S_j \mid s \text{ входит в } Q_l\}.$$

Таким образом, по множеству изображений рукописных документов и corteжу штрихов, составляющих запрос, нужно найти corteжи штрихов в документах, относящиеся к поисковому запросу.

ПРИНЦИП ПОИСКА И АЛГОРИТМ КЛАСТЕРИЗАЦИИ

Сегментация

Первым шагом является преобразование исходного изображения документа в набор элементарных штрихов. Для алгоритма поиска неважен метод сегментации. Мы применили подход [6], который анализирует топологию скелета текста. Метод выделяет связные компоненты, соответствующие отдельным движениям пера, и восстанавливает порядок их написания – выделяются элементарные штрихи в виде упорядоченных ломаных линий. Метод основан на анализе топологии скелета текста и обеспечивает сохранение порядка написания штрихов, идентификацию связных компонент как цепочек или циклов. Традиционные методы OCR плохо работают с рукописным текстом из-за вариативности почерков. Разбиение на штрихи позволяет перейти от распознавания целых символов к анализу их составляющих.

Предобработка штрихов

Полученные штрихи – это ломаные линии с переменным числом точек и различным расстоянием между ними. Поэтому выполняем аппроксимацию кубическими сплайнами – каждый штрих заменяется гладкой параметрической кривой, проходящей через все его точки. Это решает три проблемы:

- устраняет ломаный характер линии – настоящие штрихи гладкие;
- дает возможность получения гладкой линии после аугментации штриха – изменения координат составляющих его точек;
- позволяет единообразно семплировать точки (например, 100 точек на штрих).

Семплированные точки преобразуются в бинарные изображения 64×64. Это обусловлено входом выбранной архитектуры нейросети – ResNet [8]. Таким образом, предобработка позволяет превратить штрихи из ломаных в реальные сглаженные изображения следа пера (рис. 2).

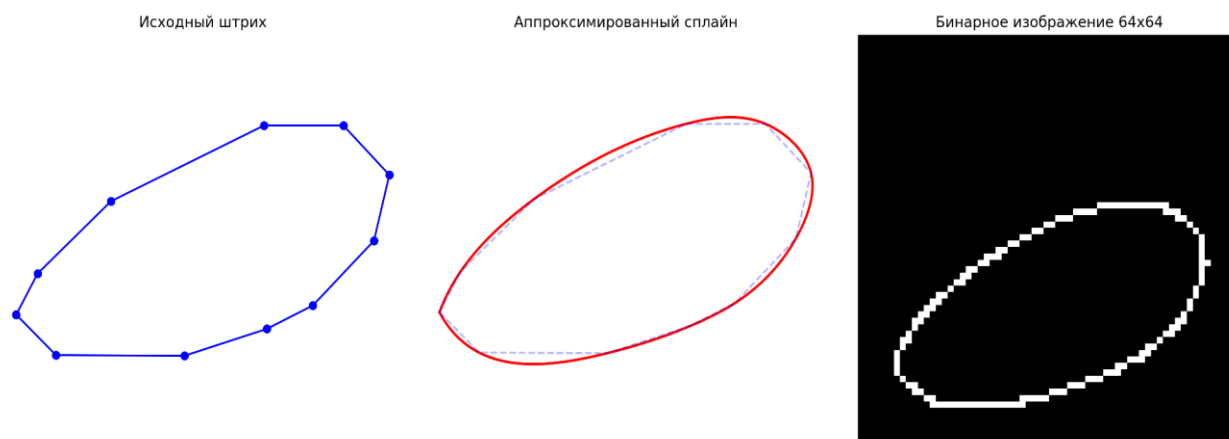


Рис. 2. Преобразование штриха: исходная ломаная, сплайн и растеризованное изображение.

Построение признакового описания штрихов

Для построения эмбеддингов штрихов использована модифицированная архитектура ResNet-18. Одноканальный входной слой адаптирован для черно-белых изображений штрихов вместо стандартных RGB. Каждый блок содержит две свертки 3×3 с пакетной нормализацией и остаточным соединением, что предотвращает затухание градиентов в глубоких слоях. Два полносвязных слоя преобразуют 512-мерные признаки в 128-мерные векторы, которые затем нормируются и проектируются на единичную сферу соответствующей головой. Это позволяет сравнивать штрихи через косинусное расстояние. Модель обучается отличать похожие штрихи от непохожих с помощью контрастной функцией потерь NT-Xent Loss (Normalized Temperature-Scaled Cross Entropy). Для каждого штриха s генерируются две аугментированные версии s_i и s_j . Пример аугментации с добавлением шума и масштабированием показан на рис. 3. Их эмбеддинги v_i и v_j должны стать ближе в векторном пространстве, а эмбеддинги других штрихов – отдалиться.

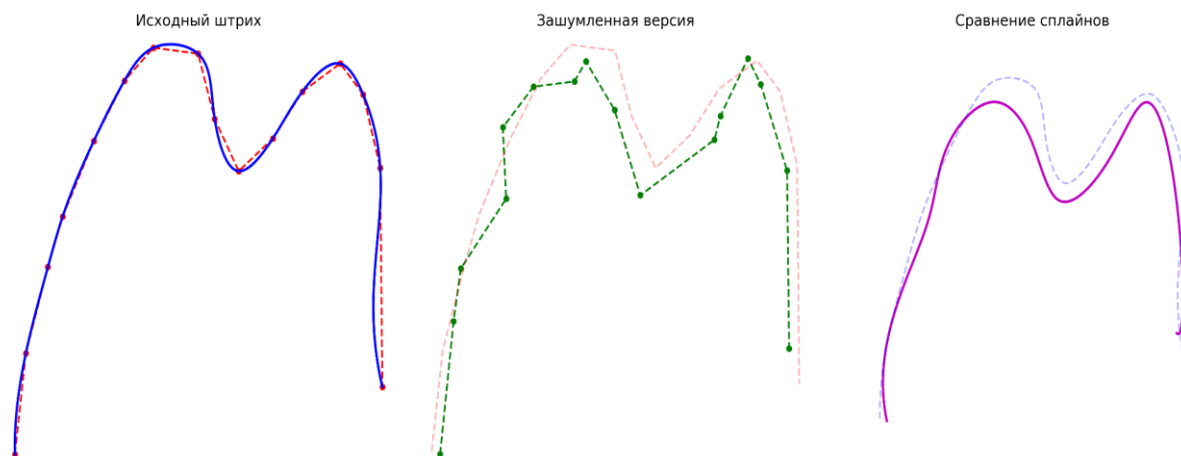


Рис. 3. Аугментация штриха: исходный штрих, зашумленные точки, сплайн аугментированного штриха.

Для батча из N штрихов с двумя аугментациями каждый (всего $2N$ эмбеддингов) функция потерь для i -го штриха вычисляется как

$$L_i = -\log \frac{\exp\left(\frac{v_i^\top v_j}{\tau}\right)}{\sum_{k=1}^{2N} 1_{k \neq i} \exp\left(\frac{v_i^\top v_k}{\tau}\right)},$$

где $v_i^\top v_j$ – косинусное сходство, τ (температура) – гиперпараметр, управляющий «резкостью» распределения (чем меньше τ , тем выше штраф за трудные негативные примеры), $1_{k \neq i}$ – индикатор, исключающий сравнение эмбеддинга с самим собой.

Матрица сходств $V^\top V$ строится для всех $2N$ векторов. Диагональные элементы (сравнение с собой) исключаются маской. Минимизация L сближает эмбеддинги аугментаций одного штриха и разводит разные штрихи. Если v_i и v_j – аугментации одного штриха, а v_k – другого, то

$$\exp(v_i^\top v_j) \gg \exp(v_i^\top v_k) \Rightarrow L_i \rightarrow 0.$$

Сверточные сети эффективно ищут локальные геометрические паттерны, а остаточные связи позволяют обучать глубокие модели без переобучения. Это критично для работы с мелкими деталями штрихов. Принцип работы контрастного обучения проиллюстрирован на рис. 4 (двумерное пространство эмбеддин-

гов для наглядности): похожие штрихи (А и В) сближаются в эмбе́ддинг-пространстве, непохожий штрих (С) отдаляется от них, расстояние определяется косинусной мерой между векторами.

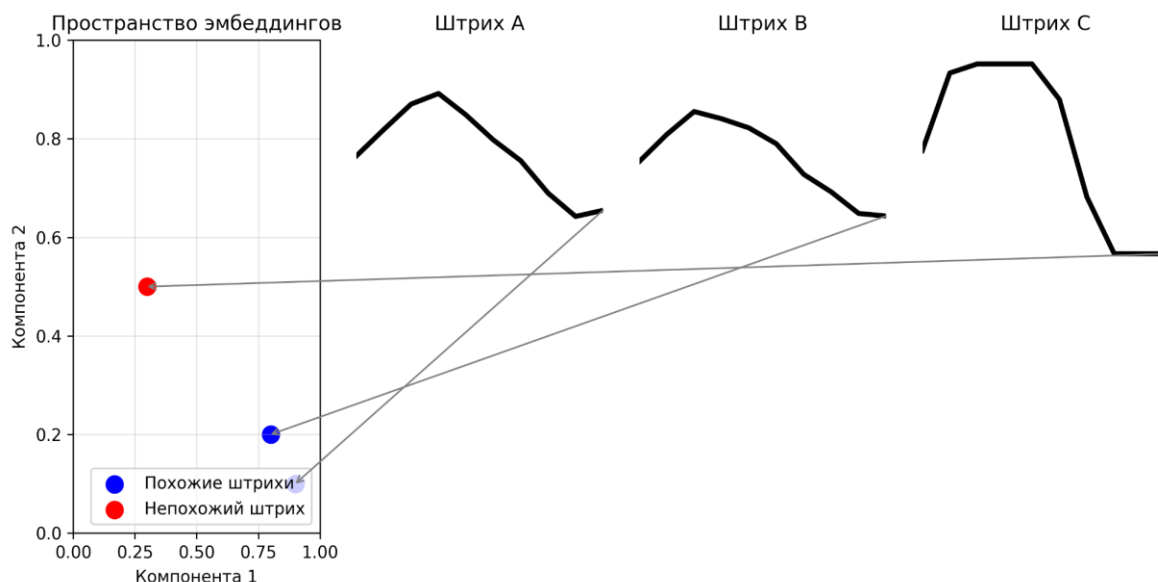


Рис. 4. Принцип контрастного обучения: похожие штрихи А и В; отличающийся штрих С.

Кластеризация и формирование словаря

После преобразования всего архива документов в эмбе́ддинги штрихов (всего M штрихов) $V = \{v_i\}_{i=1}^M \subset R^{128}$ осуществляется группировка семантически близких элементов с помощью алгоритма DBSCAN [9]. Этот метод был выбран из-за его способность и обнаруживать кластеры произвольной формы и идентифицировать выбросы (шум), что соответствует природе рукописных данных, где один и тот же штрих может иметь вариации. Метод основан на критерии плотности распределения точек в пространстве признаков: кластер формируется как максимальное множество точек, где каждая точка имеет не менее minPts соседей в ϵ -окрестности. Формально это свойство задается условиями

$$N_{\epsilon}(v_i) = \{v_j \in V \mid \|v_i - v_j\|_2 \leq \epsilon\}, \quad |N_{\epsilon}(v_i)| \geq \text{minPts}.$$

Для каждого кластера C_k вычисляется эталонный вектор $\mu_k = \frac{1}{|C_k|} \sum_{v \in C_k} v$, играющий роль «цифрового прототипа» каллиграфического элемента. Множество центроидов $M = \{\mu_k\}_{k=1}^K$ образует базовый словарь системы. Поиск по запросу q изначально сводится к решению задачи многокритериальной оптимизации в пространстве эталонов. Сначала каждый символ c_i преобразуется в эмбединг e_i (или их комбинацию) через нейросетевую модель, после чего осуществляется поиск ближайших центроидов:

$$\mu_i^* = \arg \min_{\mu_k \in M} \left(1 - \frac{e_i \mu_k}{\|e_i\| \|\mu_k\|} \right), \quad i = 1, \dots, n.$$

На рис. 5b показано, как разные реализации одного штриха (синие, зеленые, серые, коричневые точки) группируются вокруг общего центра, а рис. 5c демонстрирует механизм поиска через сопоставление с ближайшими эталонами.

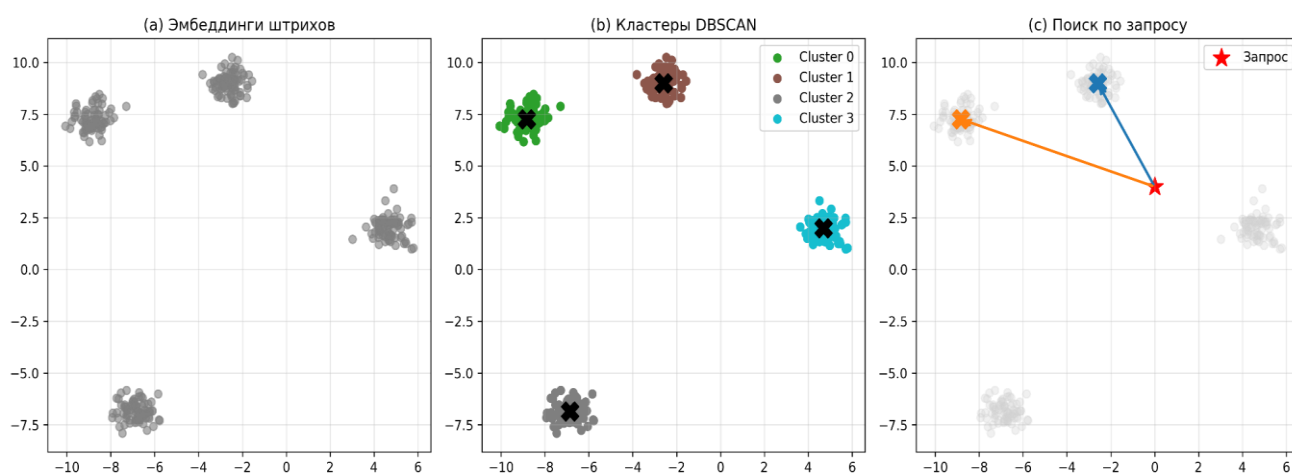


Рис. 5. Визуализация процесса кластеризации: (а) исходное распределение эмбедингов; (б) выделенные кластеры с центрами (черные кресты); (с) поиск по запросу (красная звезда).

ЭКСПЕРИМЕНТЫ

Обучение модели и настройка кластеризации

Прежде чем перейти к поиску, необходимо убедиться в качестве векторных представлений и адекватности словаря.

Процесс обучения нейросетевой модели проводился на выборке из 10000 штрихов, полученных из рукописных конспектов. Каждый штрих предварительно

обрабатывался по схеме: сегментация → сплайновая аппроксимация → растеризация в изображение 64×64 . Для аугментации применялись гауссово зашумление с $\sigma = 1$ и масштабирование в диапазоне $\alpha \in [0,9; 1,1]$.

Архитектура ResNet-18 обучалась с контрастной функцией потерь NT-Xent:

$$L = -\frac{1}{2N} \sum_{i=1}^{2N} \log \frac{\exp\left(\frac{v_i^T v_j}{\tau}\right)}{\sum_{k \neq i} \exp\left(\frac{v_i^T v_k}{\tau}\right)},$$

где температура $\tau = 0.5$ использована для калибровки шкалы сходств, а косинусная мера вычислялась между аугментированными парами. Оптимизация выполнялась алгоритмом Adam с параметром learning rate = $3 \cdot 10^{-4}$. Размер батча составлял 64 примера, каждый из которых содержал две аугментированные версии штриха. Обучение продолжалось 100 эпох с уменьшением потерь от 3.3576 до 3.0479, демонстрируя устойчивую сходимость модели (рис. 6). Характерные колебания потерь (например, локальный максимум 3,0865 на 50-й эпохе при общем тренде снижения) типичны для контрастных методов обучения. Общее время обучения составило 16 мин. на GPU NVIDIA A100.

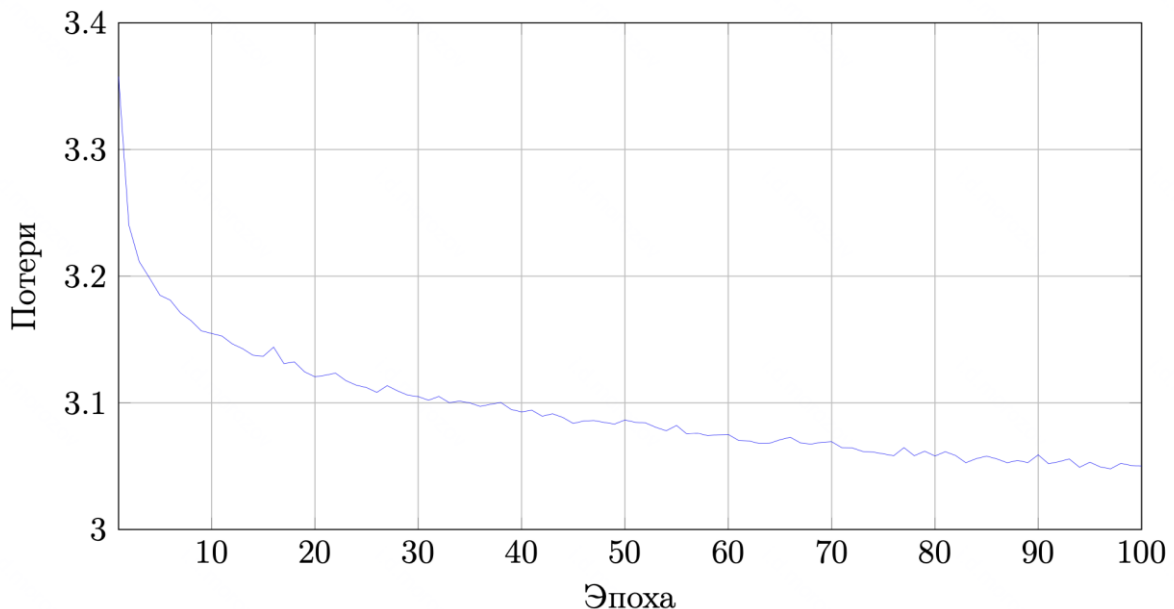


Рис. 6. Динамика функции потерь на обучении.

Для формирования базового словаря элементарных штрихов был проведен анализ зависимости числа кластеров K от параметра ϵ в алгоритме DBSCAN. График $K(\epsilon)$ (рис. 7) демонстрирует три следующих характерных режима.

Область малых значений $\epsilon \in [0.7; 0.78)$. Алгоритм выделяет крупные кластеры, объединяя семантически различные штрихи. Например, при $\epsilon = 0.7$ все штрихи группируются всего в два кластера, что явно недостаточно для описания почерка.

Оптимальный диапазон $\epsilon \in [0.78; 0.94]$. Рост $K(\epsilon)$ отражает обнаружение устойчивых структур. Каждый новый кластер соответствует уникальному типу штриха, удовлетворяющему условию:

$$|N_{\epsilon}(v)| \geq 6 \quad \text{для} \quad v \in C,$$

где $N_{\epsilon}(v)$ – соседи точки v в ϵ -окрестности. При $\epsilon \in [0.9; 0.94]$ достигается максимум $K = 10$, что соответствует набору элементарных компонент почерка.

Область больших значений $\epsilon > 0.94$. Основная причина снижения K при $\epsilon > 0.94$ – это нарушение условия минимальной плотности. Согласно алгоритму DBSCAN кластером считается группа точек, где каждая точка имеет менее чем minPts соседей в ϵ -окрестности и все точки кластера достижимы через цепочку ϵ -соседей.

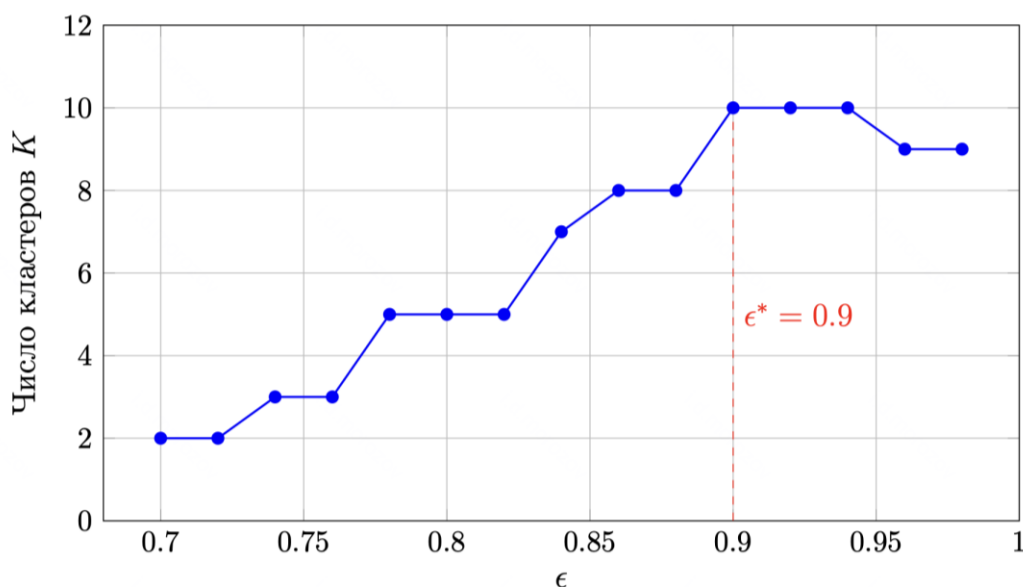


Рис. 7. График зависимости числа кластеров от ϵ . Пунктирной линией обозначено оптимальное значение параметра.

При увеличении ϵ происходит потеря связности – группы распадаются на подкластеры с менее чем \minPts элементами. Например, рассмотрим кластер $C = \{v_1, \dots, v_5\}$ из 5 точек при $\epsilon = 0.96$. Несмотря на выполнение условия связности,

$$\exists v_i \in C: |N_{0.96}(v_i)| = 4 < 6.$$

Согласно правилам DBSCAN весь кластер C классифицируется как шум.

Оптимальное значение $\epsilon^* = 0.92$ выбрано как точка, где достигается насыщение – дальнейшее увеличение ϵ не дает новых кластеров ($K(\epsilon) = 10$ при $\epsilon \geq 0.92$); сохраняется устойчивость: не менее 85% штрихов остаются в тех же кластерах при $\epsilon \pm 0.02$.

Сформированный словарь (см. рис. 8) содержит все необходимые элементы для декомпозиции рукописных символов. Таким образом, предложенная методика позволяет автоматически выделять структурные элементы почерка, обеспечивая баланс между детализацией и устойчивостью представления.

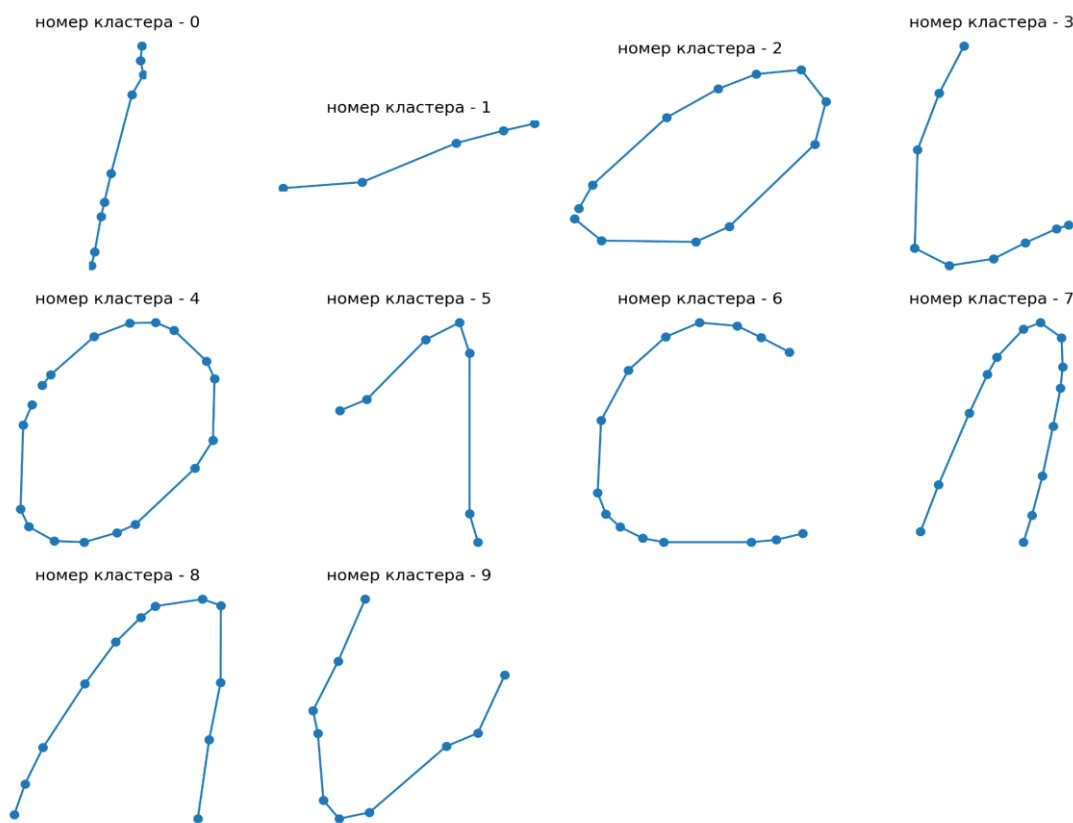


Рис. 8. Базовый словарь из 10 элементарных штрихов.

Модифицированный поиск по кортежам штрихов

Сформированный словарь позволяет искать слова, сопоставляя последовательности штрихов. Однако полная последовательность может быть избыточной и чувствительной к шуму.

В проведенном эксперименте была исследована возможность поиска рукописных слов с использованием редуцированных последовательностей штрихов. Основная гипотеза заключается в том, что для каждого текстового запроса существует оптимальная подпоследовательность штрихов, позволяющая обеспечить поиск при снижении влияния шума менее информативных штрихов. Штрихи, составляющие эти подпоследовательности, назовем главными. Обозначим эти подмножества размера k через $T_{\text{core}}^{(k)}$.

Для каждого типа штриха t_i (после отнесения к кластеру) вычислим дискриминативную силу на основе TF-IDF:

$$DS(t_i) = TF(t_i, q) \times IDF(t_i),$$

где

- N – общее количество штрихов в запросе,
- n_i – количество вхождений штриха t_i в запрос,
- $TF(t_i, q) = n_i/N$ – относительная частота штриха t_i в запросе q ,
- $|D|$ – общее количество строк во множестве документов,
- $|\{d \in D: t_i \in d\}|$ – количество строк, содержащих штрих t_i ,
- $IDF(t_i) = \log \frac{|D|}{|\{d \in D: t_i \in d\}|}$.

Для запроса q и документа d построим редуцированные последовательности штрихов:

$$q_{\text{core}}^{(k)} = \{t \in q: t \in T_{\text{core}}^{(k)}\}, \quad d_{\text{core}}^{(k)} = \{t \in d: t \in T_{\text{core}}^{(k)}\}.$$

Решение о наличии слова в документе примем на основе порогового значения нормализованного расстояния Левенштейна:

$$\text{match} = \begin{cases} 1, & \text{если } \frac{\text{Lev}(q_{\text{core}}, d_{\text{core}})}{\max(|q_{\text{core}}|, |d_{\text{core}}|)} \leq \theta; \\ 0 & - \text{иначе} \end{cases},$$

где θ – пороговое значение. Размер окна выберем, исходя из длины оригинальной (нередуцированной) последовательности штрихов.

Для эксперимента были выбраны слова различной длины и частоты встречаемости в тестовом документе объемом 15267 штрихов. Процедура эксперимента включала последовательное добавление штрихов в поисковый шаблон в порядке убывания дискриминативной силы и вычисление показателей качества для каждого k .

Эксперимент показал, что для всех тестовых слов существует оптимальное подмножество штрихов, обеспечивающее показательное по полноте (Recall) качество поиска при редукции признакового пространства (см. табл. 1).

Табл. 1. Оптимальное k и оценки качества.

Слово	Оптимальное k (количество штрихов)	Точность (Precision)	Полнота (Recall)
«конъюнкция»	6	0.56	0.75
«доказательство»	7	0.6	0.67
«следовательно»	7	0.67	0.83
«теорема»	5	0.45	0.8

Наблюдались характерный рост эффективности с добавлением наиболее информативных штрихов и последующее насыщение после достижения оптимального подмножества, что свидетельствует о наличии небольшого набора высокоинформативных признаков.

Предложенный метод продемонстрировал несколько ключевых преимуществ. Вычислительная эффективность достигается за счет редукции признакового пространства, что особенно важно при работе с крупными архивами рукописных документов. Адаптивность метода позволяет автоматически определять оптимальное подмножество штрихов для каждого слова на основе объективных метрик, без необходимости ручной настройки. Устойчивость к вариациям почерка обеспечивается за счет использования главных штрихов, которые, как пра-

вило, остаются стабильными при различных стилях написания. Снижается чувствительность к «лишним штрихам», вносящим шум. Это свойство особенно ценно при работе с историческими документами, где часто встречаются индивидуальные особенности почерка.

Практическая значимость метода состоит в возможности создания эффективных систем поиска для крупных архивов рукописных текстов. Сокращение вычислительной сложности позволяет масштабировать систему для работы с коллекциями, содержащими миллионы штрихов, что открывает новые возможности для цифровой археографии и исторических исследований.

Следует отметить, что метод имеет определенные ограничения. Качество поиска зависит от точности сегментации текста на элементарные штрихи, которая может оказаться низкой для документов плохого качества или со сложной структурой. Кроме того, для новых слов требуется предварительный анализ дискриминативной силы штрихов, что добавляет этап обучения в рабочий процесс.

ЗАКЛЮЧЕНИЕ

Предложен и экспериментально обоснован штриховой подход к поиску в рукописных документах. Ключевым достижением является создание полноценной системы, которая:

- автоматически сегментирует текст на элементарные штрихи;
- строит их семантические эмбединги с помощью контрастного обучения;
- формирует адаптивный словарь типичных штрихов методами кластеризации без учителя;
- реализует механизм поиска, основанный на классификации штрихов и анализе редуцированных последовательностей классов штрихов.

Показана возможность редукции признакового пространства (штрихов) для достижения приемлемой точности поиска, что открывает пути к созданию более быстрых и эффективных алгоритмов. Универсальность алгоритма (независимость от языка и типа документа) расширяет область его применения.

Перспективы дальнейших исследований включают разработку адаптивных алгоритмов автоматического определения оптимального размера подмножества штрихов, интеграцию контекстной информации для улучшения точности поиска,

а также применение методов глубокого обучения для более точной оценки дискриминативной силы штрихов.

Благодарность

Работа поддержана грантом РНФ №22-68-00066 «Культурное наследие России: интеллектуальный анализ и тематическое моделирование корпуса рукописных текстов».

СПИСОК ЛИТЕРАТУРЫ

1. Zhang X.-Y., Sun Z., Jin L., Ni H. & Lyons T. J. Learning Spatial–Semantic Context with Fully Convolutional Recurrent Network for Online Handwritten Chinese Text Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol. 40, no. 8. P. 1903–1917.
<https://doi.org/10.1109/tpami.2017.2732978>
2. Rahal N., Vögtlin L., Ingold R. Historical Document Image Analysis Using Controlled Data for Pre-training // International Journal on Document Analysis and Recognition (IJDAR). 2023. Vol. 26, no. 3. P. 241–254.
<https://doi.org/10.1007/s10032-023-00437-8>
3. Puigcerver J. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition? // 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). 2017. Vol. 1. P. 67–72.
<https://doi.org/10.1109/ICDAR.2017.20>
4. Rath T. M., Manmatha R. Word Spotting for Historical Documents // International Journal on Document Analysis and Recognition (IJDAR). 2007. Vol. 9, no. 2–4. P. 139–152.
<https://doi.org/10.1007/s10032-006-0027-8>
5. Местецкий Л.М. Непрерывная морфология бинарных изображений: фигуры, скелеты, циркуляры. М.: ФИЗМАТЛИТ. 2009. 231 с.
6. Mestetskiy L.M. Stroke Segmentation of Handwritten Text Based on Medial Representation // Pattern Recognition and Image Analysis: Advances in Mathematical Theory and Applications. 2024. Vol. 34, no. 4. P. 1185–1191.
<https://doi.org/10.1134/S1054661824701256>

7. Dias C. da S., Britto Jr. A. de S., Barddal J. P., Heutte L., Koerich A. L. Pattern Spotting and Image Retrieval in Historical Documents using Deep Hashing. 2022. arXiv:2208.02397
8. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 770–778. <https://doi.org/10.1109/CVPR.2016.90>
9. Ester M., Kriegel H.-P., Sander J., Xu X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // 2nd International Conference on Knowledge Discovery and Data Mining (KDD). 1996. P. 226–231.

WORD SEARCH IN HANDWRITTEN TEXT BASED ON STROKE SEGMENTATION

I. D. Morozov¹ [0009-0004-1813-3474], **L. M. Mestetskiy**² [0000-0001-6387-167X]

^{1, 2}*Lomonosov Moscow State University, Moscow, Russia*

²*Higher School of Economics, Moscow, Russia*

¹*morozov-ivan-2003@yandex.ru*, ²*mestlm@mail.ru*

Abstract

Handwritten archival documents form a fundamental part of humanity's cultural heritage. However, their analysis remains a labor-intensive task for professional researchers, such as historians, philologists, and linguists. Unlike commercial OCR applications, working with historical manuscripts requires a fundamentally different approach due to the extreme diversity of handwriting, the presence of corrections, and material degradation.

This paper proposes a method for searching within handwritten texts based on stroke segmentation. Instead of performing full text recognition, which is often unattainable for historical documents, this method allows for efficiently answering researcher search queries. The key idea involves decomposing the text into elementary strokes, forming semantic vector representations using contrastive learning, followed by clustering and classification to create an adaptive handwriting dictionary.

It is experimentally shown that search by comparing tuples of reduced sequences of the most informative strokes using the Levenshtein distance provides sufficient quality for the task at hand. The method demonstrates resilience to individual handwriting characteristics and writing variations, which is particularly important for working with authors' archives and historical documents.

The proposed approach opens up new possibilities for accelerating scientific research in the humanities, reducing the time required to find relevant information from weeks to minutes, thereby qualitatively transforming research capabilities when working with large archives of handwritten documents.

Keywords: *handwritten text, search, stroke analysis, segmentation, vector representation, contrastive learning, clustering.*

REFERENCES

1. Zhang X.-Y., Sun Z., Jin L., Ni H. & Lyons T. J. Learning Spatial–Semantic Context with Fully Convolutional Recurrent Network for Online Handwritten Chinese Text Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol. 40, no. 8. P. 1903–1917.
<https://doi.org/10.1109/tpami.2017.2732978>
2. Rahal N., Vögtlin L., Ingold R. Historical Document Image Analysis Using Controlled Data for Pre-training // International Journal on Document Analysis and Recognition (IJDAR). 2023. Vol. 26, no. 3. P. 241–254.
<https://doi.org/10.1007/s10032-023-00437-8>
3. Puigcerver J. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition? // 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). 2017. Vol. 1. P. 67–72.
<https://doi.org/10.1109/ICDAR.2017.20>
4. Rath T. M., Manmatha R. Word Spotting for Historical Documents // International Journal on Document Analysis and Recognition (IJDAR). 2007. Vol. 9, no. 2–4. P. 139–152. <https://doi.org/10.1007/s10032-006-0027-8>
5. Mestetskii L.M. Continuous Morphology of Binary Images: Figures, Skeletons, Circulars. M.: FIZMATLIT, 2009. 231 p.

6. *Mestetskiy L.M.* Stroke Segmentation of Handwritten Text Based on Medial Representation // Pattern Recognition and Image Analysis: Advances in Mathematical Theory and Applications. 2024. Vol. 34, no. 4. P. 1185-1191.

<https://doi.org/10.1134/S1054661824701256>

7. *Dias C. da S., Britto Jr. A. de S., Barddal J. P., Heutte L., Koerich A. L.* Pattern Spotting and Image Retrieval in Historical Documents using Deep Hashing. 2022. arXiv:2208.02397

8. *He K., Zhang X., Ren S., Sun J.* Deep Residual Learning for Image Recognition // IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 770–778. <https://doi.org/10.1109/CVPR.2016.90>

9. *Ester M., Kriegel H.-P., Sander J., Xu X.* A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // 2nd International Conference on Knowledge Discovery and Data Mining (KDD). 1996. P. 226–231.

СВЕДЕНИЯ ОБ АВТОРАХ



МОРОЗОВ Иван Дмитриевич – магистр кафедры Математические методы прогнозирования факультета вычислительной математики и кибернетики МГУ имени М. В. Ломоносова. Область научных интересов: машинное обучение, распознавание рукописных текстов, математика.

Ivan Dmitrievich MOROZOV – Master's student at the Department Mathematical Forecasting Methods of the Faculty of Computational Mathematics and Cybernetics of Lomonosov Moscow State University. Research interests: machine learning, handwriting recognition, mathematics.

email: morozov-ivan-2003@yandex.ru

ORCID: 0009-0004-1813-3474



МЕСТЕЦКИЙ Леонид Моисеевич – доктор технических наук, академик РАН, профессор кафедры математических методов прогнозирования МГУ. Научные интересы: вычислительная геометрия, анализ и распознавание изображений.

Leonid Moiseevich MESTETSKIY – Doctor of Engineering Sciences, Academician of the Russian Academy of Natural Sciences, Professor at the Department of Mathematical Forecasting Methods at Moscow State University. His research interests include computational geometry and image analysis and recognition.

email: mestlm@mail.ru

ORCID: 0000-0001-6387-167X

Материал поступил в редакцию 2 ноября 2025 года

УДК 004.422+004.9

АТРИБУЦИЯ АРХИВНЫХ РУКОПИСНЫХ ПИСЕМ С ИСПОЛЬЗОВАНИЕМ СИАМСКИХ НЕЙРОННЫХ СЕТЕЙ

Н. М. Пронина^[0009-0008-1336-4512]

НИУ «Высшая школа экономики», г. Москва, Россия

natalka-pronina@mail.ru

Аннотация

Предложен метод автоматической атрибуции архивных рукописных писем на основе сиамской нейронной сети, решающий ключевую проблему цифровой гуманитаристики – установление авторства исторических документов. Актуальность исследования обусловлена массовой оцифровкой архивов XVII–XIX вв., атрибуция которых затруднена из-за неполных исходных сведений об авторах.

Метод адаптирован к работе с реальным корпусом текстов и учитывает характерные для архивов проблемы: некачественные оцифровки, значительную вариативность почерка и выраженный дисбаланс классов (от 1 до 50 и более образцов на автора). Применение сиамской архитектуры позволяет получать дискриминативные векторные представления, эмбединги, на основе которых выполняется не только классификация документов известных авторов, но и эффективно выявляются рукописи, не принадлежащие ни одному из них. Это сужает круг кандидатов для последующей экспертной проверки.

Представлен алгоритм предобработки данных и проведено сравнительное исследование двух подходов к анализу текста: на уровне фрагментов изображения (300 × 300 пикселей) и уровне отдельных строк. Разработанный инструмент предлагает архивным работникам и филологам эффективное решение для предварительной сортировки и атрибуции крупных массивов рукописных документов.

Ключевые слова: сиамская нейронная сеть, идентификация, верификация, атрибуция, рукописный текст, архивные документы, сверточная нейронная сеть, рекуррентная нейронная сеть.

ВВЕДЕНИЕ

С развитием цифровых технологий все большее значение приобретает автоматизация анализа исторических документов. Особую актуальность приобретает задача атрибуции рукописных текстов – установления авторства и проверки подлинности на основе уникальных характеристик почерка (см., например, рис. 1). Традиционно эта задача решалась экспертами-палеографами, однако автоматические методы анализа могут значительно ускорить и упростить этот процесс, особенно при работе с большими архивами оцифрованных документов.

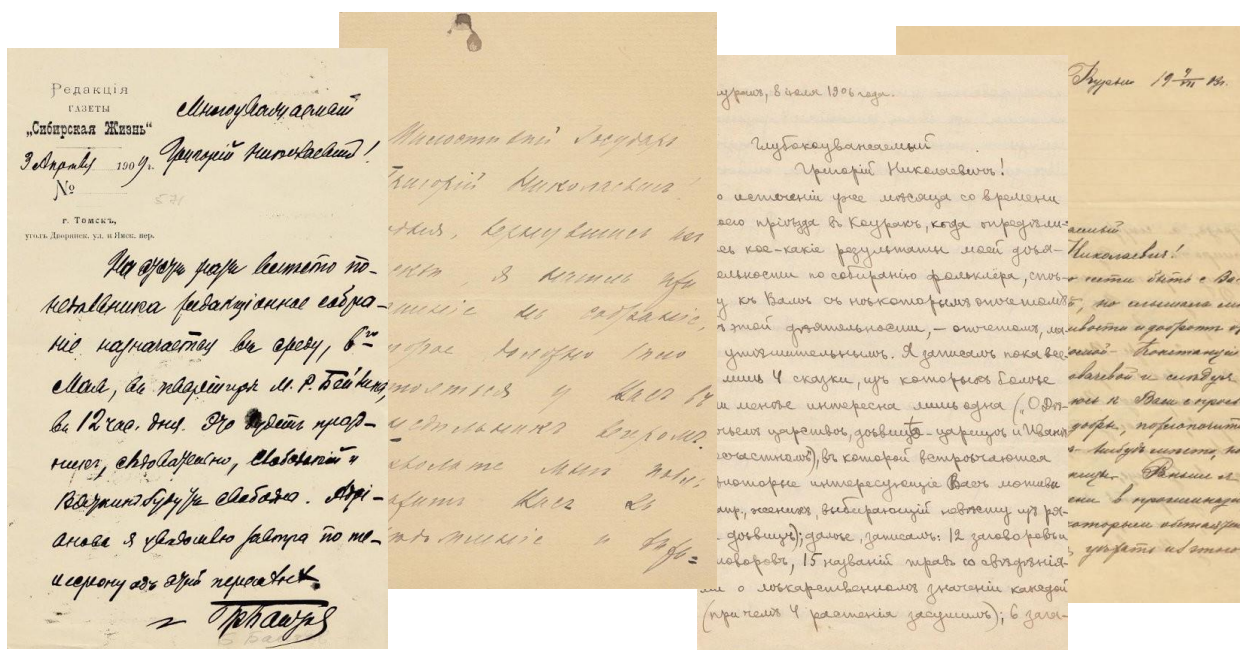


Рис. 1. Письма к Г. Н. Потанину.

Имеется несколько особенностей автоматической атрибуции рукописных текстов:

- дисбаланс количества образцов для разных авторов. Метод должен качественно выделять авторов и с малым количеством примеров в архиве;

- учет индивидуальных особенностей почерка при наличии значительной вариативности даже у одного автора;
- различение авторов со схожими стилями письма;
- обработка документов разного качества.

В настоящей работе предложен метод атрибуции, основанный на использовании сиамских нейронных сетей для сравнения и анализа уникальных характеристик почерка, стиля письма. Такой способ обучения позволяет получать мощные дискриминационные признаки изображения, эмбединги, на основе которых можно произвести качественную классификацию почерка автора, обходя проблемы, описанные выше.

Проведено сравнение двух архитектур:

- картиночной модели на основе сверточной сети ResNet18 [1], анализирующей локальные паттерны [2];
- строчной модели, состоящей из сверточного энкодера и рекуррентного модуля LSTM, который рассматривает строку как последовательность столбцов пикселей.

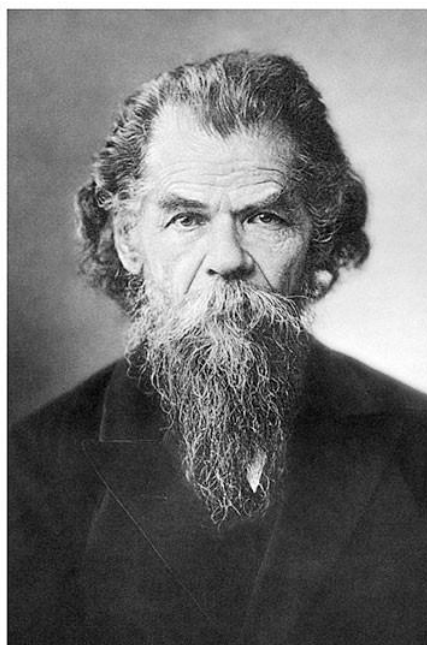
Актуальность настоящей работы обусловлена тем, что машинный поиск текстов с почерком определенного автора в составе больших баз данных растровых изображений рукописных документов, во-первых, значительно расширяет возможности исследователей, в частности литературоведов или историков, в верификации того или иного лица в архивных собраниях текстов. Во-вторых, он дает архивным работникам удобный инструмент для автоматической классификации: ранжированный список сформированных авторов, выявленный программой, может служить основой для описания рукописи и внесения соответствующей информации в архив.

ПОСТАНОВКА ЗАДАЧИ

В качестве объекта исследования взят архив рукописных писем к Григорию Николаевичу Потанину¹. В архиве содержатся сканы первых страниц писем (рис.

¹ Григорий Николаевич Потанин (21 сентября 1835 г. – 30 июня 1920 г.) — русский географ, этнограф, один из основателей и крупнейший идеолог сибирского областничества, член Императорского Русского географического общества (ИРГО), почетный член совета Томского технологического института.

1), адресованных Г. Н. Потанину (рис. 2), от различных авторов. В архиве упоминается 811 авторов, в том числе «неизвестные», для каждого приведено от 1 до 67 писем. Всего писем – 2604. Писем с неизвестным автором – 40.



ПОТАНИН
Григорий Николаевич
Почетный гражданин города Томска

Рис. 2. Григорий Николаевич Потанин.

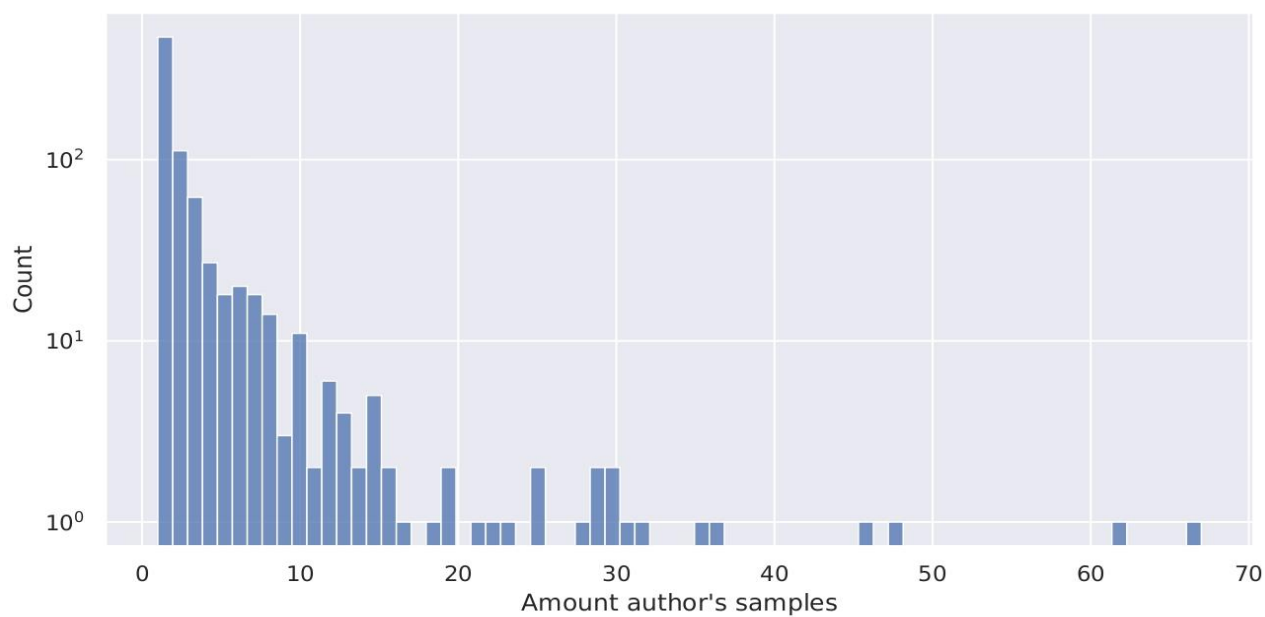


Рис. 3. Распределение количества образцов для каждого автора.

На рис. 3 приведено распределение количества писем. Видно, что подавляющее большинство имеет по одному образцу. Наблюдается сильный дисбаланс классов. Для понимания масштабов проблемы стоит отметить: два и более образца имеют 328 авторов, десять и более – всего 54 автора.

Формальная постановка задачи: требуется разработать алгоритм, входом которого является сканированный неатрибутированный документ, на выходе – ранжированный по убыванию вероятности список возможных его авторов, а также вероятность того, что автор документа неизвестен, т. е. не упоминался в архиве.

ОСНОВНЫЕ ПОНЯТИЯ

Сиамская нейронная сеть

Решить главную проблему – малое количество примеров для большинства авторов – планировалось с помощью алгоритма однократного обучения – *сиамской нейронной сети* (рис. 4). Сиамские сети были впервые представлены в начале 1990-х годов Бромли и Лекуном для решения задачи верификации подписи [3]. Данный тип сети для глубокого обучения использует две или более идентичных подсетей с одинаковой архитектурой, также они используют одни и те же весовые параметры для обучения.

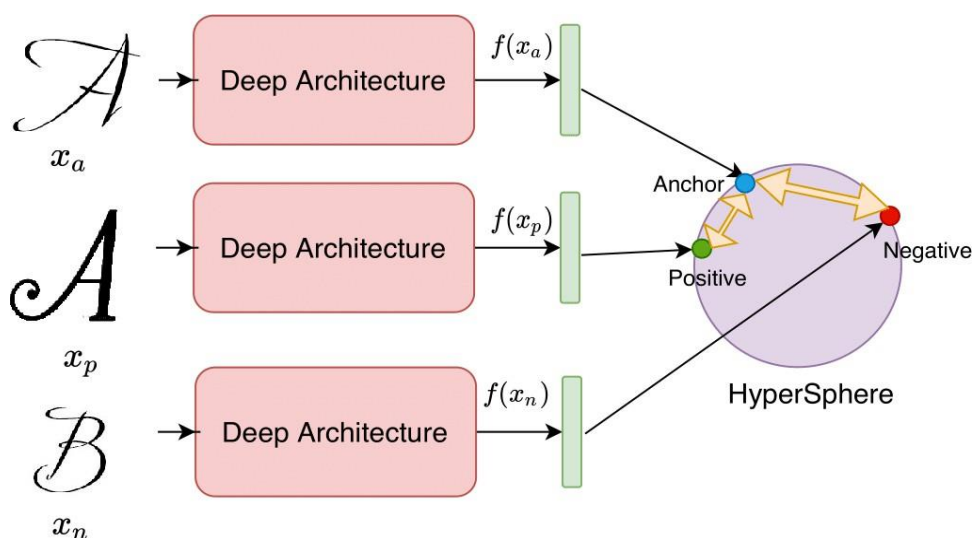


Рис. 4. Обучение сиамской сети с помощью триплетной функции потерь.

Сиамские сети особенно полезны при работе с большим количеством категорий, в каждой из которых представлено лишь несколько объектов. В таких условиях традиционным глубоким сверточным нейросетям не хватает данных для обучения. Кроме того, добавление новых разновидностей потребовало бы изменения архитектуры модели и ее повторного обучения. Вместо этого сиамская сеть решает задачу бинарной классификации пар: определяет, относятся ли два примера к одной группе или разным. Такой подход обеспечивает гибкость и эффективность при ограниченном объеме данных.

Такой вид сетей позволяет получить векторы признаков, эмбединги двух объектов, отражающие их семантическое сходство или различие. Примеры приложений для сиамских сетей: верификация подписи [3], распознавание лиц [4], идентификация перефразирования предложений [5].

Метки класса для авторов получают после обучения простого классификатора на эмбедингах. В настоящей работе использована двухслойная полносвязная нейронная сеть. Аналогичный метод классификации символов был использован в [6].

Две наиболее популярные функции потерь для обучения сиамских нейронных сетей – контрастивная попарная функция потерь (pairwise contrastive loss function) [7, 8] и триплетная функция потерь (triplet loss function) [9].

Контрастивная попарная функция потерь

Контрастивная попарная функция потерь (pairwise contrastive loss function) принимает на вход пару объектов (x и y), каждый из которых относится либо к положительному, либо к отрицательному классу:

$$L(x, y) = (1 - Z(x, y)) d^2(x, y) + Z(x, y) \max(0, \text{margin} - d^2(x, y)),$$

$$Z(x, y) = \begin{cases} 0 & | \ x, y \text{ из одного класса,} \\ 1 & | \ x, y \text{ из разных классов,} \end{cases}$$

$$d(x, y) = \|x - y\|_p.$$

При минимизации такой функции потерь расстояние между объектами одного класса минимизируется, а расстояние между объектами разных классов стремится стать больше заранее заданного отступа (*margin*).

Триплетная функция потерь

Улучшением контрастивной функции потерь является триплетная функция потерь (*triplet loss function*). В отличие от первой здесь используются три объекта: объект рассматриваемого класса; якорь (*anchor*), с которым будет проводиться сравнение; а также два других объекта: принадлежащий к тому же классу, позитив (*positive*), и объект противоположного класса, негатив (*negative*):

$$L(a, p, n) = \max\{d(a, p) - d(a, n) + \textit{margin}, 0\},$$

$$d(x, y) = \|x - y\|_p.$$

При минимизации такой функции потерь расстояние между объектами одного класса (*anchor* и *positive*) уменьшается, и увеличивается расстояние между объектами разных классов (*anchor* и *negative*).

Параметр *margin* – это заранее задаваемый параметр, показывающий, за какую разность расстояний следует штрафовать, т. е. при *margin* = 0 достаточно, чтобы позитивный объект был ближе к якорному, чем негативные. С параметром *margin* = 1 минимизация будет происходить до тех пор, пока позитивный объект не станет ближе, чем негативный, хотя бы на 1.

При обучении модели с триплетной функцией потерь требуется меньше образцов для сходимости, поскольку сеть обновляется одновременно, используя больше объектов, включая как похожие, так и непохожие образцы. Поэтому в работе была использована данная функция потерь.

Перечислим преимущества в задаче атрибуции [9]:

- эффективное обучение при дисбалансе классов;
- формирование компактных кластеров для одного автора;
- устойчивость к вариациям почерка.

Меры качества

В задачах классификации с большим количеством классов стандартные показатели, такие как правильность (accuracy), недостаточно информативны, поскольку получить высокую правильность модели практически невозможно. Например, если всего классов 100 и модель имеет невысокую долю точных угадываний, но при этом верный класс входит в пять самых вероятных классов, то для пользователя такая модель имеет вполне хорошее качество.

Поэтому в задачах с огромным числом категорий часто используют альтернативные показатели из задач ранжирования и рекомендательных систем. Например, $\text{hits}@k$ оценивает, попадает ли истинный класс в первые k значений, предсказанных моделью. Этот показатель позволяет гибко оценить качество модели, не требуя точного совпадения единственного предсказания с истиной.

Определение. $\text{hits}@k$ описывает долю истинных объектов, которые появляются в первых k объектах отсортированного рангового списка:

$$\text{hits}@k = \frac{1}{|I|} \sum_{r \in I} \mathbb{I}[r \leq k],$$

где I – множество рангов. Нетрудно заметить, что $\text{hits}@1$ полностью совпадает с правильностью (accuracy).

В задачах классификации с сильным дисбалансом классов правильность (accuracy) может вводить в заблуждение, т. к. она не учитывает распределение объектов по категориям. Например, если один класс составляет 90% данных, модель с константным предсказанием достигнет точности в 90%, хотя на самом деле она не решает задачу классификации для малых классов. Поэтому в таких случаях предпочтительнее использовать показатели, чувствительные к дисбалансу: точность (precision), чувствительность (recall), F1, а также их макро- (macro) и взвешенные (weighted) варианты. Они позволяют более адекватно оценить качество модели, учитывая как способность правильно детектировать редкие классы (recall), так и точность предсказаний (precision).

Определение. Точность (Precision) – это доля верно предсказанных положительных объектов среди объектов, которые модель отнесла к положительному классу:

$$\text{Precision} = \frac{TP}{TP + FP}.$$

Определение. Чувствительность (Recall) – это доля верно предсказанных положительных объектов среди всех реальных положительных объектов:

$$\text{Recall} = \frac{TP}{TP + FN}.$$

Определение. $F1$ – это гармоническое среднее точности и чувствительности, которое позволяет оценить баланс между ними:

$$F1 = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}.$$

Определение. Precision@k, Recall@k, F1@k – аналоги соответствующих показателей, рассчитанные с поправкой предсказаний модели: если истинный класс находится в первых k самых вероятных классах, то предсказанный моделью класс заменяется на истинный. Далее соответствующий показатель вычисляется на исправленных предсказаниях.

Определение. Macro-Precision@k, Macro-Recall@k, Macro-F1@k – усреднение соответствующей метрики по всем классам, N – число всех классов:

$$\text{Macro - Precision@k} = \frac{1}{N} \sum_{i=1}^N (\text{Precision@k})_i,$$

$$\text{Macro - Recall@k} = \frac{1}{N} \sum_{i=1}^N (\text{Recall@k})_i,$$

$$\text{Macro - F1@k} = \frac{1}{N} \sum_{i=1}^N (F1@k)_i.$$

МЕТОД РЕШЕНИЯ

Бинаризация

Чтобы предотвратить переобучение модели на фон, т. е. плохую обобщающую способность модели по причине того, что в данных присутствует дополнительная информация – фон, от которой не должен зависеть результат, бинаризуем изображения с помощью предобученной нейронной сети DocEnTr [10], состоящей из трансформерного энкодера и декодера (рис. 5). Входное изображение разбивается на фрагменты, которые преобразуются в эмбединги, к ним добавляется информация о местоположении фрагмента. Результирующая последовательность векторов подается в энкодер для получения скрытых представлений, далее полученные скрытые представления подаются в декодер для

получения вектора, который линейно проецируется на векторы пикселей, представляющих участки выходного изображения.



Рис. 5. Бинаризация с помощью DocEnTr.

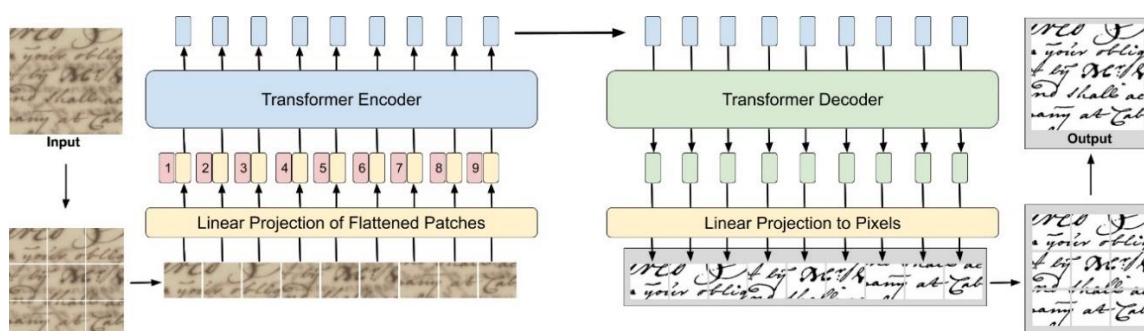


Рис. 6. Архитектура сети для бинаризации (источник: DocEnTr).

На рис. 6 представлен пример бинаризации сложных примеров. Для предотвращения попадания «пустых» изображений в обучающую выборку ауг-

ментация производилась таким образом, чтобы доля белых пикселей была не более 95%.

Нарезка на строки

Для обучения строчной модели необходимо нарезать на строки каждую страницу. Выделение строк производилось с помощью библиотеки *kraken* для Python [11]. В качестве результата выделения одной строки модуль выдает последовательность координат строки, которая описывает область строки, а также ломаную, задающую линию строки. Все эти ломаные показаны на рис. 7 разными цветами для каждой строки. Видно, что для ровного почерка выделение строк выполняется качественно, но, если строки изгибаются, алгоритм разрезает на экстремально маленькие подстроки, что мешает обучению строчной сети. Возможно, эту проблему можно решить постобработкой результата: слиянием подстрок в одну строку. Наивное решение – слияние строк на основании близости ординат – в некоторых случаях приводило к неправильному слиянию с чужой строкой. Поэтому было принято решение не делать постобработку, а просто удалить выбросы из выборки.



Рис. 7. Выделение строк с помощью библиотеки *kraken*.

По распределению длин строк (рис. 8) видно, что библиотека *kraken* выдает очень много коротких строк. Второй пик отражает правильно выделенные строки (средняя ширина изображения как раз около 500 пикселей).

В распределении высот строк (рис. 9) также наблюдаются выбросы. Так, алгоритм выделяет строки, написанные вертикально. Они также удаляются.

Постобработка, использованная в работе, состоит в следующем:

- удаление всех строк с шириной менее 300 пикселей;
- удаление всех строк высотой более 100 пикселей;
- линейное приведение (сжатие/расширение) всех строк к высоте 50 пикселей.

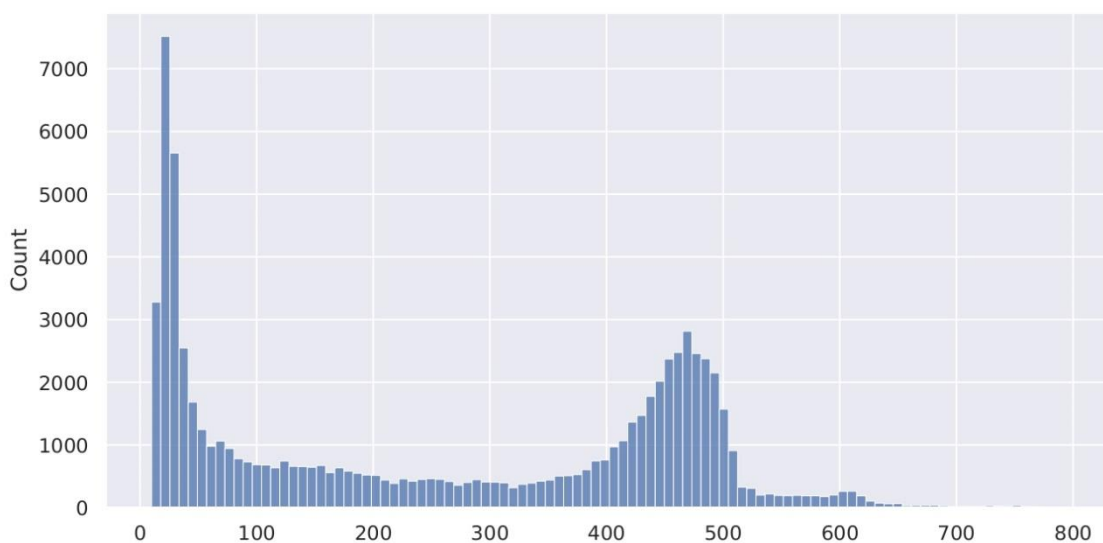


Рис. 8. Распределение длин строк.

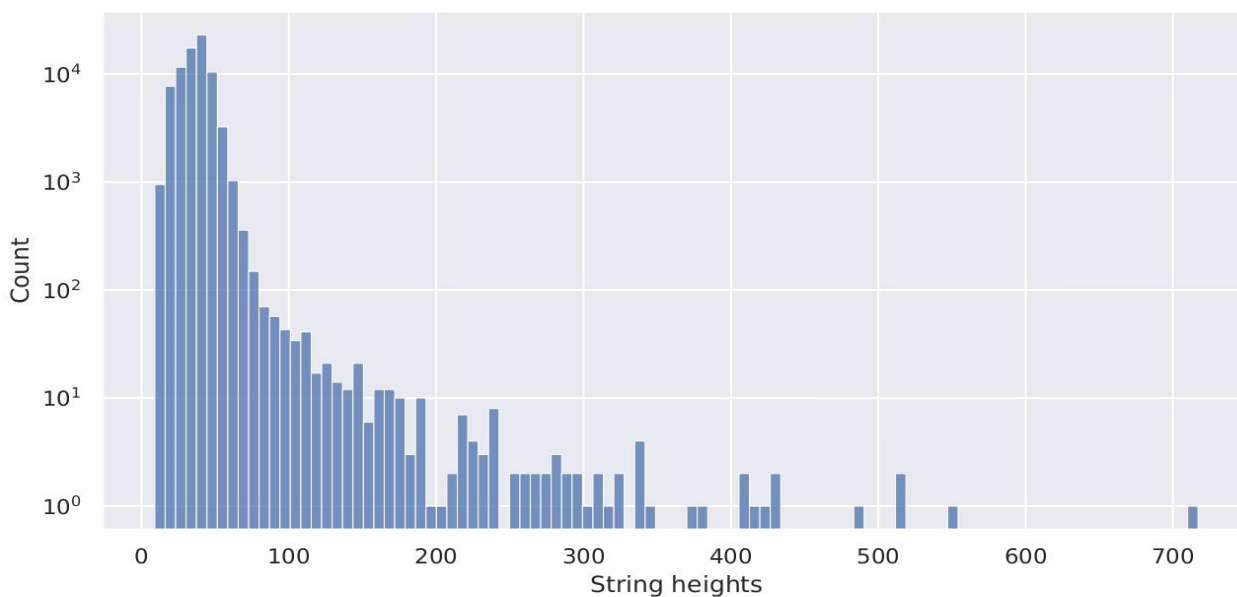


Рис. 9. Распределение высот строк.

Картиночная модель

Была применена идея обучения сиамской сети на бинарных изображениях, фрагментах размера 300×300 пикселей. Был взят предобученный ResNet18. Последний слой имеет размерность 1000 на выходе, поэтому сиамская сеть выучила 1000-размерный эмбединг изображения. Обучение проводилось только для последних двух слоев, 513000 обучаемых параметров.

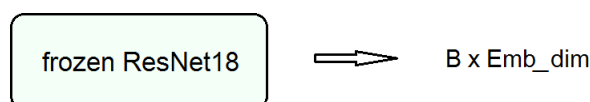


Рис. 10. Архитектура картиночной модели.

При обучении в сеть будут подаваться тройки изображений, триплеты: якорь (anchor), позитив (positive), негатив (negative). В качестве якоря подается очередная строка, в качестве позитива – случайная строка того же автора, в качестве негатива – случайная строка другого автора. При оценке качества сиамской сети правильность (ассигасу) вычисляется как доля триплетов, для которых евклидово расстояние между эмбедингами якоря и позитива меньше, чем между эмбедингами якоря и негатива.

На полученных эмбедингах обучим классификатор: двухслойную полносвязную нейронную сеть с 513538 параметрами.

Итоговое предсказание для письма получено с помощью агрегации выходов сети для нескольких случайных фрагментов этого письма. В качестве функции агрегации было выбрано усреднение.

Строчная модель

В качестве строчной модели была выбрана гибридная модель, сочетающая сверточные и рекуррентный слои. Для обработки последовательностей строк она обучалась со случайной инициализацией.

Сверточный блок состоит из трех слоев, включающих: свертку, нормализацию по батчу, активацию и max-пуллинг с асимметричным шагом. Последний выбран, чтобы по высоте строки сжимать меньше, чем по ширине. В качестве

рекуррентного слоя была выбрана однонаправленная LSTM (205507 обучаемых параметров).

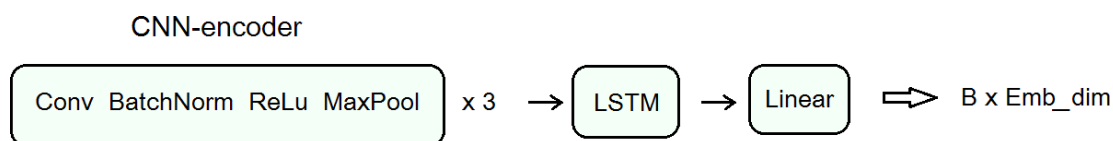


Рис. 11. Архитектура строчной модели.

Классификатор тот же самый – 513538 параметров.

Итоговое предсказание для письма производилось с помощью усреднения выходов сети для всех строк данного письма.

ЭКСПЕРИМЕНТЫ

Были проведены эксперименты с разным числом классов и способом обучения. Всюду далее термин *сбалансированный (balanced)* указывает на то, что при обучении модели объекты подавались равновероятно для разных классов. По умолчанию же использован обычный способ – каждый объект подается равновероятно, поэтому классы с бóльшим числом примеров при обучении встречаются чаще остальных. В таком смысле обучение является *несбалансированным*.

Картиночная модель

Картиночная модель обучается классифицировать фрагменты рукописных документов размера 300 × 300 пикселей, но итоговое предсказание для изображения производится на основе нескольких случайных фрагментов. Было проведено исследование, с целью выяснить какое количество изображений оптимально для качественного определения автора данного письма.

В табл. 1 представлена зависимость качества картиночной модели от числа фрагментов в задачах классификации на 3, 10, 20, 100 классов. Можно заметить, что увеличение количества фрагментов при агрегации практически всегда приводит к увеличению и точности, и *F1*. Поэтому всюду далее используется максимальное число фрагментов для классификации одного документа – 20 шт.

Табл. 1. Accuracy и F1 для картиночной модели

Windows num \ Metric	1	5	10	20
3 classes, Hits@1	0.9209	0.9379	0.9322	0.9492
3 classes, Macro-F1@1	0.9203	0.9385	0.9338	0.9510
10 classes, Hits@1	0.7073	0.7878	0.7854	0.8171
10 classes, Macro-F1@1	0.6856	0.7766	0.7768	0.8091
20 classes, Hits@1	0.6416	0.7621	0.7731	0.7731
20 classes, Macro-F1@1	0.6198	0.7363	0.7598	0.7566
100 classes, Hits@1	0.4415	0.5549	0.5650	0.5882
100 classes, Macro-F1@1	0.3208	0.4239	0.4286	0.4624

Был проведен такой же эксперимент со сбалансированным обучением сети для изучения, какой эффект оно окажет на *F1*. Сравнение табл. 1 и 2 показывает, что сбалансированное обучение дает незначительное улучшение в редких случаях (такие случаи выделены жирным шрифтом), в основном результат становится немного хуже. Вероятно, это объясняется тем, что малые классы и так легко детектируются, поэтому увеличение их частоты встречаемости при обучении не дает сильного прироста качества.

Табл. 2. Accuracy и F1 для картиночной модели при сбалансированном обучении

Windows num \ Metric	1	5	10	20
3 classes, Hits@1	0.9209	0.9209	0.9435	0.9209
3 classes, Macro-F1@1	0.9217	0.9217	0.9469	0.9220
10 classes, Hits@1	0.6854	0.7488	0.7732	0.7634
10 classes, Macro-F1@1	0.6856	0.7408	0.7656	0.7567
20 classes, Hits@1	0.5696	0.6682	0.6854	0.6964
20 classes, Macro-F1@1	0.5496	0.6331	0.6584	0.6705
100 classes, Hits@1	0.3642	0.4783	0.4993	0.5094
100 classes, Macro-F1@1	0.3146	0.4251	0.4457	0.4609

Строчная модель

Были проведены эксперименты для строчной модели. В табл. 3 представлено сравнение моделей. Для картиночной модели приведены лучшие результаты, с 20 окнами, при несбалансированном обучении. Для строчной модели уже виден прирост в качестве при переходе к сбалансированному обучению (выделено жирным шрифтом), особенно заметен прирост в $F1$ в многоклассовой классификации. В большинстве случаев строчная модель справляется лучше, но по качеству классификации на 100 классов все же уступает.

Табл. 3. Сравнение показателей для картиночной и строчной моделей

Metric \ Model	10 classes			20 classes			100 classes		
	win	str	bal	win	str	bal	win	str	bal
Hits@1	0.817	0.657	0.694	0.773	0.552	0.515	0.588	0.526	0.582
Hits@2	0.915	0.813	0.851	0.889	0.761	0.727	0.692	0.657	0.712
Hits@3	0.946	0.888	0.908	0.930	0.832	0.815	0.758	0.741	0.771
Hits@4	0.968	0.935	0.943	0.955	0.880	0.859	0.808	0.796	0.812
Hits@5	0.980	0.965	0.960	0.969	0.919	0.905	0.852	0.835	0.847
Macro-F1@1	0.809	0.585	0.676	0.757	0.484	0.497	0.462	0.395	0.579
Macro-F1@2	0.913	0.786	0.843	0.884	0.724	0.722	0.594	0.564	0.708
Macro-F1@3	0.948	0.884	0.908	0.927	0.801	0.812	0.681	0.674	0.765
Macro-F1@4	0.969	0.938	0.946	0.954	0.861	0.861	0.745	0.743	0.804
Macro-F1@5	0.981	0.968	0.962	0.968	0.904	0.904	0.796	0.793	0.840

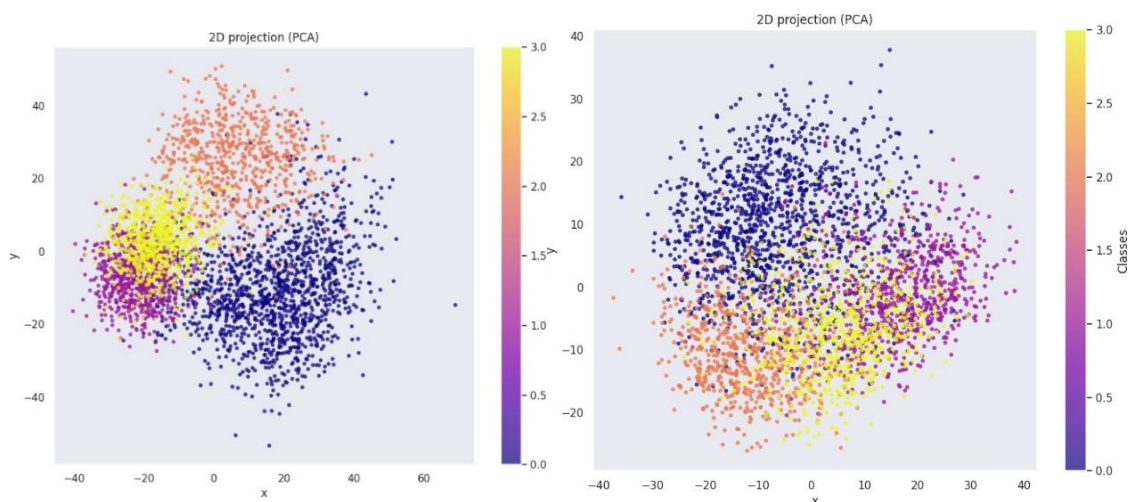
Для картиночной модели была использована предобученная сеть и с бóльшим числом параметров (205507 против 513000). Были проведены эксперименты еще с двумя архитектурами строчной модели: предобученным ResNet18 и измененной архитектурой (10), с 4 слоями в энкодере вместо 3, с двунаправленным LSTM-блоком (770936 обучаемых параметров).

Табл. 4. Сравнение метрик для разных архитектур строчных моделей

Metric \ Model	10 classes			20 classes			100 classes		
	bal	res	str2	bal	res	str2	bal	res	str2
Hits@1	0.694	0.968	0.973	0.515	0.862	0.910	0.582	0.715	0.782
Hits@2	0.851	0.990	0.990	0.727	0.943	0.957	0.712	0.818	0.876
Hits@3	0.908	0.990	0.990	0.815	0.965	0.971	0.771	0.855	0.921

Hits@4	0.943	0.998	0.998	0.859	0.975	0.981	0.812	0.879	0.943
Hits@5	0.960	0.998	0.998	0.905	0.983	0.987	0.847	0.901	0.965
Macro-F1@1	0.676	0.967	0.972	0.497	0.860	0.894	0.579	0.739	0.727
Macro-F1@2	0.843	0.991	0.990	0.722	0.945	0.950	0.708	0.836	0.849
Macro-F1@3	0.908	0.991	0.990	0.812	0.968	0.968	0.765	0.871	0.902
Macro-F1@4	0.946	0.997	0.997	0.861	0.976	0.980	0.804	0.893	0.935
Macro-F1@5	0.962	0.997	0.997	0.904	0.983	0.988	0.840	0.911	0.961

В табл. 4 представлены лучшие результаты строчной модели при сбалансированном обучении и лучшие результаты двух новых строчных моделей. Для ResNet качество лучше при сбалансированном обучении, для большой LSTM модели – при несбалансированном, но в обоих случаях качество заметно выше,



чем у картиночной.

Рис. 12. Две главные компоненты для несбалансированного и сбалансированного обучения сиамской сети.

Кроме того, по визуализации проекции эмбедингов большой LSTM-модели на двумерную плоскость (с помощью метода главных компонент) видно, что сбалансированное обучение ухудшает разделимость эмбедингов строк для четырех самых больших классов (рис. 12).

Определение неизвестного класса

После обучения модели на задачу классификации с функцией потерь перекрестной энтропии, модель не в состоянии «отказаться от классификации», если встречает объект класса, который не был представлен в обучающих дан-

ных, но в текущей постановке задачи обязательно нужно иметь такую возможность. Такая классификация называется классификацией в открытом мире (open-world classification), или открытой классификацией (open classification).

Идея перехода к открытой классификации заключается в том, чтобы заменить стандартный softmax-слой, т. к. функция softmax по умолчанию распределяет 100% вероятностей между известными классами, не оставляя места для «неизвестных». Поэтому softmax был заменен на функцию сигмоиды, по одной на каждый класс, функция потерь – на бинарную перекрестную энтропию [12]. Таким образом, задача мультиклассовой классификации, в которой softmax дает взаимоисключающие вероятности, перешла в задачу множественной бинарной классификации, где сигмоиды позволяют объекту принадлежать к нескольким классам или ни к одному.

Итоговое предсказание модели построено следующим образом: если сигмоиды всех классов меньше заранее заданного порога (использовался порог, равный 0.5), то объект классифицируется как неизвестный, иначе выдается класс с наибольшим значением сигмоиды.

Эксперименты показали, что подобная замена последнего softmax-слоя и функции потерь практически никак не влияет на меры качества, полученные на обучающей и валидационных выборках.

Анализ предсказаний

Была визуализирована матрица ошибок для строчной модели, показавшей наивысшее качество, – большой LSTM-модели, чтобы зафиксировать самые частые перепутывания авторов. На рис. 13 видно, что ошибки распределены равномерно по строкам и столбцам, значит, модель одинаково хорошо разделяет всех авторов. Самый яркий пример ошибки – модель атрибутирует некоторые документы Антоновой Валентины Константиновны Киселевой Ольге, но на рис. 14 видно, что эти почерки и правда похожи.

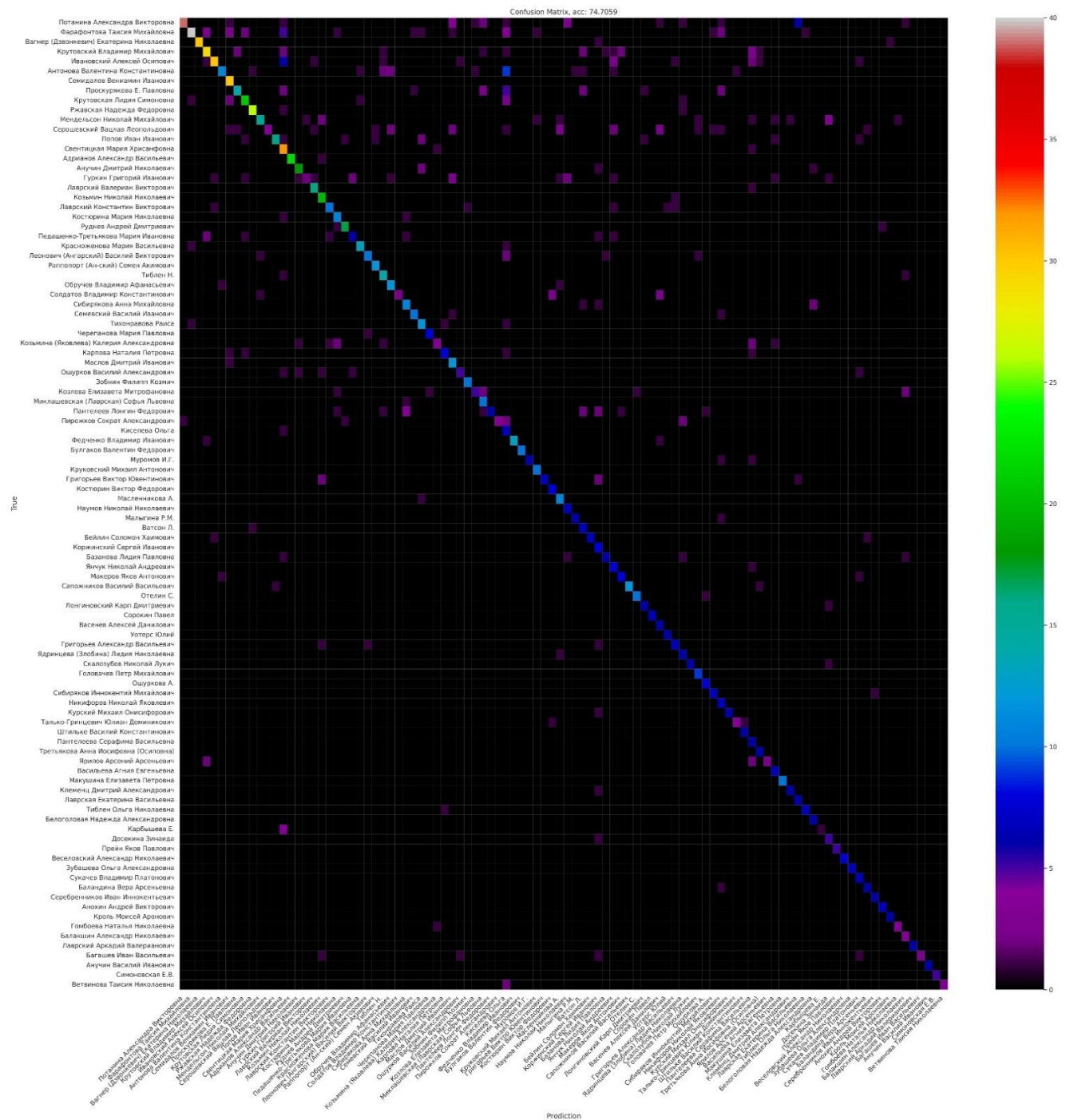


Рис. 13. Матрица ошибок большой LSTM-модели.

Далее были получены вероятности для 40 писем с неизвестным автором. Для большинства изображений максимальная вероятность авторства получилась меньше 30%. Выделяются только два изображения с вероятностью более 85% (рис. 15). Названия типа «КККМ ОФ 9999:9999» обозначают музейный шифр конкретного письма.

КККМ ОФ 7928:2707

0.2767 Проскурякова Е. Павловна
0.2403 Маслов Дмитрий Иванович
0.0958 Потанина Александра Викторовна
0.0691 Веселовский Александр Николаевич
0.0463 Свентицкая Мария Хрисанфовна

КККМ ОФ 7928:2707

0.3495 Проскурякова Е. Павловна
0.1253 Маслов Дмитрий Иванович
0.1250 **Неустановленное лицо**
0.0908 Наумов Николай Николаевич
0.0721 Потанина Александра Викторовна

Рис. 15. Результаты для писем с неизвестным автором.

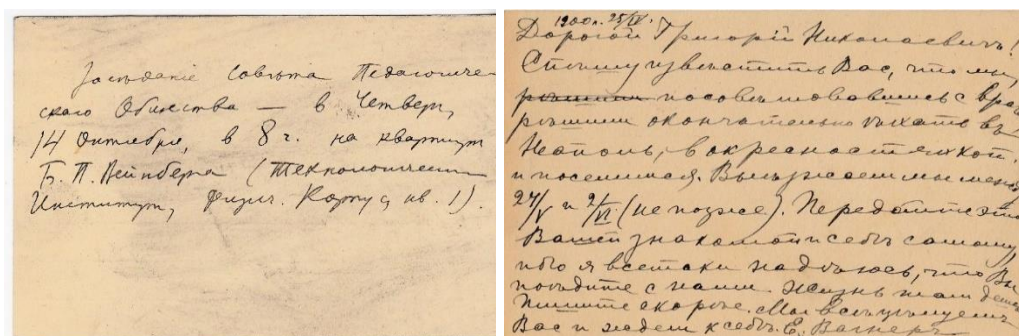


Рис. 16. КККМ ОФ 7928:2480 и почерк Е. Н. Вагнер (Дзвонкевич).

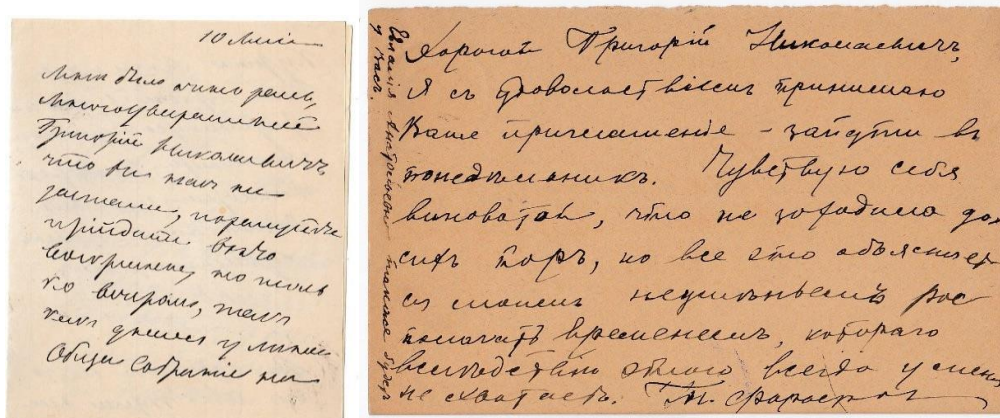


Рис. 17. КККМ ОФ 7928:2719 и почерк Т. М. Фарафоновой.

На рис. 16 и 17 показано сравнение с почерками соответствующих авторов из размеченных данных. Более детальный анализ эксперта [13] признал оба прогноза реалистичными.

ЗАКЛЮЧЕНИЕ

Предложен подход к задаче атрибуции рукописных писем, основанный на использовании сиамской нейронной сети для сравнения и анализа уникальных характеристик почерка, стиля письма. Предложенное решение позволяет не только сравнить рукописи и определить принадлежность их известным авторам, но и выделить документы, чьи создатели отсутствуют в архиве.

Основным результатом является алгоритм, входом которого является сканированный неатрибутированный документ, а на выходе выдается ранжированный по убыванию вероятности список возможных его авторов, а также вероятность того, что автор документа неизвестен, т. е. не упоминался в архиве. Полученное решение задачи атрибуции включает предобработку сканированных документов и качественное обучение сиамских нейронных сетей. Существенным результатом работы является решение задачи классификации авторов на реальном корпусе текстов с реальными недостатками, с которыми сталкиваются исследователи при работе с архивами: некачественное сканирование, значительная вариативность написания даже для рукописей одного автора, большой дисбаланс классов.

Исследованы два варианта анализа сканированного текста: анализ фрагментов изображения и анализ каждой строки рукописного текста отдельно. В первом случае, для картиночной модели, в качестве архитектуры сиамской сети использован ResNet18; во втором случае, для строчной модели, использована гиб-ридная модель, сочетающая сверточные и рекуррентный слои для обработки последовательностей строк.

Эксперименты показали, что модели устойчивы к дисбалансу данных, способны работать даже с малым числом образцов почерка, показывая высокое значение $F1$, подтвердили практическую применимость метода для задач архивной атрибуции.

Благодарности

Работа поддержана грантом РНФ № 22-68-00066 «Культурное наследие России: интеллектуальный анализ и тематическое моделирование корпуса рукописных текстов».

СПИСОК ЛИТЕРАТУРЫ

1. He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 770–778. <https://doi.org/10.1109/CVPR.2016.90>.
2. Kiselev V., Kropotov D., Pronina N. Handwritten documents author verification based on the siamese network // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2024. Vol. XLVIII-2/W5-2024. P. 73–78.
<https://doi.org/10.5194/isprs-archives-XLVIII-2-W5-2024-73-2024>
3. Bromley J., Bentz J., Bottou L., Guyon I., Lecun Y., Moore C., Sackinger E., Shah R. Signature verification using a "siamese" time delay neural network // International Journal of Pattern Recognition and Artificial Intelligence. 1993. Vol. 7, No. 4. P. 669–688. <https://doi.org/10.1142/S0218001493000339>
4. Solomon E., Woubie A., Emiru E.S. Deep learning-based face recognition method using siamese network. 2024. <https://doi.org/10.48550/arXiv.2312.14001>
5. Yin W., Schütze H. Convolutional neural network for paraphrase identification // Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015. P. 901–911. <https://doi.org/10.3115/v1/N15-1091>
6. Koch G., Zemel R., Salakhutdinov R. et al. Siamese neural networks for one-shot image recognition // ICML Deep Learning Workshop. 2015. Vol. 2, No. 1. P. 1–30.
7. Chopra S., Hadsell R., LeCun Y. Learning a similarity metric discriminatively, with application to face verification // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. Vol. 1. P. 539–546.
<https://doi.org/10.1109/CVPR.2005.202>
8. Hadsell R., Chopra S., LeCun Y. Dimensionality reduction by learning an invariant mapping // 2006 IEEE Computer Society Conference on Computer Vision

and Pattern Recognition (CVPR'06). 2006. Vol. 1. P. 1735–1742.

<https://doi.org/10.1109/CVPR.2006.100>

9. *Schroff F., Kalenichenko D., Philbin J.* Facenet: A unified embedding for face recognition and clustering // 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015. P. 815–823.

<https://doi.org/10.1109/CVPR.2015.7298682>

10. *Souibgui M.A., Biswas S., Jemni S.K., Kessentini Y., Fornés A., Llado's J., Pal U.* Docentr: An end-to-end document image enhancement transformer. 2022. P. 1699–1705. <https://doi.org/10.1109/ICPR56361.2022.9956101>.

11. *Wood D.E., Salzberg S.L.* Kraken: ultrafast metagenomic sequence classification using exact alignments // *Genome Biology*. 2014. Vol. 15, No. 1. P. R46. <https://doi.org/10.1186/gb-2014-15-3-r46>

12. *Shu L., Xu H., Liu B.* Doc: Deep open classification of text documents. 2017. P. 2911–2916. <https://doi.org/10.18653/v1/D17-1314>.

13. *Киселев В.С., Пронина Н.М.* Машинная атрибуция почерка в решении источниковедческих проблем (на материале переписки Г. Н. Потанина) // *Имагология и компаративистика*. 2025. № 24.

ARCHIVAL HANDWRITTEN LETTER ATTRIBUTION USING SIAMESE NEURAL NETWORKS

N. M. Pronina^[0009-0008-1336-4512]

National Research University Higher School of Economics, Moscow, Russia

natalka-pronina@mail.ru

Abstract

This paper presents a method for the automated attribution of archival handwritten letters based on a Siamese neural network, addressing a key challenge in digital humanities – the authentication of historical documents. The research is motivated by the mass digitization of 17th to 19th-century archives, where attribution is often hindered by incomplete or inaccurate metadata about the authors.

The method is designed for real-world document collections and accounts for challenges typical of archival materials: poor-quality scans, significant handwriting

variation, and substantial class imbalance (from 1 to over 50 samples per author). The use of a Siamese network architecture enables the extraction of discriminative vector representations (embeddings). Based on these embeddings, the method not only classifies documents by known authors but also effectively identifies manuscripts that do not match any known author in the archive. This significantly narrows down the pool of candidates for subsequent expert verification.

The study introduces a data preprocessing algorithm and provides a comparative analysis of two approaches to text analysis: at the image fragment level (300×300 px) and at the individual text line level. The developed tool offers archivists and philologists an effective solution for the preliminary sorting and attribution of handwritten documents large collections.

Keywords: *siamese neural network, identification, verification, attribution, handwritten text, archival documents, convolutional neural network, recurrent neural network.*

REFERENCES

1. He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 770–778. <https://doi.org/10.1109/CVPR.2016.90>.
2. Kiselev V., Kropotov D., Pronina N. Handwritten documents author verification based on the siamese network // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2024. Vol. XLVIII-2/W5-2024. P. 73–78.
<https://doi.org/10.5194/isprs-archives-XLVIII-2-W5-2024-73-2024>
3. Bromley J., Bentz J., Bottou L., Guyon I., Lecun Y., Moore C., Sackinger E., Shah R. Signature verification using a "siamese" time delay neural network // International Journal of Pattern Recognition and Artificial Intelligence. 1993. Vol. 7, No. 4. P. 669–688. <https://doi.org/10.1142/S0218001493000339>
4. Solomon E., Woubie A., Emiru E.S. Deep learning-based face recognition method using siamese network. 2024. <https://doi.org/10.48550/arXiv.2312.14001>
5. Yin W., Schütze H. Convolutional neural network for paraphrase identification // Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015.

P. 901–911. <https://doi.org/10.3115/v1/N15-1091>

6. Koch G., Zemel R., Salakhutdinov R. et al. Siamese neural networks for one-shot image recognition // ICML Deep Learning Workshop. 2015. Vol. 2, No. 1. P. 1–30.

7. Chopra S., Hadsell R., LeCun Y. Learning a similarity metric discriminatively, with application to face verification // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. Vol. 1. P. 539–546. <https://doi.org/10.1109/CVPR.2005.202>

8. Hadsell R., Chopra S., LeCun Y. Dimensionality reduction by learning an invariant mapping // 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). 2006. Vol. 1. P. 1735–1742. <https://doi.org/10.1109/CVPR.2006.100>

9. Schroff F., Kalenichenko D., Philbin J. Facenet: A unified embedding for face recognition and clustering // 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015. P. 815–823. <https://doi.org/10.1109/CVPR.2015.7298682>

10. Souibgui M.A., Biswas S., Jemni S.K., Kessentini Y., Fornés A., Llado's J., Pal U. Docentr: An end-to-end document image enhancement transformer. 2022. P. 1699–1705. <https://doi.org/10.1109/ICPR56361.2022.9956101>.

11. Wood D.E., Salzberg S.L. Kraken: ultrafast metagenomic sequence classification using exact alignments // Genome Biology. 2014. Vol. 15, No. 1. P. R46. <https://doi.org/10.1186/gb-2014-15-3-r46>

12. Shu L., Xu H., Liu B. Doc: Deep open classification of text documents. 2017. P. 2911–2916. <https://doi.org/10.18653/v1/D17-1314>.

13. Kiselev V., Pronina N. Machine attribution of handwriting in solving source studies problems (based on the correspondence of G.N. Potanin) // Imagology and Comparative Studies. 2025. No. 24.

СВЕДЕНИЯ ОБ АВТОРЕ



ПРОНИНА Наталья Михайловна – магистр кафедры математических методов прогнозирования факультета вычислительной математики и кибернетики Московского государственного университета имени М.В. Ломоносова.

Область научных интересов: нейронные сети, задачи распознавания образов, расшифровка рукописных текстов. Число научных публикаций – 5.

Nataliia Mikhailovna PRONINA – Master's degree in Mathematical Forecasting Methods Faculty of Computational Mathematics and Cybernetics Lomonosov Moscow State University.

Research interests: neural networks, image recognition problems, handwriting recognition. Number of scientific publications: 5.

email: natalka-pronina@mail.ru

ORCID: 0009-0008-1336-4512

Материал поступил в редакцию 9 ноября 2025 года

УДК 004

АВТОМАТИЧЕСКИЕ И ПОЛУАВТОМАТИЧЕСКИЕ МЕТОДЫ ПОСТРОЕНИЯ ГРАФА ЗНАНИЙ ПРЕДМЕТНОЙ ОБЛАСТИ И РАСШИРЕНИЯ ОНТОЛОГИИ

А. П. Халов¹ [0009-0005-4584-8245], О. М. Атаева² [0000-0003-0367-5575]

¹Московский физико-технический институт, г. Долгопрудный,
Московская обл., Россия

^{1, 2}Федеральный исследовательский центр «Информатика и управление»
Российской академии наук, г. Москва, Россия

¹khalov.a@phystech.edu, ²oataeva@frccsc.ru

Аннотация

Рассмотрен цикл построения графа знаний и расширения онтологии для специальной предметной области, описывающей процесс управления потоками данных в службах информационной поддержки. Предложена методика формирования корпуса данных для наполнения онтологии с автоматической псевдоразметкой, включающей специальные категории для фиксации ранее не представленных классов и отношений. Обучена специализированная модель извлечения именованных сущностей на корпусе данных объемом 3 млн токенов с 92 метками. Результаты были использованы для интеграции извлеченных фактов, что увеличило граф знаний до 0.98 млн триплетов, при этом коэффициент расширения графа (отношение общего числа фактов к явным триплетам) увеличился с 2.65 до 3.52 при сохранении логической согласованности. Наборы токенов с одинаковыми метками были преобразованы в устойчивые семантические множества, что позволило полуавтоматически расширить онтологию. В онтологию добавлены 12 новых классов, которые были извлечены из неструктурированных текстовых данных. Показан прикладной пример запросов и дальнейшей аналитики.

Ключевые слова: онтология, DOLCE, граф знаний, NER, BIO-разметка, RDF/OWL, SPARQL.

ВВЕДЕНИЕ

Онтологии и графы знаний стали классическими инструментами для интеграции разнородных данных и поддержки принятия решений в бизнес-процессах управления (IT-сервисы) потоками данных (IT-домен), (IT Service Management, ITSM). Ранее [1] была сформирована объединенная онтология IT-домена путем расширения онтологии верхнего уровня DOLCE [2] доменно-специфичной онтологией ITSMO [3], описывающей концепции библиотеки ITIL. На практике ценность графа знаний определяется скоростью и точностью пополнения фактами из текстов (данных IT-домена, который управляется IT-сервисами) при одновременной способности адаптировать схему (ITSMO-онтологию) под изменяющийся тезаурус домена.

Для динамического построения и пополнения графов знаний, а также для расширения онтологии необходимо наличие специализированных моделей извлечения именованных сущностей и отношений (далее модели NER/RE), обученных на корпусах данных, объекты в которых размечены в соответствии с онтологией. Однако на момент настоящего исследования отсутствуют открытые наборы данных для ITSM, размеченные в соответствии с онтологией из работы [1] (далее онтология). При этом есть доступ к большим объемам текстовых данных в неразмеченном виде. Ручная или полуавтоматическая разметки для обучения модели NER/RE являются трудоемким и дорогостоящим процессом.

В настоящей работе исследован метод полностью автоматической разметки обучающего корпуса данных. Одним из известных способов разметки для задач NER/RE является формат BIO (Beginning-Inside-Outside), который содержит три типа меток: для обозначения первого токена сущности (B); любого последующего токена сущности (I); «фонового» токена (O). Еще одной задачей исследования является создание метода расширения онтологии, т. е. обнаружения в текстах сущностей и отношений, которые не находят соответствия в имеющейся онтологии, однако могут являться источником ценной информации для ее расширения.

В исследовании можно выделить три этапа:

- 1) формирование BIO-корпуса текстов с псевдоразметкой с помощью больших языковых моделей (Large Language Model, LLM), где мы вводим три

множества меток, а также специальные UNK-метки (от англ. UNKnown) в каждом из определенных множеств для разметки слов, не имеющих соответствий в текущей онтологии;

- 2) обучение модели-энкодера из семейства BERT для задачи извлечения именованных сущностей и отношений с проверкой корректности на уровне типов меток;
- 3) автоматическое пополнение графа знаний и полуавтоматическое расширение онтологии через экспертную валидацию кластеров векторных представлений (эмбедингов) слов, которые были размечены как UNK.

Исследуемая предметная область характеризуется высокой динамикой изменения терминов и большим объемом текстовых артефактов (например, заявки клиентов, инциденты, изменения конфигураций). Данные характеристики определяют потребность в методе, сочетающем экономичную и масштабируемую разметку данных с автоматическим обогащением графа и контролируемым расширением онтологии при сохранении логической согласованности.

Статья имеет следующую структуру: в разд. 1 представлены обзор предметной области и связанные работы; в разд. 2 описаны корпус данных и методики псевдоразметки; в разд. 3 представлена архитектура модели; в разд. 4 – конвейер SQL-RDF-NER-RDF; в разд. 5 описаны основные полученные результаты; в разд. 6 – полуавтоматическое расширение онтологии; заключение.

В приложении приведены текст инструкций для LLM (промт) на этапе тестирования моделей-кандидатов (приложение 1), промт для итоговой псевдоразметки целевого корпуса данных (приложение 2), результаты эксперимента с тестированием разных моделей на BIO-бенчмарках (приложение 3), распределение меток после применения метода псевдоразметки (приложение 4), примеры устойчивых семантических множеств (далее синсеты, англ. synset, сокращение от synonym set) и их преобразований в объекты онтологии (приложение 5).

1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ И СВЯЗАННЫЕ РАБОТЫ

Онтология – это формальная спецификация концептуализации [4, 5]; в более прикладной трактовке: согласованное представление предметной области, разделяемое сообществом [6, 7]. В задачах извлечения информации и построения графов знаний онтология выступает схемой данных: сущности и отношения

между ними выражаются через классы и свойства с явными логическими ограничениями (аксиомами), что обеспечивает интерпретируемость, логический вывод и проверку согласованности результата.

В нашей работе использована объединенная онтология: верхний уровень задан известной онтологией DOLCE с абстрактными категориями (объект, процесс, событие), что гарантирует онтологическую стройность; прикладной уровень образует онтология ITSMO, охватывающая сущности и отношения известной методологии ITIL (например, Incident, Service, Change, SLA) и их взаимосвязи. Такое объединение дает единую семантически согласованную структуру, одновременно концептуально строгую и практически релевантную для автоматической аннотации текстов и обогащения графа. Роль онтологии в этом исследовании: (i) это каркас для проекции извлеченных фактов в RDF-триплеты; (ii) это живая схема, которая должна расширяться по мере появления новых терминов и связей в поступающих данных.

1.1. Онтологии и графы знаний для предметной области IT

Работы по онтологическому моделированию охватывают как верхнеуровневые онтологии (SUMO, BFO, DOLCE [6]), так и прикладные доменные онтологии, например ITSMO для процессов ITIL [8]. Слияние ITSMO с DOLCE следует рекомендованной стратегии наследования и позволяет избежать ручного проектирования больших фрагментов схемы [9]. Практики построения и эксплуатации корпоративных графов знаний систематизированы в руководствах [7, 10], где акцент сделан на роли онтологий и логического вывода для контроля качества данных и согласованности.

1.2. Совместное извлечение именованных сущностей и отношений

Автоматическое пополнение графов часто сводится к извлечению из текста сущностей и отношений. Классические каскады последовательного извлечения сущностей и отношений из текстов (NER+RE) постепенно уступают место унифицированным решениям.

Ранним сквозным подходом стала архитектура, предложенная в работе [12]: две параллельные нейросети на основе общего BiLSTM-энкодера используются для одновременного извлечения сущностей и отношений. Такой подход обеспечивает стремление модели к согласованным предсказаниям, поскольку

противоречивые метки (когда токен одновременно помечается и как сущность, и как отношение) приводят к увеличению общей функции потерь модели.

Дальнейшие работы усилили логические ограничения в обучении: семантическая функция потерь, учитывающая априорные символические знания [13], позволила повысить когерентность предсказаний; исследованы взаимодействия задач NER и RE в единой архитектуре [14]; предложены модели для иерархической мультиклассовой классификации, гарантирующие непротиворечивость выходов за счет структуры сети [15]. Параллельно развивались линии структурно согласованных подходов извлечения информации (information extraction, далее IE): формулировка NER как разбора зависимостей [16], а в [17] авторы предложили генеративные универсальные модели IE. Эффективность подобных подходов растет при дообучении (Domain-adaptive pretraining) моделей методом моделирования естественного языка с маскированием части токенов входящей последовательности (Masked Language Modeling – MLM) [18], в этой же работе показано, что снижение функции потерь (и связанной с ним специфичной для MLM метрики perplexity) имеет прямую корреляцию с улучшением производительности модели на прикладных задачах.

1.3. Псевдоразметка и формирование словаря BIO-меток

Другим направлением, близким нашей задаче, является автоматическая разметка данных (далее псевдоразметка) с помощью генеративных моделей трансформеров (GPT, generative pretrained transformer). Авторы модели GPT-3 показали способность модели выполнять произвольные задачи с минимальной настройкой (one-shot/few-shot learning) [19]. В работе [20] продемонстрировано, что открытые LLM (например, LLaMA, Falcon) могут эффективно аннотировать тексты, приближаясь по качеству к модели GPT-4. В то же время отдельные работы указывают на риски применения генеративного искусственного интеллекта (ИИ) для подобных целей [21], в первую очередь это риск переноса в итоговый датасет галлюцинаций и предвзятости (biases). Стоит заметить, что указанные работы сфокусированы на задачах аннотации текстов в «свободной форме», т. е. в них не рассматриваются специализированные подходы к аннотации на уровне последовательности токенов, как этого требует BIO-формат разметки для задачи NER/RE.

В нашей работе мы опираемся на эти идеи и применяем модели GPT для полностью автоматической BIO-разметки специализированного русскоязычного корпуса обращений в службу ИТ-поддержки.

2. ПСЕВДОРАЗМЕТКА BIO

Для эксперимента использовался доменно-специфичный корпус, включающий в себя свыше 660000 текстовых обращений в службу ИТ-поддержки (датасет преимущественно на русском языке, порядка 20% текстов на английском). Доменные понятия сформированы на базе онтологии.

Из онтологии мы получили номенклатуру из 92 меток, куда кроме классов и отношений были включены специальные метки «UNK-х» (отдельно для классов, отношений/свойств и типов), а также вспомогательное отношение «I_A», обозначающее связь «is-a».



Рис. 1. Схема формирования словарей меток классов.

Рассматривается задача преобразования исходных текстов в датасет, который в дальнейшем (см. разд. 4) использован для обучения модели NER/RE. Для каждого текста требуется сгенерировать три параллельные последовательности BIO-меток. Первая последовательность фиксирует класс сущности (в том числе не входящий в онтологию класс), вторая – класс отношения (в том числе не входящее в онтологию отношение), третья, более абстрактная, последовательность указывает тип: токен относится к классу (CLA), отношению (REL), не распознан (UNK) либо не несет семантической нагрузки (O).

Специальные метки семейства UNK содержатся во всех трех типах: UNC-C – для неизвестных классов, UNK-R – для неизвестных отношений, UNK – для обозначения неопределенного типа. Метки UNK нужны для выполнения поставленной

задачи по двум причинам: во-первых, они позволяют модели корректно обработать неизвестную сущность без принудительного отнесения ее к ближайшему известному классу или отношению; во-вторых, они формируют отдельный набор данных. Токены, которые были размечены как UNK-C и UNK-R, после объединения в слова, образуют множество лексических единиц, которые после векторизации (создания эмбеддингов) можно кластеризовать. Кластеры образуют множество синсетов, позволяющих выявить кандидатов на новые классы и отношения/свойства, которые могут расширять онтологию. На рис. 1 показана схема разделения состава онтологии на подмножества классов и отношений. Объединенная онтология указана на рисунке как Seed.

На процесс псевдоразметки наложено ограничение, которое определяет максимально возможный объем разметки, что вызывает необходимость тщательно оптимизировать как объем корпуса, так и длину инструкций (промптов), передаваемых языковой модели.

2.1. Методика выбора модели для псевдоразметки

Для оценки способности языковых моделей к BIO-разметке были выбраны четыре известных корпуса для задач NER, размеченных вручную (далее бенчмарки, англ. benchmark):

- CoNLL 2003 [22] содержит 20 тыс. газетных предложений, размеченных девятью аббревиатурными метками; он считается классическим англоязычным эталоном;
- WikiAnn [23] представлен в двух версиях: англоязычной (en) и русскоязычной (ru), каждая включает примерно 40 тыс. объектов и аббревиатурных меток, что позволяет проверить мультиязычность моделей;
- Корпус WNUT 17 [24] – значительно меньший по объему (около 5.5 тыс. объектов), он отличается тем, что здесь метки являются самостоятельными понятиями: *corporation*, *creative work* и т. п., в отличие от остальных, где метки реализованы как трехбуквенные индексы.

Испытаны 12 LLM: 8 проприетарных (модели от поставщиков OpenAI и Anthropic) и 4 открытых модели (Llama3, Mistral, DeepSeek). Промпт содержит инструкции и примеры разметки [25] (см. приложение 1). Каждой модели было отправлено 2000 заданий на разметку, после чего была определена оптимальная

модель для псевдоразметки целевого корпуса текстов, которая показала максимальные значения метрик.

При тестировании моделей в качестве критерия оценивания использовалась метрика F1 двух типов. Token F1 рассматривает каждый токен отдельно: показатель может быть высоким, если модель верно предсказывает метки большинства токенов в последовательности. Seqeval F1 – более строгий критерий; предсказание считается верным только тогда, когда модель верно определяет и границы, и тип всей сущности. Разница между Token F1 и Seqeval F1 показывает способность модели удерживать целостность последовательности меток на уровне сущности целиком.

2.2. Методика псевдоразметки целевого корпуса текстов

В процессе псевдоразметки на каждой итерации языковая модель одновременно могла использовать все 92 метки. На этапе постобработки метки последовательности разделялись на два непересекающихся множества меток (классы и отношения) и третье производное множество, которое фиксирует только тип токена: класс, отношение, неизвестное (UNK) или токен без семантической нагрузки (O). Это гарантирует, что три BIO-последовательности взаимно исключаются и логически согласованы.

Промпт состоит из четырех частей: инструкция, пример разметки, текст для разметки и словарь меток с кратким описанием семантики каждой метки (см. приложение 2).

2.3. Иллюстрация разметки

Иллюстрация конвейера обработки представлена, как условная мини-онтология (на рис. 2 вверху), сущности которой преобразуются в набор меток, а далее применяются к предложению и раскладываются на три параллельные BIO-последовательности: классы (CLA set), отношения (REL set) и типы токенов (TYPE set), показано на рисунке внизу. Элементы, отсутствующие в мини-онтологии, размечены как UNK-C или UNK-R.

Для демонстрации (на рис. 2) использована примитивная онтология с тремя классами (PER – персона, SFT – программное обеспечение, DEV – устройство) и одним отношением (USD – используется). В предложении также присутствуют неизвестное отношение с меткой UNK-R и неизвестный класс с меткой UNK-C, которых

нет в онтологии, но они явно выражены в тексте. Это позволяет системе помечать кандидатов для последующего расширения онтологии.

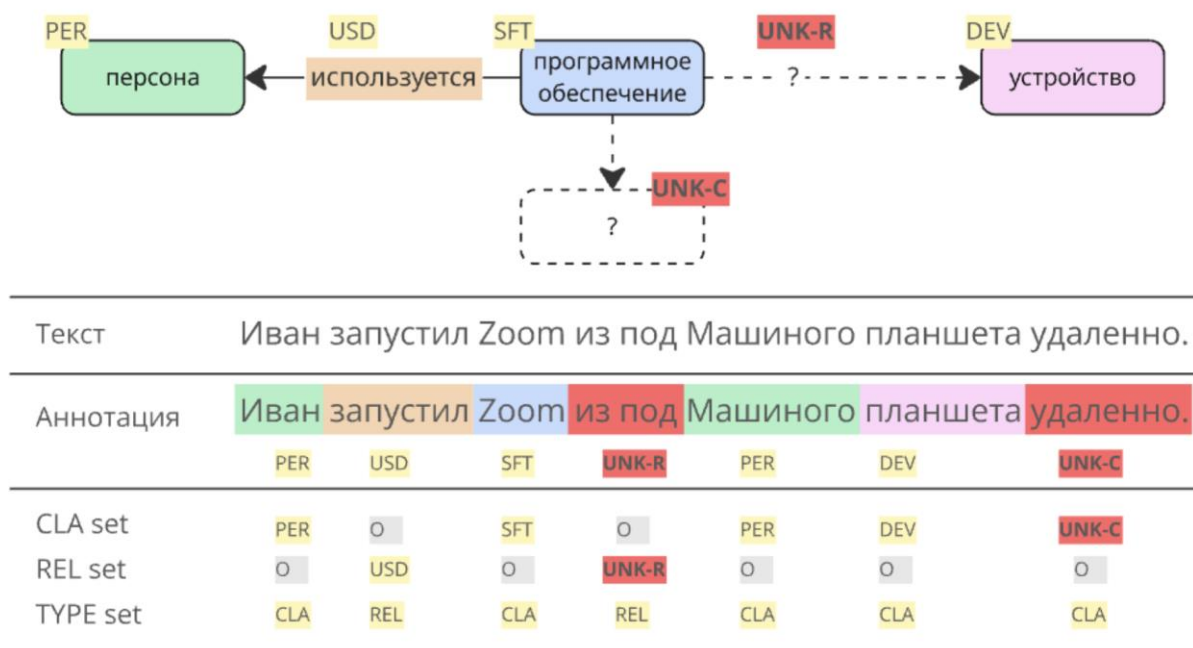


Рис. 2. Иллюстрация разметки на примере мини-онтологии.

Например, слово «удаленно» помечается как UNK-C, так как понятия «способ»/«режим» не имеют подходящего класса в онтологии; а словосочетание «из под» – как UNK-R, поскольку в онтологии нет подходящего отношения (например, им может стать новое отношение «выполнено»).

2.4. Формирование подмножества из корпуса данных для разметки

Для соблюдения ограничений внешних ресурсов необходимо сформировать репрезентативное подмножество объектов из корпуса данных (далее подвыборка) для псевдоразметки. Для каждого текстового объекта мы сгенерировали эмбединг при помощи модели из [1], затем провели кластеризацию и вычислили геометрические медианы кластеров. Предполагалось, что для сохранения семантической репрезентативности подвыборки нужно выбрать объекты, расположенные в пределах заданного радиуса от геометрической медианы каждого кластера в пространстве векторных представлений объектов корпуса данных.

На первом шаге, для уменьшения вычислительной сложности, размерность эмбедингов была снижена с 768 (размер выходного скрытого слоя модели) до 128 с помощью анализа главных компонент (PCA):

$$X' = \text{PCA}(X), \quad X \in \mathbb{R}^{N \times D},$$

где X – исходная матрица эмбедингов N объектов размерности D , а X' – матрица после снижения размерности.

К эмбедингам пониженной размерности затем применялся алгоритм кластеризации HDBSCAN [26]:

$$L = \text{HDBSCAN}(X', m),$$

где $L = \{-1, 0, 1, \dots, K - 1\}$ – вектор кластерных меток (-1 означает шум), а m – гиперпараметр минимального размера кластера по количеству объектов.

Для каждого обнаруженного кластера $C_k \subseteq X'$, где $k = 0, \dots, K - 1$, вычисляли геометрическую медиану по алгоритму Вайцфельда [27]:

$$\mu_k = \underset{\mu}{\operatorname{argmin}} \sum_{x \in C_k} \|x - \mu\|_2, \mu_k \in \mathbb{R}^D,$$

которая выступает как координата «наиболее типичной» точки кластера k .

Для каждого кластера C_k мы подбирали такой радиус r_k , что если оставить все объекты $x_i \in C_k$, удовлетворяющие условию

$$\|x - \mu\|_2 \leq r_k,$$

то общее число выбранных объектов по всем кластерам не превышает лимит 90000 (лимит был задан в соответствии с внешними ограничениями).

Радиусы r_k можно находить либо итеративным двоичным поиском, либо выбрав общий квантиль расстояния от медиан для всех кластеров.

Итоговая подвыборка имеет вид

$$X_{\text{distil}} = \bigcup_{k=0}^{K-1} \{x_i \in C_k : \|x_i - \mu_k\|_2 \leq r_k\}.$$

Таким образом, можно обобщить порядок действий. Сначала сжимались эмбединги исходных текстов (PCA), затем кластеризовались (HDBSCAN) и характеризовались с помощью геометрических медиан кластеров. Объекты для псевдоразметки отбирались по расстоянию до медианы – подход, позволяющий эффективно сформировать подвыборку из исходного корпуса данных при минимальной потере репрезентативности.

3. АРХИТЕКТУРА И ОБУЧЕНИЕ МОДЕЛИ NER/RE

Наличие 92 категорий в словаре меток свидетельствует о необходимости снижения вычислительной сложности и компенсации дисбаланса меток в выборке. Было предложено реализовать параллельный процесс разметки тремя множествами меток: CLA, REL, TYPE. Модель NER/RE представляет собой три нейросети, которые получают на вход выходной вектор каждого токена с последнего скрытого слоя энкодерной модели BERT (показана на рис. 3, далее базовый энкодер). Сети были обучены параллельно на одной и той же входящей последовательности токенов. Рассмотрены два варианта архитектуры: полносвязные слои и слои с механизмом внимания. Для каждого варианта мы протестировали 2, 4 и 8 слоев нейросетей, уменьшая размер скрытого представления вдвое на каждом слое до тех пор, пока он оставался большим или равным числу выходов соответствующего набора меток для каждой сети.

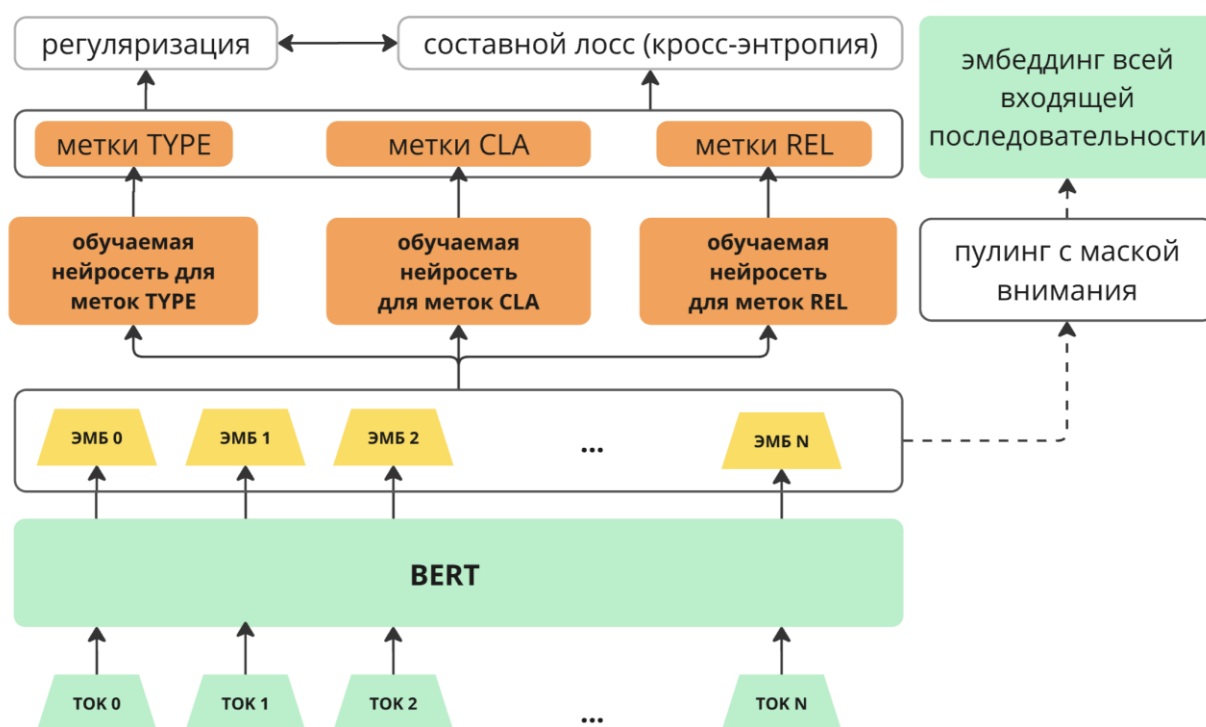


Рис. 3. Архитектура модели NER/RE.

3.1. Функция потерь для задачи NER/RE с регуляризацией

Представление токенов и последовательности реализовано следующим образом. Пусть $X = (x_1, \dots, x_n)$ – матрица входящей последовательности токенов.

Базовый энкодер порождает скрытые состояния $H = f_{\text{enc}}(X) \in \mathbb{R}^{n \times d}$. Каждая дополнительная нейросеть сопоставляет скрытому состоянию вектора выхода внутреннего слоя (логиты токенов) по своему отдельному словарю меток:

$$\hat{y}^{(a)} = f_a(h_i) \in \mathbb{R}^{|C_a|}, \quad a \in \{\text{type}, \text{class}, \text{rel}\}.$$

Дополнительно архитектура модели позволяет получать эмбединг всей последовательности через усреднение состояний всех токенов с учетом маски внимания базового энкодера (пулинг, от англ. mean pooling), за исключением отступов (паддингов, от англ. padding) [28]. Кодирование последовательности целиком потребуется на этапе обработки UNK-наборов, для формирования синсетов расширения онтологии:

$$h_{\text{seq}} = \frac{\sum_{i=1}^n m_i h_i}{\max(\sum_{i=1}^n m_i, \varepsilon)}, \quad \varepsilon = 10^{-9}, \quad m_i \in \{0, 1\}.$$

Этот прием принят в моделях BERT, ориентированных на векторные представления всей входящей последовательности сразу, поскольку дает более стабильные и выразительные эмбединги, чем кодирование одного только CLS-токена (особенно для задач семантического поиска и кластеризации).

Функция потерь L (далее лосс, от англ. loss function) является суммой кросс-энтропийных потерь по всем трем сетям. Дополнительно введена специальная регуляризационная функция логической согласованности предсказаний

$$\mathcal{L} = \sum_a \mathcal{L}_{\text{head}}^{(a)} + \lambda \mathcal{L}_{\text{reg}}, \quad a \in \{\text{type}, \text{class}, \text{rel}\},$$

где для каждой сети функция лосс рассчитана по стандартной формуле

$$\mathcal{L}_{\text{head}}^{(a)} = -\frac{1}{N} \sum_{i=1}^N t_i^{(a)\top} \log \text{softmax}(y_i^{(a)}),$$

$$[\text{softmax}(y_i^{(a)})]_k = \frac{\exp(y_{i,k}^{(a)})}{\sum_{j=1}^{K_a} \exp(y_{i,j}^{(a)})}.$$

Регуляризатор \mathcal{L}_{reg} увеличивает значение \mathcal{L} в случае, если предсказания модели не проходят проверку согласованности, и тем самым штрафует модель за логические ошибки в выходных последовательностях меток.

Пусть $\hat{T}_i = \operatorname{argmax} \operatorname{softmax}(\hat{y}_i^{(a)})$ – предсказанный тип токена i , а $w_i = \max \operatorname{softmax}(\hat{y}_i^{(a)})$ оценивает степень его предсказания («уверенность предсказания»). Введем предсказательные метки для классов и отношений: C_i и R_i соответственно, а также «скобки Айверсона» $[P] \in \{0, 1\}$, равные 1, если предсказание истинно. Обозначим метки a_i для класса и b_i для отношения:

$$a_i = [C_i \neq \mathcal{O}], \quad b_i = [R_i \neq \mathcal{O}],$$

где \mathcal{O} – обозначение О-токена (фоновый токен, которому присваивается класс outside) из BIO-разметки. Тогда индикатор нарушения для токена i равен v_i :

$$v_i = [\hat{T}_i = CLA]([C_i = \mathcal{O}] \vee [R_i \neq \mathcal{O}]) + [\hat{T}_i = REL]([C_i \neq \mathcal{O}] \vee [R_i = \mathcal{O}]) + \\ + [\hat{T}_i = \mathcal{O}]([C_i \neq \mathcal{O}] \vee [R_i \neq \mathcal{O}]) + [\hat{T}_i = UNK](1 - [C_i \neq \mathcal{O}] \oplus [R_i \neq \mathcal{O}]),$$

где \vee – логическое «или», \oplus – исключающее «или» (XOR). Тогда можно записать $a \oplus b = a + b - 2ab$.

После подстановки получим

$$v_i = [\hat{T}_i = CLA](1 - a_i(1 - b_i)) + [\hat{T}_i = REL](1 - b_i(1 - a_i)) + \\ + [\hat{T}_i = \mathcal{O}](a_i + b_i - a_i b_i) + [\hat{T}_i = UNK](1 - a_i - b_i + 2a_i b_i).$$

Поскольку события (факт разметки) взаимно исключают друг друга, для каждого токена активной будет только одна скобка – v_i , которая принимает значение 1 или 0. Итоговый регуляризатор с весами уверенности по TYPE может быть записан как

$$\mathcal{L}_{\text{reg}} = \frac{1}{N} \sum_{i=1}^N w_i v_i.$$

Обозначим множество сетей как $\mathcal{A} = \{\text{type, class, rel}\}$, а кросс-энтропию как $CE(t, p) = -t^T \log p$ (для one-hot вектора t). Тогда лосс примет вид

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \left[\sum_{a \in \mathcal{A}} CE(t_i^{(a)}, \operatorname{softmax}(y_i^{(a)})) + \lambda w_i v_i \right].$$

Такая запись делает логические ограничения частью цели обучения. Были проведены тестовые циклы обучения как с регуляризатором ($\lambda > 0$), так и без него ($\lambda = 0$).

3.2. Оценка модели

Для оценки обученной модели NER/RE в качестве метрики оптимизации мы выбрали точность (далее – метрика *precision*), под которой понимаем долю корректно размеченных токенов среди всех токенов, которым модель присвоила ненулевую (не-«О») метку класса или отношения. В контексте обогащения графа знаний это позволяет минимизировать число ложноположительных фактов в RDF-графе. Каждая ошибочно добавленная сущность или связь нарушает семантическую согласованность и требует ручной очистки, тогда как пропущенные факты могут быть дополнительно извлечены на последующих итерациях конвейера. Это определяет приоритет метрики *precision* при умеренном снижении полноты, т. е. доли корректно размеченных токенов данного класса среди всех токенов данного класса в выборке (далее – метрика *recall*).

4. СОЗДАНИЕ И ОБОГАЩЕНИЕ ГРАФА ЗНАНИЙ

Для включения извлеченных фактов (в форме триплетов субъект – предикат – объект) в граф знаний мы разработали конвейер, преобразующий исходные данные и результаты BIO-разметки в формат RDF. На первом шаге из реляционной базы данных были выбраны основные таблицы: *Tasks* (заявки), *Companies* (компании-клиенты), *Devices* (оборудование). Эти таблицы были экспортированы, и для каждой записи был создан экземпляр соответствующего класса онтологии. Так, например, для строки в таблице *Tasks* генерируется экземпляр класса «*ServiceRequest*» (заявка). Поля заявки (тема, описание проблемы, комментарий решения, трудозатраты, временные метки и т. д.) преобразуются в литералы или связи графа. Приведение к соответствию (далее маппинг, от англ. *mapping*) объектов базы данных классам онтологии обеспечивает начальное наполнение графа знаний структурированной информацией из базы данных, схематично показано на рис. 4. На этом этапе было извлечено подмножество из 10000 заявок, которые далее трансформировались в начальный граф для последующего обогащения моделью NER/RE.

На втором шаге выполнялось извлечение знаний из текста при помощи ранее обученной модели NER/RE (разд. 4). Для каждой заявки извлекались тексто-

вые поля (тема, описание, комментарии) и соединялись в один текстовый фрагмент. Модель NER/RE обрабатывала текстовый фрагмент и возвращала последовательность меток для токенов.

Последовательности токенов длиной n с BIO-метками одного класса $\{B^c, I_1^c \dots I_n^c\}$ объединялись в единую именованную сущность, экземпляр соответствующего класса или отношения онтологии. Каждому такому объекту присваивалось уникальное имя URI (Uniform Resource Identifier, в пространстве имен экземпляров, например <http://itog.it/ExtractedEntity/UUID>).

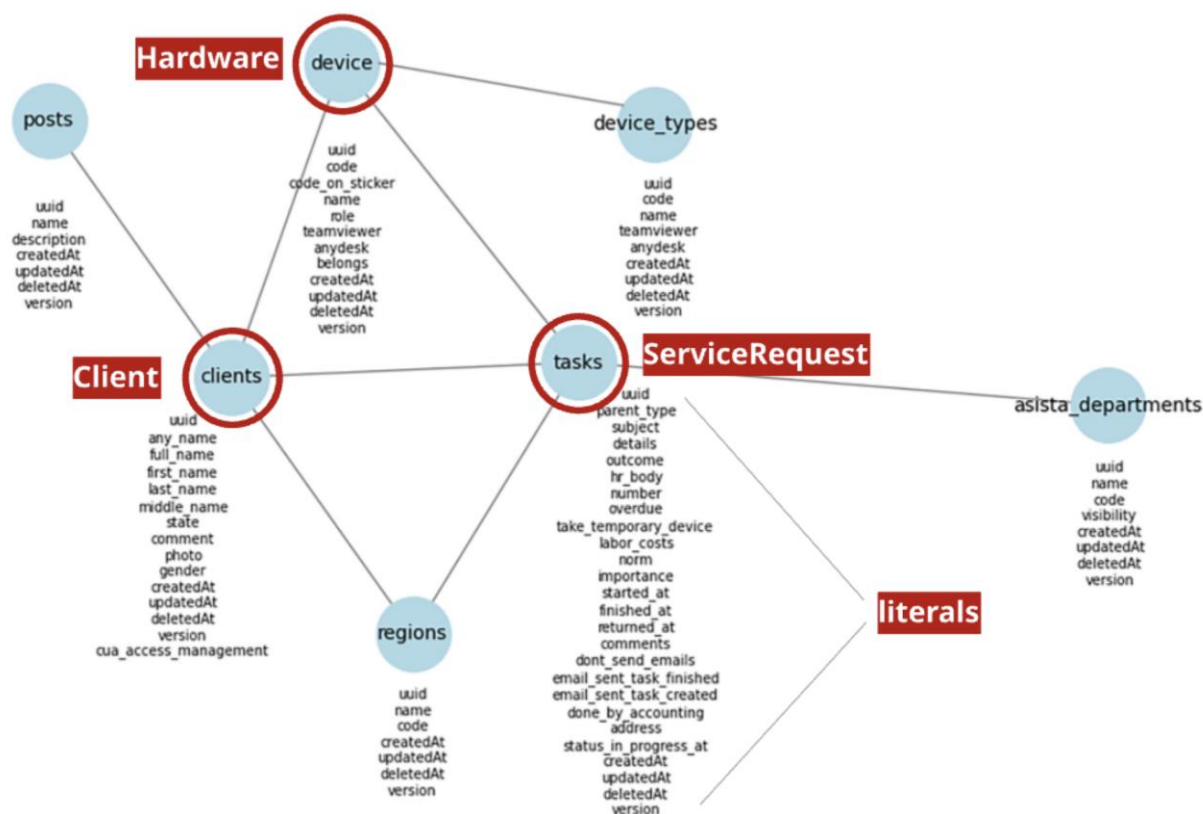


Рис. 4. Схема сопоставления таблиц из базы данных классам онтологии.

Затем на основе предсказанных отношений проверялся шаблон: если была обнаружена последовательность «сущность – отношение – сущность» (например, «сервер обеспечивает приложение») и при этом предсказание соответствовало онтологическому свойству, которое допустимо между классами этих сущностей, то формировалась RDF-тройка, связывающая две извлеченные сущности данным свойством.

5. РЕЗУЛЬТАТЫ

5.1. Тестирование и выбор модели для BIO псевдоразметки

Модели сравнивались на четырех бенчмарках (CoNLL 2003, WikiAnn (en), WikiAnn (ru), WNUT 17) по двум типам метрик F1. На рис. 5 показаны диаграммы распределения метрик, сгруппированных по датасетам. Модели отмечены разными цветами. На верхней диаграмме показаны значения метрики Token F1, на нижней – значения Seqeval F1 для каждой из моделей.

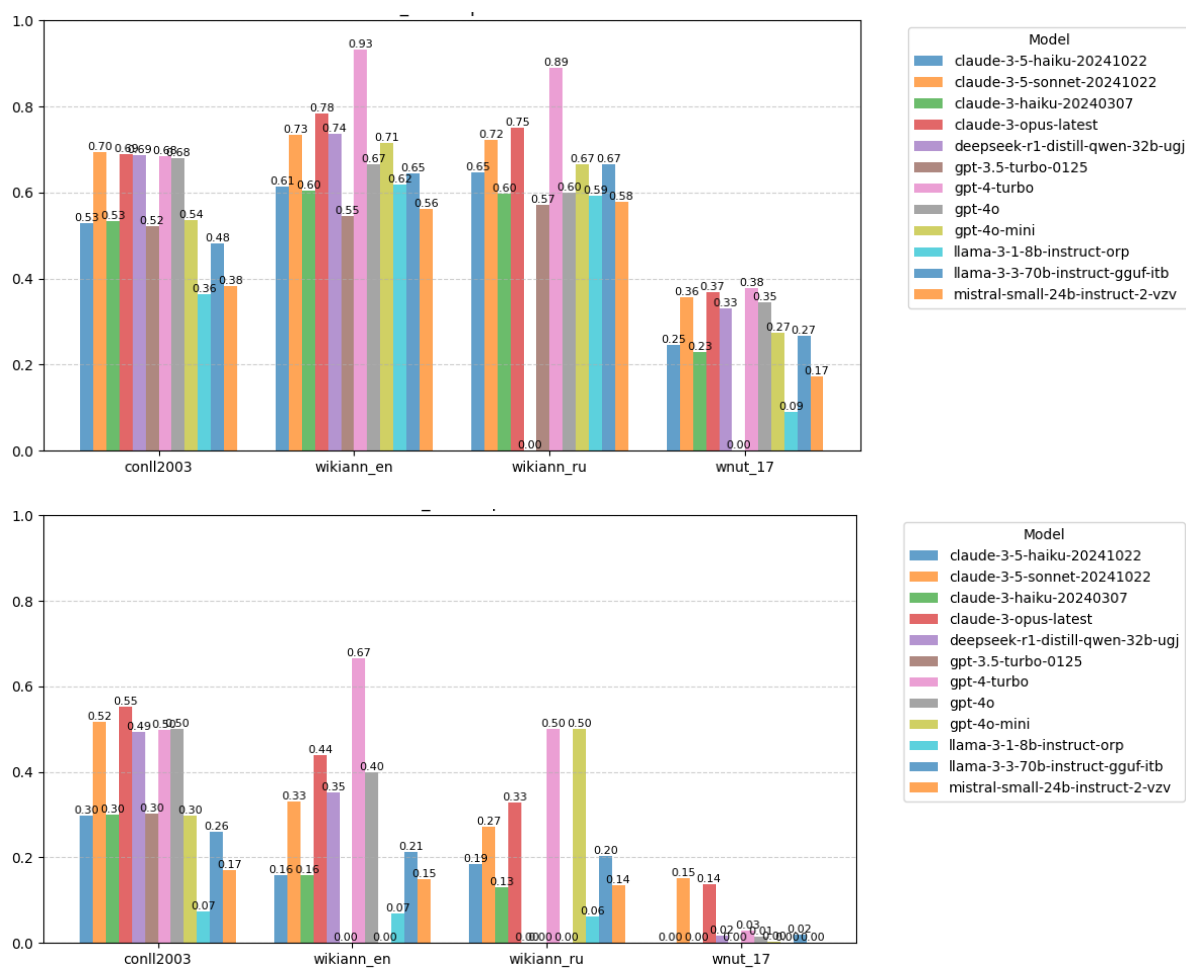


Рис. 5. Сравнительная диаграмма метрик Token F1 и Seqeval F1.

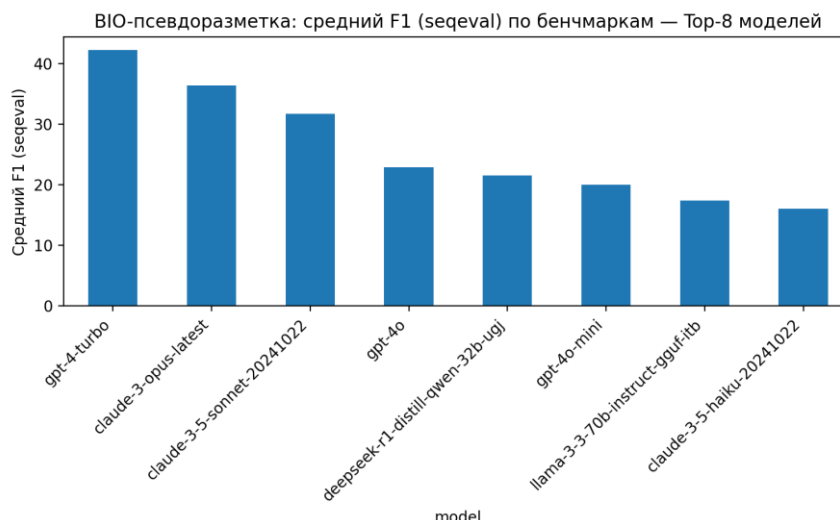


Рис. 6. Сравнение моделей по усредненной Seqeval F1.

Модель GPT-4-turbo на задаче BIO-разметки превосходит конкурентов в большинстве случаев. Интересно, что эта модель на момент проведения экспериментов не являлась самой современной, но показала лучшие результаты. На рис. 6 показана диаграмма метрики Seqeval F1, усредненная по всем испытаниям. Самые худшие результаты получены на датасете WNUT 17, предположительно это может быть вызвано со спецификой меток: модели сложнее проводить разметку, если сами метки представлены большими последовательностями символов или содержат в себе самостоятельный смысл.

Таким образом, можно по среднему Seqeval F1 среди всех датасетов сделать следующие выводы: по усредненной оценке лидируют крупные модели; модель GPT-4-turbo входит в число лучших по среднему и занимает первое место на двух наборах WikiAnn. На английском новостном корпусе CoNLL 2003 первое место у модели Claude-3-opus, а WNUT 17 остается сложным для всех претендентов из списка моделей-кандидатов для BIO-разметки (см. приложение 3).

Результаты эксперимента показали, что: наилучшим практическим выбором является модель GPT-4-turbo; она стабильно работает на мультиязычных бенчмарках, демонстрирует высокую метрику precision, и дает лучшую/сопоставимую метрику Seqeval F1 на мультиязычных корпусах.

5.2. Формирование подмножества выборки и псевдоразметка

Несмотря на то что модель GPT-4-turbo продемонстрировала лучшие результаты на бенчмарках, для финальной разметки корпуса данных была выбрана

модель GPT-4o. Такое решение обусловлено следующими факторами: (1) значительная экономия затрат при сопоставимом качестве на русскоязычных данных; (2) лучшие показатели на мультиязычных корпусах, соответствующих профилю датасета; (3) оптимальное соотношение precision/recall (см. приложение 2).

Приблизительный расчет позволил получить количество объектов, которые можно будет разметить в рамках внешних ограничений: 90000 объектов к разметке или порядка 13% от исходного корпуса данных. Была проведена фильтрация по методу, указанному в п. 2.4. На рис. 7 показана двумерная проекция кластеризованных эмбедингов объектов исходного корпуса данных (слева) и отфильтрованного подмножества (справа).

После применения модели GPT-4o были извлечены три подмножества токенов, размеченных как типы, классы и отношения. В результате получился размеченный корпус данных объемом 81505 объектов, содержащий 3037348 токенов, из них 773324 размеченных (размеченным считается слово с любой меткой кроме “O”) (см. приложение 4).

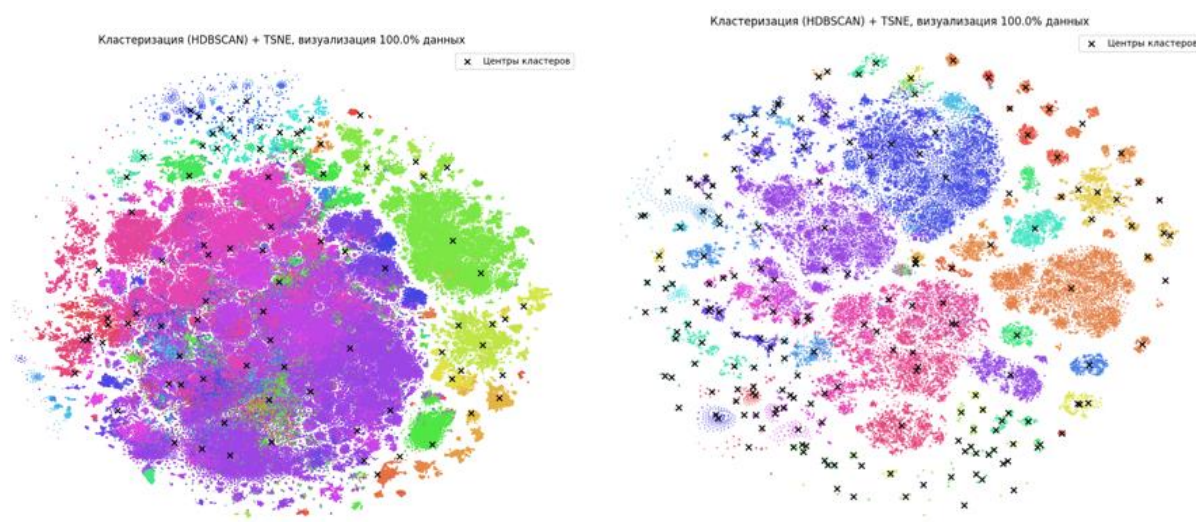


Рис. 7. Двухмерная проекция эмбедингов до и после фильтрации.

5.3. Дообучение базовой модели

В качестве базовой выбрана модель XLM-RoBERTa-large (24 слоя трансформера, ~550 млн параметров), предобученная на корпусе CommonCrawl для 100 языков [29]. Чтобы учесть специфику и увеличить качество модели на доменных данных, мы провели дополнительное доменное дообучение модели. Обучающий корпус для задачи MLM (общим объемом > 650 тыс. объектов) был составлен из

исходного корпуса данных, состоящего из текстов внутренней базы заявок техподдержки и документации.

При обучении в последовательности маскировались 15% токенов, что является рекомендуемым гиперпараметром для задачи моделирования естественного языка методом маскирования. Обучение проводилось в течение 33 эпох, из табл. 1 видно, что лосс и метрика perplexity значительно уменьшились после обучения, что указывает на рост способности модели производить доменно-специфичные векторные репрезентации текстов на естественном языке. Дообучение проводилось на одном GPU ускорителе NVIDIA A100 в течение 250 часов.

Табл. 1. Сравнение метрик до и после дообучения модели XLM-RoBERTa-large на валидационной (вал.) и тестовой (тест.) выборках.

	Loss (вал.)	Perplexity (вал.)	Loss (тест.)	Perplexity (тест.)
Исходная модель	1.52	4.55	1.53	4.60
Дообученная модель	0.52	1.68	0.52	1.68

5.4. Обучение модели NER/RE

На этапе подбора гиперпараметров было установлено, что предлагаемая в разд. 4 регуляризация негативно влияет на метрики (см. приложение 5). Архитектуры с механизмом внимания показали наилучшие результаты. В табл. 2 показаны сравнительные метрики по лучшим тестовым циклам обучения, в колонке «head» отмечены два типа сетей – с механизмом внимания (att, attention) и без него (ff, ffed forward), в колонке «reg» указан коэффициент регуляризации, а в «layers» – количество слоев нейросети. Полная таблица по всем тестовым циклам обучения модели NER/RE представлена в приложении 5.

Табл. 2. Сравнительные метрики моделей.

			f1			accuracy			precision			recall		
head	reg	layers	class	rel	avg	class	rel	avg	class	rel	avg	class	rel	avg
ff	0	2	0.56	0.72	0.64	0.45	0.58	0.51	0.85	0.95	0.90	0.45	0.58	0.51
att	0	4	0.57	0.76	0.67	0.46	0.61	0.54	0.85	0.96	0.91	0.46	0.61	0.54

5.5. Оценка модели

Оценка в табл. 2 выполнена на всей входящей последовательности токенов с учетом фонового класса «О», здесь также использована метрика Token F1. Такой режим демонстрирует высокую метрику precision (≈ 0.91) при умеренных значениях recall (≈ 0.54), что соответствует выбранной стратегии: модель не стремится предсказать все потенциально информативные токены, однако при назначении метки CLASS или REL демонстрирует высокую надежность.

Табл. 3. Список наиболее сложных лейблов и процент ошибок в протоколе оценки «без О»

Labels hardest by recall			Labels hardest by precision		
Label	Type	Confusion (% err)	Label	Type	Confusion (% err)
PES	class	PCS (26.9%)	CMO	class	SDY (27.8%)
RRE	class	STM (22.0%)	TLE	class	AET (25.0%)
RNT	class	DDY (13.6%)	SST	class	SDY (40.4%)
QIE	class	CCY (38.4%)	CPO	class	SDY (68.6%)
CAM	class	HWE (19.5%)	CON	class	CUN (17.3%)
UNK-R	relation	O (14.9%)	HRS	relation	HER (42.1%)
HSR	relation	HRS (60.9%)	HCB	relation	HER (25.0%)
HRS	relation	HER (42.1%)	HIE	relation	HLY (33.3%)
RIS	relation	UES (22.6%)	HAT	relation	HIS (16.7%)
HES	relation	HRS (20.9%)	UES	relation	HIE (15.0%)
HIS	relation	HES (21.3%)	HME	relation	HES (33.3%)
DEF	relation	O (37.5%)	HVE	relation	HRS (40.0%)

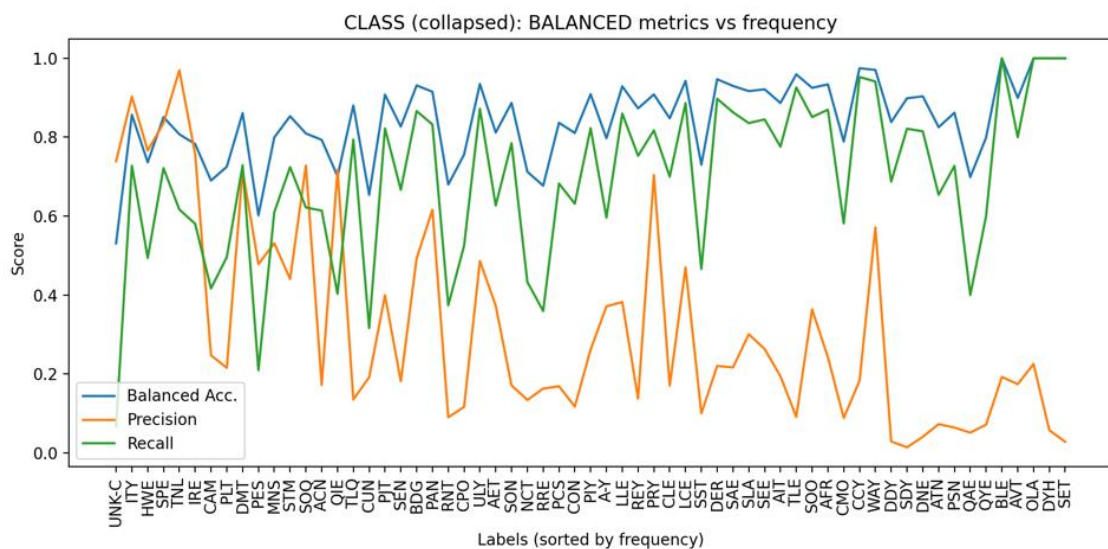


Рис. 8. График изменения метрик в зависимости от частоты метки.

После завершения обучения мы провели оценку модели в протоколе «без О», т. е. с исключением вклада фоновых токенов. В этом режиме внутриклассовая метрика точности (accuracy) и F1 остаются высокими, тогда как метрика precision систематически падает по мере уменьшения поддержки метки – ожидаемый эффект «длинного хвоста». На рис. 8 показано влияние частоты меток в датасете на метрики. В табл. 3 приведены наиболее сложные метки по метрикам recall и precision с указанием топ-ошибок. Причинами ошибок можно назвать семантическую близость объектов и токен-уровневую природу BIO аннотации.

5.6. NER-обогащение графа знаний

Представлены суммарные количественные и качественные эффекты применения предложенного конвейера к корпоративной базе ITSM. Эти результаты демонстрируют не только рост объема знаний, но и то, что новые сущности и связи корректно вписываются в структуру графа знаний и остаются логически согласованными с ограничениями онтологии.

Из табл. 4 видно, что уже на первом этапе обработки каждый объект преобразуется около 12 триплетов, каждый триплет можно интерпретировать как факт, полученный из данных.

Табл. 4. Сравнение базы данных и графов знаний.

Данные	Источник и объем	Узлы	Явные триплеты	Выведенные	Всего фактов	Expansion ratio ¹
SQL	10 000 строк tasks (+ ключи внутри строк)	≈ 10000	–	–	–	–
Базовый RDF	прямой маппинг SQL → RDF	13899	115661	191262	306923	2.65
NER- RDF	базовый граф + автоматически извлеченные факты	66194	279546	705330	984876	3.52

После применения модели NER извлекается еще более 50 тыс. сущностей и образуется более 160 тыс. связей, автоматический вывод новых фактов приумножает их в 2.5 раза. В результате общий объем знаний, извлеченный из 10 тыс. заявок в службу поддержки, приближается к миллиону триплетов, а коэффициент

¹ Expansion ratio рассчитывается как пропорция общего количества фактов в графе к явным триплетам (которые явно были импортированы в граф), является важным показателем вывода новых фактов автоматически.

расширения растет с 2.65 до 3.52. Доля выводимых фактов достигла приблизительно 71.6% от всего графа, что свидетельствует об интенсивном расширении – автоматически выведенные факты составляют большую часть графа. При этом целостность структуры графа сохранилась: как и прежде, все узлы связаны в единую компоненту (благодаря узлам-классам из онтологии).

Таким образом, каждая исходная строка базы данных трансформируется в сотни взаимно связанных фактов, что существенно повышает выразительность данных и открывает возможность сложных запросов на языке SPARQL, недоступных ни на уровне языка SQL, ни в графе знаний в первоначальном состоянии.

Из табл. 5 видно, что увеличение узлов и ребер сопровождается ожидаемым «разрежением» сети, но появление ненулевой кластеризации (0.08) показывает, что новые сущности формируют связанные смысловые кластеры. Это означает, что новые узлы образовали локальные группы. Например, упоминания названий программного обеспечения могли связаться через общий класс *SoftwarePackage* или через одно устройство, класс *Hardware*, к которому они относятся.

Табл. 5. Топологические изменения графа после NER обогащения.

Метрика	До NER	После NER	Δ
Количество узлов	13899	66194	+ 376%
Количество ребер	40094	151670	+ 278%
Средняя степень $\langle k \rangle$	5.77	4.58	-21%
Плотность	$4.1 \cdot 10^{-4}$	$6.9 \cdot 10^{-5}$	$\downarrow \times 6$
Кластеризация \bar{C}	0.00	0.08	+0.08

Получившаяся семантическая сеть обладает свойствами безмасштабной структуры: распределение степеней узлов монотонно убывает, в распределении существуют узлы с относительно высокой степенью – это узлы онтологии и наиболее частые сущности, такие как, например, типовые услуги; большинство же узлов имеет малую (в десятки раз меньше) степень. Такой характер свидетельствует о корректности интеграции новых данных: они не превратили граф в хаотичную структуру, где все связано со всем, а вписались в уже существующую семантическую конструкцию, образуя смысловые кластеры вокруг известных концепций.

На рис. 9 сравнивается одна и та же сущность, соответствующая исходному объекту данных (заявка в службу поддержки) до и после NER-обогащения. Как видно, в исходном состоянии с центральным узлом связан ограниченный набор объектов, а после обогащения появилось девять новых связей, которых не было в базе данных. Они описывают временные интервалы, режим, организацию, ответственных лиц и т. д., вся эта значимая информация содержалась имплицитно (не явно) в тексте заявки на естественном языке.

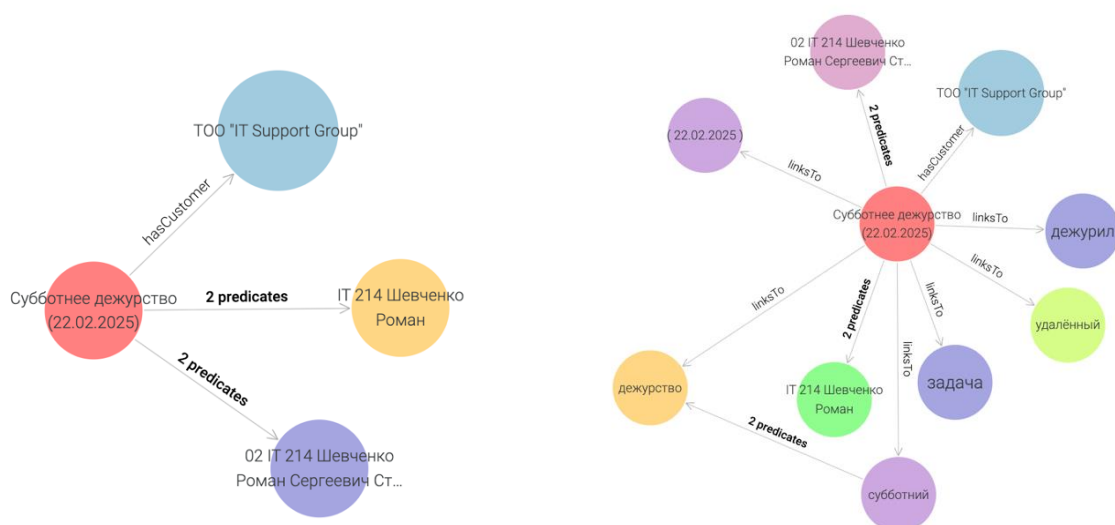


Рис. 9. Фрагмент графа знаний до и после NER-обогащения.

Модель автоматически обнаружила экземпляры классов, после чего между ними были установлены взаимосвязи в соответствии с аксиомами, которые в явном виде содержатся в онтологии. Кластерный и семантический анализ сущностей с метками категорий UNK-C и UNK-R далее позволит выявить новые классы и отношения для расширения онтологии.

5.7. Обработка UNK-сущностей и расширение онтологии

Все сущности с метками UNK-C (кандидаты классов) и UNK-R (кандидаты отношений) были кластеризованы для синсетов, пригодных для экспертной валидации и включения в онтологию.

Для каждой UNK-сущности были построены эмбединги способом, указанным в разд. 4.

Начальный набор содержал 244679 элементов. На первом этапе мы устранили шумовые и несловесные единицы и провели фильтрацию по косинусной близости, задав порог как 10-й перцентиль распределения расстояний внутри пар отдельно для классов и для отношений. После этой процедуры осталось 50630 кандидатов классов и 141317 кандидатов отношений. Дополнительная проведенная очистка по длине токена (не менее трех символов) сократила выборку до 44631 и 112695 объектов соответственно. Далее эмбединги были понижены до размерности 128 в помощью метода главных компонент (PCA), после чего была выполнена кластеризация методом k -средних (mini-batch k -means) с гиперпараметром $k=12$ для каждого набора объектов. Для каждого кластера была вычислена геометрическая медиана по алгоритму Вайцфельда; вокруг этой медианы сформировалась компактная окрестность фиксированного размера (не более 50 ближайших), исходные объекты которой сформировали синсет для экспертного анализа. На рис. 10 показан эффект фильтрации по геометрической медиане в кластерах объектов, размеченных как UNK-CLA (неизвестные представители классов, извлеченные из входящих текстов), представлена двухмерная проекция до и после фильтрации. В результате мы получили плотные и однородные локальные семантические множества (приложение 6).

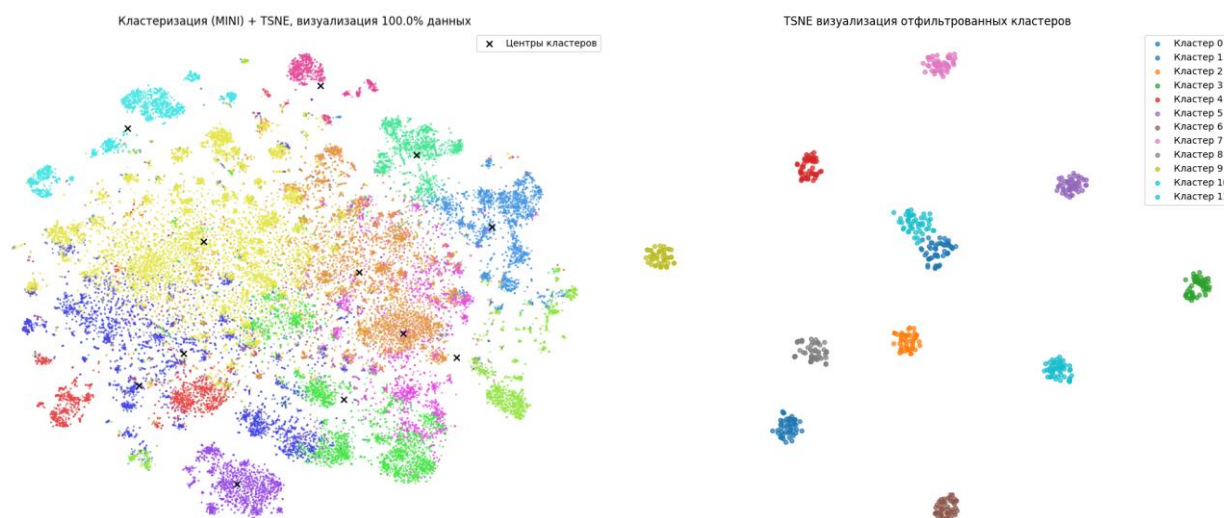


Рис. 10. Двухмерная проекция эмбедингов UNK до и после фильтрации.

Качественный анализ синсетов, проведенный экспертом, показал семантическую когерентность групп и позволил предложить онтологические спецификации. Так, синсет, в котором преобладают такие глагольные формы, как «выдал», «поменяли», «сохранил», «протестил», «восстановили», был проинтерпретирован как класс процессов модификации: *rdfs:label* – *ModificationProcess*; *rdfs:comment* – «тип процесса, отвечающий за изменение конфигурационных единиц, настроек и параметров». Такие свойства, как «ответственность» и «роль», в этом случае наследуются из онтологии и необходимы для дополнения RDF-троек.

6. ПРАКТИЧЕСКАЯ ЗНАЧИМОСТЬ И СЦЕНАРИЙ ИСПОЛЬЗОВАНИЯ

Автоматизация обогащения графа знаний существенно снижает трудоемкость поддержки актуальности знаний в ITSM. В отличие от ручного дополнения онтологии экспертами, предложенный метод может быть реализован как процесс в фоновом режиме, с минимальным вмешательством. Интеграция результатов NER/RE с онтологическими ограничениями обеспечивает высокую точность извлечения знаний из текстов: модель реализует консервативную стратегию извлечения и добавляет в граф только те факты, в которых уверена и которые соответствуют схеме. Благодаря этому свойству достигается семантическая согласованность данных – система не нарушает правила и ограничения, определенные онтологией. Показательно, что даже после почти пятикратного увеличения граф сохраняет строгую структуру и связность.

Автоматизированный граф удовлетворяет специфические запросы, которые недостижимы в реляционной модели данных. На рис. 11 показан интерфейс GraphDB Chat: LLM, подключенная к обогащенному графу, отвечает на вопрос «что у компании „XXXXXX“ ломается чаще всего и что можно им предложить?». Далее по фактам графа выводятся наиболее проблемные устройства клиента и генерируется предложение, релевантное поведению клиента, которое хранится как связанный набор фактов в графе знаний.

Этапы формирования ответа:

1. SPARQL-подзапрос извлекает агрегированную статистику поломок (COUNT по сущностям класса Hardware, связанных с ServiceRequest и ClientCompany).

2. Второй запрос ищет оборудование того же типа, отсутствующее у клиента (FILTER NOT EXISTS { ?new hw itsmo:hasCustomer :XXX }).
3. LLM интерпретирует результаты, ранжируя их по частоте.

Компания [REDACTED] чаще всего сталкивается с проблемами в следующем оборудовании:

1. Сервер HP DL360 G9
2. Принтер Canon iR2520
3. Маршрутизатор Cisco 1841

Чтобы предложить оборудование, которое мы можем продать компании [REDACTED] можно рассмотреть следующее оборудование, которое еще не связано с ними:

1. Dell PowerEdge R740
2. Xerox WorkCentre 6515
3. Cisco ISR 4331

Это оборудование может быть предложено в качестве замены или дополнения к текущему, чтобы улучшить качество их ИТ-инфраструктуры.

Called SPARQL: Direct query

```
SELECT ?hardwareName (COUNT(?serviceRequest) AS ?requestCount) WHERE {
  ?serviceRequest a kz:ServiceRequest;
                  kz:hasCustomer ?client;
                  kz:hasDevice ?hardware.
  ?client rdfs:Label "[REDACTED]".
  ?hardware rdfs:Label ?hardwareName.
} GROUP BY ?hardwareName ORDER BY DESC(?requestCount) LIMIT 5
```

Called SPARQL: Direct query

```
SELECT ?hardwareName WHERE {
  ?hardware a itsmo:Hardware;
            rdfs:Label ?hardwareName.
  FILTER NOT EXISTS {
    ?serviceRequest a kz:ServiceRequest;
                    kz:hasCustomer ?client;
                    kz:hasDevice ?hardware.
    ?client rdfs:Label "[REDACTED]".
  }
}
```

Рис. 11. Интерфейс обработки запросов к графу знаний на естественном языке.

В табличной SQL-схеме такой синтез осложнен: факты о состоянии оборудования хранятся строками разных таблиц, а понятие отсутствующего, но совместимого оборудования не представлено явно. Графовая модель вместе со специализированной языковой моделью позволяют задать это условие SPARQL-конструкцией. Таким образом, продемонстрировано, что наш конвейер повышает экспрессивность запросов и открывает дорогу к рекомендательным сценариям без ручной подготовки витрин данных.

ЗАКЛЮЧЕНИЕ

Разработан и экспериментально валидирован воспроизводимый цикл «граф – модель – граф» для автоматического обогащения корпоративного графа знаний и полуавтоматического расширения онтологии в домене ITSM. Ключевыми компонентами предложенного подхода являются:

- единая онтологическая схема, объединяющая онтологию DOLCE с доменной онтологией ITSMO;
- значительный BIO-корпус (3 млн токенов, 92 метки);
- оригинальная архитектура модели NER/RE;
- механизм UNK-меток.

Проведено расширение онтологии двенадцатью новыми классами и двенадцатью новыми отношениями. Включение осуществлялось с проверкой согласованности по иерархиям и ограничениям онтологии.

Предложенный цикл – от разметки до онтологически оформленных конструкторов – опирается на приоритет точности, что отмечено в разд. 4: мы сознательно выбираем консервативную модель с высокой метрикой *precision*, поскольку для полуавтоматической интеграции важнее надежность единичного решения, чем исчерпывающий охват. На практике такая стратегия уменьшает долю ложноположительных кандидатов в наборах UNK, снижает нагрузку на эксперта и ускоряет прохождение этапов «UNK – синсет – спецификация – включение».

В совокупности это демонстрирует, что UNK-метки, усиленные контекстными эмбедингами и компактной кластерной выборкой, служат эффективным механизмом выявления ранее отсутствующих понятий и свойств. Метод обеспечивает воспроизводимое, управляемое и реактивное расширение онтологии полуавтоматическим способом.

Эффективность подхода подтверждена количественными метриками: увеличение графа в 4.76 раза (с 13899 до 66194 узлов), рост коэффициента расширения с 2.65 до 3.52 при сохранении логической согласованности. UNK-метки для полуавтоматического определения новых классов и отношений позволяют значительно сократить время необходимо для актуализации онтологий.

Представлен набор методов создания синтетических наборов данных с использованием словаря онтологии как пространства меток, показаны возможности явной и неявной логических регуляризаций в процессе обучения модели-энкодера, проведена апробация результатов исследования в рабочих сценариях.

СПИСОК ЛИТЕРАТУРЫ

1. *Khalov A., Ataeva O. Automating Ontology Mapping in IT Service Management: A DOLCE and ITSMO Integration // Data Science Journal. 2025. Vol. 24. P. 23. <https://doi.org/10.5334/dsj-2025-023>*
2. *Borgo S. et al. DOLCE: A descriptive ontology for linguistic and cognitive engineering // Applied Ontology. 2023. Vol. 17, No. 1. P. 45–69.*
3. *IT Service Management Ontology (ITSMO). Canonical resolver; catalog entry in LOV "IT Service Management Ontology (itsmo)". <https://w3id.org/itsmo>; ontology.it; lov.linkeddata.es (Accessed: 08 August 2025).*

4. *Gruber T.R.* A translation approach to portable ontology specifications // Knowledge Acquisition. 1993. Vol. 5, No. 2. P. 199–220.
<https://doi.org/10.1006/knac.1993.1008>
5. *Gruber T.R.* Toward principles for the design of ontologies used for knowledge sharing // International Journal of Human-Computer Studies. 1995. Vol. 43, No. 5–6. P. 907–928. <https://doi.org/10.1006/ijhc.1995.1081>
6. *Smith B.* Ontology (Science) // Formal Ontology in Information Systems, IOS Press, 2008. P. 21–35. <https://doi.org/10.1038/npre.2008.2027.2>
7. *Studer R., Benjamins V. R., Fensel D.* Knowledge Engineering: Principles and Methods // Data & Knowledge Engineering. 1998. Vol. 25, No. 1–2. P. 161–197. [https://doi.org/10.1016/S0169-023X\(97\)00056-6](https://doi.org/10.1016/S0169-023X(97)00056-6)
8. *El Yamami A. et al.* An ontological representation of ITIL framework service level management process // Smart Data and Computational Intelligence. Lecture Notes in Networks and Systems. 2019. Vol. 66. P. 88–94.
https://doi.org/10.1007/978-3-030-11914-0_9
9. *Barrasa J., Webber J.* Building Knowledge Graphs: A Practitioner's Guide. O'Reilly Media, 2023. 250 p.
10. *Hogan A., Blomqvist E., Cochez M. et al.* Knowledge Graphs. Morgan & Claypool Publishers, 2021. 257 p.
11. *Valiente M.-C., Vicente-Chicote C., Rodriguez D.* An Ontology-Based and Model-Driven Approach for Designing IT Service Management Systems // Int. J. of Service Science, Management, Engineering, and Technology. 2011. Vol. 2 (2). P. 65–81.
12. *Miwa M., Bansal M.* End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, 2016. P. 1105–1116.
<https://doi.org/10.18653/v1/P16-1105>
13. *Xu J., Zhang Z., Friedman T., Liang Y., Van den Broeck G.* A Semantic Loss Function for Deep Learning with Symbolic Knowledge // Proceedings of the 35th International Conference on Machine Learning (ICML). PMLR, 2018. Vol. 80. P. 5502–5511. URL: <https://proceedings.mlr.press/v80/xu18h.html>

14. *Sun K., Zhang R., Mensah S., Mao Y., Liu X.* Learning Implicit and Explicit Multi-task Interactions for Information Extraction // *ACM Transactions on Information Systems*. 2023. Vol. 41, No. 2. P. 1–29. <https://doi.org/10.1145/3533020>

15. *Giunchiglia E., Lukasiewicz T.* Coherent Hierarchical Multi-label Classification Networks // *Advances in Neural Information Processing Systems* 33 (NeurIPS 2020). 2020. URL: <https://proceedings.neurips.cc/paper/2020/file/6dd4e10e3296fa63738371ec0d5df818-Paper.pdf>

16. *Yu J., Bohnet B., Poesio M.* Named Entity Recognition as Dependency Parsing // *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2020. P. 6470–6476. <https://doi.org/10.18653/v1/2020.acl-main.577>

17. *Lu Y., Liu Q., Dai D., Xiao X., Lin H., Han X., Sun L., Wu H.* Unified Structure Generation for Universal Information Extraction // *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022. P. 5755–5772. <https://doi.org/10.18653/v1/2022.acl-long.395>

18. *Gururangan S., Marasović A., Swayamdipta S., Lo K., Beltagy I., Downey D., Smith N.A.* Don't Stop Pretraining: Adapt Language Models to Domains and Tasks // *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2020. P. 8342–8360. <https://doi.org/10.18653/v1/2020.acl-main.740>

19. *Brown T. B. et al.* Language Models are Few-Shot Learners // *Advances in Neural Information Processing Systems*. 2020. Vol. 33. URL: <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>

20. *Alizadeh M. et al.* Open-source LLMs for text annotation: a practical guide for model setting and fine-tuning // *Journal of Computational Social Science*. 2025. Vol. 8. P. 17. <https://doi.org/10.1007/s42001-024-00345-9>

21. *Eiras F. et al.* Position: Near to Mid-term Risks and Opportunities of Open-Source Generative AI // *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*. *Proceedings of Machine Learning Research*. 2024. Vol. 235. P. 12348–12370. URL: <https://proceedings.mlr.press/v235/eiras24b.html>

22. *Tjong Kim Sang E. F., De Meulder F.* Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition // *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. Edmonton, Canada: Association for Computational Linguistics, 2003. P. 142–147.

<https://doi.org/10.3115/1119176.1119195>

23. *Pan X., Zhang B., May J., Nothman J., Knight K., Ji H.* Cross-lingual Name Tagging and Linking for 282 Languages // Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics, 2017. P. 1946–1958.

<https://doi.org/10.18653/v1/P17-1178>

24. *Derczynski L., Nichols E., van Erp M., Limsopatham N.* Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition // Proceedings of the 3rd Workshop on Noisy User-generated Text (W-NUT 2017). Copenhagen, Denmark: Association for Computational Linguistics, 2017. P. 140–147.

<https://doi.org/10.18653/v1/W17-4418>

25. *Brown T.B. et al.* Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems 33 (NeurIPS 2020). 2020. URL: <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html>.

26. *Campello R.J.G.B., Moulavi D., Zimek A., Sander J.* Hierarchical density estimates for data clustering, visualization, and outlier detection // ACM Transactions on Knowledge Discovery from Data. 2015. Vol. 10, No. 1. P. 1–51.

<https://doi.org/10.1145/2733381>

27. *Vardi Y., Zhang C.-H.* A modified Weiszfeld algorithm for the Fermat–Weber location problem // Mathematical Programming. 2001. Vol. 90. P. 559–566.

<https://doi.org/10.1007/PL00011435>

28. *Reimers N., Gurevych I.* Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics, 2019. P. 3982–3992. <https://doi.org/10.18653/v1/D19-1410>

29. *Hugging Face.* XLM-RoBERTa (large): specs (24 layers, ~550M params). 2020–2024.

URL: https://huggingface.co/transformers/v3.4.0/pretrained_models.html

30. *Côté M.-A. et al.* TextWorld: A Learning Environment for Text-Based Games // Computer Games (CGW@IJCAI 2018). 2019. Vol. 1017 (CCIS). P. 41–75.

https://doi.org/10.1007/978-3-030-24337-1_3

31. *Russell S., Norvig P.* Artificial Intelligence: A Modern Approach. 4th ed. Pearson, 2020. Chapter 11: Automated planning.

32. *Schmidhuber J.* Gödel Machines: Self-Referential Universal Problem Solvers Making Provably Optimal Self-Improvements // Artificial General Intelligence. 2007. P. 199–226. https://doi.org/10.1007/978-3-540-68677-4_7

33. *Yin X. et al.* Gödel Agent: A Self-Referential Agent Framework for Recursively Self-Improvement // Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2025). 2025. P. 27890–27913. <https://aclanthology.org/2025.acl-long.1354/>

34. *Атаева О.М., Серебряков В.А.* Онтология цифровой семантической библиотеки LibMeta // Информатика и ее применения. 2018. Т. 12, № 1. С. 2–10.

Приложение 1. Промпт, используемый для тестирования способностей моделей к BIO-разметки. В `sorted_labels` и `text` подаются все метки классов из словаря классов и текст для разметки соответственно.

```
Below is a sentence. You have to annotate each token using BIO format for named entity recognition.
The possible entity classes are: {sorted_labels}
Use the 'O' tag if the token does not belong to any entity.
Sentence: {text}
Return the annotations as valid JSON with a single key 'ner', whose value is a list of the same length as the token list.
Each element in that list should be a BIO tag (e.g., 'B-PER', 'I-PER', 'O').
For example: ["B-PER", "O", "B-LOC", ...]
```

Приложение 2. Промпт, используемый LLM псевдоразметки. Фрагмент-промпта для многоклассовой NER-разметки состоит из двух основных частей: часть с примерами, приведенными в требуемом JSON формате, примеры содержат все типы меток, вторая часть – это сам промпт, который содержит в себе рабочую нагрузку в виде текста, подлежащего разметке и список меток с описанием семантики каждой метки (в промпте не указаны аннотации каждого класса).

```
few_shot_examples = (
    Example 1:
    Input: Не работает Windows, не удается подключиться к базе
    Output (JSON):
    {
        "ner": {
            "не": "O",
            "работает": "O",
            "Windows": "B-SPE",
            ", ": "O",
            "не": "O",
            "удается": "O",
            "подключиться": "B-RIS",
            "к": "O",
            "базе": "B-UNK-C"          // unknown class (e.g. data-base)
        }
    }

    system_prompt = (
        "You're an expert NER and Relation Extraction model specialized in the IT domain and ITIL. Annotate each token using BIO tagging for both Named Entity Recognition (NER) and relation extraction."
```

Приложение 3. Результаты эксперимента с тестированием разных моделей на BIO-бенчмарках.

benchmark	model	F1		precision		recall	
		seq.	bin	seq.	bin	seq.	bin
conll2003	claude-3-5-haiku-20241022	29.9%	52.8%	27.2%	52.2%	33.1%	53.4%
	claude-3-5-sonnet-20241022	51.6%	69.5%	47.4%	66.7%	56.6%	72.5%
	claude-3-haiku-20240307	30.1%	53.5%	28.0%	51.7%	32.4%	55.3%
	claude-3-opus-latest	55.2%	69.0%	53.6%	65.5%	56.9%	73.0%
	deepseek-r1-distill-qwen-32b-ugj	49.3%	68.6%	45.1%	61.2%	54.3%	78.2%
	gpt-3.5-turbo-0125	30.3%	52.3%	30.4%	52.2%	30.2%	52.3%
	gpt-4-turbo	49.7%	68.4%	48.0%	62.3%	51.6%	75.9%
	gpt-4o	50.2%	67.9%	47.5%	63.1%	53.1%	73.6%
	gpt-4o-mini	29.9%	53.6%	28.0%	49.2%	32.0%	58.9%
	llama-3-1-8b-instruct-orp	7.3%	36.4%	5.3%	25.2%	11.7%	65.8%
	llama-3-3-70b-instruct-gguf-itb	26.0%	48.2%	23.8%	44.5%	28.7%	52.5%
wikiann_en	mistral-small-24b-instruct-2-vzv	17.1%	38.3%	15.2%	35.8%	19.6%	41.1%
	claude-3-5-haiku-20241022	15.8%	61.3%	14.3%	75.0%	17.8%	51.8%
	claude-3-5-sonnet-20241022	33.1%	73.3%	29.7%	85.1%	37.5%	64.4%
	claude-3-haiku-20240307	15.8%	60.4%	14.1%	74.8%	17.9%	50.7%
	claude-3-opus-latest	44.0%	78.3%	40.8%	86.7%	47.7%	71.3%
	deepseek-r1-distill-qwen-32b-ugj	35.1%	73.6%	31.9%	84.9%	39.1%	64.9%
	gpt-3.5-turbo-0125	0.0%	54.5%	0.0%	75.0%	0.0%	42.9%
	gpt-4-turbo	66.7%	93.3%	66.7%	87.5%	66.7%	100.0%
	gpt-4o	40.0%	66.7%	50.0%	80.0%	33.3%	57.1%
	gpt-4o-mini	0.0%	71.4%	0.0%	71.4%	0.0%	71.4%
	llama-3-1-8b-instruct-orp	6.9%	61.8%	5.2%	57.5%	10.2%	66.9%
wikiann_ru	llama-3-3-70b-instruct-gguf-itb	21.3%	64.5%	18.4%	74.1%	25.3%	57.1%
	mistral-small-24b-instruct-2-vzv	15.0%	56.1%	13.5%	71.8%	16.8%	46.1%
	claude-3-5-haiku-20241022	18.5%	64.8%	15.9%	74.8%	22.1%	57.1%
	claude-3-5-sonnet-20241022	27.1%	72.2%	23.1%	83.8%	32.6%	63.5%
	claude-3-haiku-20240307	13.1%	59.8%	11.2%	69.5%	15.8%	52.4%
	claude-3-opus-latest	32.9%	75.0%	29.5%	84.5%	37.4%	67.4%
	deepseek-r1-distill-qwen-32b-ugj	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	gpt-3.5-turbo-0125	0.0%	57.1%	0.0%	66.7%	0.0%	50.0%
	gpt-4-turbo	50.0%	88.9%	50.0%	100.0%	50.0%	80.0%
	gpt-4o	0.0%	60.0%	0.0%	60.0%	0.0%	60.0%
	gpt-4o-mini	50.0%	66.7%	50.0%	60.0%	50.0%	75.0%
wnut	llama-3-1-8b-instruct-orp	6.1%	59.3%	4.4%	52.4%	9.6%	68.4%
	llama-3-3-70b-instruct-gguf-itb	20.4%	66.6%	17.3%	73.6%	24.9%	60.9%
	mistral-small-24b-instruct-2-vzv	13.5%	57.9%	12.3%	71.7%	15.0%	48.6%
	claude-3-5-haiku-20241022	0.0%	24.5%	0.0%	17.9%	0.0%	39.0%
	claude-3-5-sonnet-20241022	15.2%	35.6%	10.6%	26.5%	27.1%	54.2%
	claude-3-haiku-20240307	0.0%	23.0%	0.0%	16.1%	0.0%	40.3%
	claude-3-opus-latest	13.6%	36.7%	9.6%	26.2%	23.3%	61.5%
	deepseek-r1-distill-qwen-32b-ugj	1.7%	33.0%	1.1%	22.2%	3.3%	64.4%
	gpt-3.5-turbo-0125	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	gpt-4-turbo	2.8%	37.7%	2.1%	26.8%	4.5%	63.2%
	gpt-4o	1.4%	34.5%	1.0%	24.1%	2.3%	60.7%
	gpt-4o-mini	0.2%	27.4%	0.1%	20.2%	0.3%	42.8%
	llama-3-1-8b-instruct-orp	0.1%	9.0%	0.1%	4.9%	0.6%	55.3%
	llama-3-3-70b-instruct-gguf-itb	2.0%	26.8%	1.3%	19.1%	3.7%	44.6%
	mistral-small-24b-instruct-2-vzv	0.0%	17.3%	0.0%	12.4%	0.0%	28.7%

Приложение 4. Распределение меток в синтезированном датасете

[Распределение cat_rel_none]:

O: 2232028
 CLA: 667908
 REL: 105496
 UNK: 31916

[Распределение class_tags]:

O: 2369495	SOQ: 8429	PCS: 2136	CLE: 815	DNE: 303
UNK-C: 125902	ACN: 6258	ULY: 1992	SAE: 736	ATN: 258
ITY: 100113	QIE: 5575	SON: 1954	SLA: 698	PSN: 229
HWE: 81238	TLQ: 4470	CON: 1808	CCY: 651	SDY: 213
SPE: 80533	CUN: 3938	RRE: 1769	TLE: 632	BLE: 151
TNL: 62252	PJT: 3242	PIY: 1472	SOO: 564	AVT: 146
IRE: 49713	PAN: 2772	PRY: 1124	DDY: 515	SET: 115
CAM: 18107	SEN: 2545	LLE: 1076	AFR: 513	OLA: 102
PLT: 16653	BDG: 2539	DER: 974	AIT: 511	QYE: 94
PES: 15758	CPO: 2384	SST: 969	SEE: 481	DYH: 82
DMT: 14996	NCT: 2249	LCE: 935	CMO: 437	
MNS: 13858	RNT: 2179	A-Y: 927	WAY: 409	
STM: 1295	AET: 2159	REY: 873	QAE: 370	

[Распределение rel_tags]:

O: 2931877	HOE: 1536	HGN: 679	HCB: 269	_A: 100
RIS: 39290	HRS: 1494	HPS: 671	HIE: 227	HIY: 67
UNK-R: 30466	HSD: 1255	HVE: 511	HER: 184	HRY: 19
HSR: 15628	HNE: 1157	HME: 469	HET: 174	HNH: 18
HIS: 2782	HNS: 1131	HGS: 440	HOY: 168	
HES: 2361	HFD: 863	DEF: 313	HCN: 143	
UES: 1933	HAT: 720	HEN: 279	HLY: 12	

Приложение 5. Синсеты и их преобразование в объект онтологии

Пример класса и его определение:

'выдал',	'Отмена',	'Сбросили',	'добавила',
'поменяли',	'сохранил',	'сменила',	'Выслал',
'сделал',	'сбросил',	'Добавил',	'сохранил',
'Прописал',	'Поменял',	'Узнал',	'переименую',
'Выслал',	'Создал',	'сбросила',	'Сменил',
'сменила',	'сменила',	'Отправил',	'Отмена',
'сохранил',	'Запросила',	'Изменили',	'Выбрал',
'Сменили',	'ввела',	'Узнал',	'Просит',
'Прописала',	'дал',	'Освободил',	'сменил',
'Сменил',	'Выслал',	'изменений',	'Отмена',
'Протестил',	'Выслать',	'Проверил',	'Включена']
'Сменил',	'Readdressed',	'восстановили',	
'Вывел',	'применятся',	'Запросила',	

Пример отношения и его определение:

'Дмитрием.',	'проверить.',	'тимвивера:.',	'евернот.',
'отсутствует.',	'Вн.',	'отсутствует.',	'отсутствует.',
'отсутствует.',	'доставленно.',	'модем.',	'отсутствует.',
'UEFI.',	'ноутбуком.',	'переподкл.',	'переадрес:.',
'отсутствует.',	'скайп.',	'Дарье.',	'email.',
'отсутствует.',	'окау.',	'hSPjds.',	'год.',
'отсутствует.',	'Planned.',	'ключи.',	'Жанатов.',
'195.',	'issue.',	'аутлук.',	'Астана.',
'Mail.',	'тех.',	'папка.',	'lift.',
'NdqkyRI.',	'битрикс.',	'Запланир.',	'работает.',
'евернот.',	'отсутствует.',	'Дмитрий.',	'эв.'],
'отсутствует.',	'неделю.',	'Джимайл.',	
'Ugpx.',	'отсутствует.',	'Султан.',	

rdfs:label: ModificationProcess

rdfs:comment: A type of process that handles the modification or change of a configuration item, setting, or parameter.

Потенциальные Properties: hasResponsible, hasAccountable, hasInformed, и т. п. (уже определенные в ITSMO для процессов)

owl:DatatypeProperty: hasNote

rdfs:comment: Holds additional notes or short remarks about the resource's current status or environment.

```
<owl:DatatypeProperty rdf:about="http://ontology.it/itsmo/v1#hasNote">
  <rdfs:domain rdf:resource="http://ontology.it/itsmo/v1#RunnableResource"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:label>hasNote</rdfs:label>
  <rdfs:comment>Additional short remark or note about a re-source.</rdfs:comment>
</owl:DatatypeProperty>
```

AUTOMATIC AND SEMI-AUTOMATIC METHODS FOR DOMAIN KNOWLEDGE-GRAPH CONSTRUCTION AND ONTOLOGY EXPANSION

A. P. Khalov¹ [0009-0005-4584-8245], O. M. Ataeva² [0000-0003-0367-5575]

¹Moscow Institute of Physics and Technology (MIPT), Dolgoprudny, Russia

^{1, 2}Federal Research Center “Computer Science and Control” of the Russian Academy of Sciences, Moscow, Russia

¹khalov.a@phystech.edu, ²oataeva@frccsc.ru

Abstract

We present a combined pipeline for knowledge-graph construction and ontology expansion. The approach builds a BIO-tagged corpus via fully automatic LLM-based pseudo-annotation and introduces dedicated UNK reserve categories to capture previously unseen classes and relations. A specialized NER/RE model is trained on a 3-million-token dataset with 92 labels. The model exhibits a conservative quality profile – high precision with moderate recall – suited for safe graph enrichment: integrating the extracted facts expands the graph to ~0.98 million triples, while the expansion ratio (total inferred facts to explicit triples) increases from 2.65 to 3.52, with logical consistency preserved. UNK label pools are converted into stable synsets, enabling semi-automatic ontology expansion; 12 new classes derived from unstructured texts were added. We also demonstrate practical value for querying and analytics using an LLM + SPARQL setup.

Keywords: *ontology, DOLCE, knowledge graph, NER, BIO tagging, RDF/OWL, SPARQL.*

REFERENCES

1. Borgo S. et al. DOLCE: A descriptive ontology for linguistic and cognitive engineering // *Applied Ontology*. 2023. Vol. 17, No. 1. P. 45–69.
2. *IT Service Management Ontology (ITSMO)*. Canonical resolver; catalog entry in LOV “IT Service Management Ontology (itsmo)”. <https://w3id.org/itsmo>; ontology.it; lov.linkeddata.es (Accessed: 08 August 2025).

3. *Khalov A., Ataeva O.* Automating Ontology Mapping in IT Service Management: A DOLCE and ITSMO Integration // *Data Science Journal*. 2025. Vol. 24. P. 23. <https://doi.org/10.5334/dsj-2025-023>
4. *Gruber T.R.* A translation approach to portable ontology specifications // *Knowledge Acquisition*. 1993. Vol. 5, No. 2. P. 199–220. <https://doi.org/10.1006/knac.1993.1008>
5. *Gruber T.R.* Toward principles for the design of ontologies used for knowledge sharing // *International Journal of Human-Computer Studies*. 1995. Vol. 43, No. 5–6. P. 907–928. <https://doi.org/10.1006/ijhc.1995.1081>
6. *Smith B.* *Ontology (Science)* // *Formal Ontology in Information Systems*, IOS Press, 2008. P. 21–35. <https://doi.org/10.1038/npre.2008.2027.2>
7. *Studer R., Benjamins V.R., Fensel D.* *Knowledge Engineering: Principles and Methods* // *Data & Knowledge Engineering*. 1998. Vol. 25, No. 1–2. P. 161–197. [https://doi.org/10.1016/S0169-023X\(97\)00056-6](https://doi.org/10.1016/S0169-023X(97)00056-6)
8. *Hogan A., Blomqvist E., Cochez M. et al.* *Knowledge Graphs*. Morgan & Claypool Publishers, 2021. 257 p.
9. *Barrasa J., Webber J.* *Building Knowledge Graphs: A Practitioner's Guide*. O'Reilly Media, 2023. 250 p.
10. *El Yamami A. et al.* An ontological representation of ITIL framework service level management process // *Proceedings of the 3rd International Conference on Signals, Distributed Systems and Artificial Intelligence (SDSAI 2018)*. 2019. Springer.
11. *Valiente M.-C., Vicente-Chicote C., Rodriguez D.* An Ontology-Based and Model-Driven Approach for Designing IT Service Management Systems // *Int. J. of Service Science, Management, Engineering, and Technology*. 2011. Vol. 2 (2). P. 65–81.
12. *Miwa M., Bansal M.* End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures // *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2016. P. 1105–1116. Berlin, Germany: Association for Computational Linguistics. <https://doi.org/10.18653/v1/P16-1105>
13. *Xu J., Zhang Z., Friedman T., Liang Y., Van den Broeck G.* A Semantic Loss Function for Deep Learning with Symbolic Knowledge // *Proceedings of the 35th International Conference on Machine Learning (ICML)*. PMLR, 2018. Vol. 80. P. 5502–5511. URL: <https://proceedings.mlr.press/v80/xu18h.html>

14. Sun K., Zhang R., Mensah S., Mao Y., Liu X. Learning Implicit and Explicit Multi-task Interactions for Information Extraction // ACM Transactions on Information Systems. 2023. Vol. 41, No. 2. P. 1–29. <https://doi.org/10.1145/3533020>

15. Giunchiglia E., Lukasiewicz T. Coherent Hierarchical Multi-label Classification Networks // Advances in Neural Information Processing Systems 33 (NeurIPS 2020). 2020. URL: <https://proceedings.neurips.cc/paper/2020/file/6dd4e10e3296fa63738371ec0d5df818-Paper.pdf>

16. Yu J., Bohnet B., Poesio M. Named Entity Recognition as Dependency Parsing // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL). 2020. P. 6470–6476. <https://doi.org/10.18653/v1/2020.acl-main.577>

17. Lu Y., Liu Q., Dai D., Xiao X., Lin H., Han X., Sun L., Wu H. Unified Structure Generation for Universal Information Extraction // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2022. P. 5755–5772. <https://doi.org/10.18653/v1/2022.acl-long.395>

18. Gururangan S., Marasović A., Swayamdipta S., Lo K., Beltagy I., Downey D., Smith N. A. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL). 2020. P. 8342–8360. <https://doi.org/10.18653/v1/2020.acl-main.740>

19. Brown T.B. et al. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems. 2020. Vol. 33. URL: <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf>

20. Alizadeh M., Kubli M., Samei Z., Dehghani S., Zahedivafa M., Bermeo J.D., Korobeynikova M., Gilardi F. Open-source LLMs for text annotation: a practical guide for model setting and fine-tuning // Journal of Computational Social Science. 2025. Vol. 8. Article 17. <https://doi.org/10.1007/s42001-024-00345-9>

21. Eiras F. et al. Position: Near to Mid-term Risks and Opportunities of Open-Source Generative AI // Proceedings of the 41st International Conference on Machine Learning (ICML 2024). Proceedings of Machine Learning Research. 2024. Vol. 235. P. 12348–12370. URL: <https://proceedings.mlr.press/v235/eiras24b.html>

22. Tjong Kim Sang, E. F., De Meulder F. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: HLT-NAACL 2003 (CoNLL-2003).

23. Zhang B., May J., Nothman J., Knight K., Ji H. Cross-lingual Name Tagging and Linking for 282 Languages. ACL 2017.

24. Derczynski, L. et al. Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition. W-NUT 2017 (ACL Workshop).

25. Brown T.B. et al. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems. 2020. Vol. 33.

26. Campello R.J.G.B., Moulavi D., Sander J. Hierarchical density estimates for data clustering, visualization, and outlier detection // ACM Transactions on Knowledge Discovery from Data (TKDD). 2015. Vol. 10 (1). P. 5. <https://doi.org/10.1145/2733381>

27. Vardi Y., Zhang C.-H. A modified Weiszfeld algorithm for the Fermat–Weber location problem // Mathematical Programming. 2001. Vol. 90. P. 559–566. <https://doi.org/10.1007/PL00011435>

28. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019. P. 3982–3992. Hong Kong, China: Association for Computational Linguistics. <https://doi.org/10.18653/v1/D19-1410>

29. Hugging Face. XLM-RoBERTa (large): specs (24 layers, ~550M params). 2020–2024.
URL: https://huggingface.co/transformers/v3.4.0/pretrained_models.html

30. Côté M.-A. et al. TextWorld: A Learning Environment for Text-Based Games // Computer Games (CGW@IJCAI 2018). 2019. Vol. 1017 (CCIS). P. 41–75. https://doi.org/10.1007/978-3-030-24337-1_3

31. Russell S., Norvig P. Artificial Intelligence: A Modern Approach. 4th ed. Pearson, 2020. Chapter 11: Planning and Acting.

32. Schmidhuber J. Gödel Machines: Self-Referential Universal Problem Solvers Making Provably Optimal Self-Improvements // Artificial General Intelligence. 2007. P. 199–226. https://doi.org/10.1007/978-3-540-68677-4_7

33. Yin X. et al. Gödel Agent: A Self-Referential Agent Framework for Recursively Self-Improvement // Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2025). 2025. P. 27890–27913. <https://aclanthology.org/2025.acl-long.1354/>

34. *Ataeva O.M., Serebryakov V.A.* Ontology of the Digital Semantic Library Lib-Meta // Informatics and Its Applications. 2018. Vol. 12, No. 1. P. 2–10 (In Russian).

СВЕДЕНИЯ ОБ АВТОРАХ



ХАЛОВ Андрей Петрович – аспирант МФТИ (ФПМИ), кафедры «Интеллектуальные системы». Область научных интересов: онтологическое моделирование, графы знаний, извлечение знаний из текстов (NER/RE, RAG), многоагентные системы и планирование, применение LLM в корпоративных ИС.

Andrey Petrovich KHALOV – PhD student, Moscow Institute of Physics and Technology (MIPT), Phystech School of Applied Mathematics and Informatics, Department of Intelligent Systems. Research interests: ontological modeling, knowledge graphs, information extraction from text (NER/RE, RAG), multi-agent systems and planning, application of LLMs in enterprise information systems.

email: khalov.a@phystech.edu

ORCID: 0009-0005-4584-8245



АТАЕВА Ольга Муратовна – старший научный сотрудник Вычислительного центра им. А.А. Дородницына Федерального исследовательского центра «Информатика и управление» Российской академии наук, к. т. н. Область научных интересов: системное программирование, базы данных, инженерия знаний и онтологии.

Olga Muratovna ATAIEVA – Senior Researcher, Dorodnitsyn Computing Center, Federal Research Center “Computer Science and Control” of the Russian Academy of Sciences (FRC CSC RAS); Candidate of Technical Sciences (PhD). Research interests: systems programming, databases, knowledge engineering, ontologies.

email: oataeva@frccsc.ru

ORCID: 0000-0003-0367-5575

Материал поступил в редакцию 11 ноября 2025 года