20. *Viet N.T., Kravets A.G.* Algoritm raboty web-kraulera dlya resheniya zadachi sbora dannykh iz otkrytykh internet istochnikov [The algorithm of the web crawler for solving the problem of collecting data from open Internet sources], *Izvestiya Sankt-Peterburgskogo gosudarstvennogo tekhnologicheskogo instituta (tekhnicheskogo universiteta)* [Izvestiya of Saint-Petersburg State Technological Institute (Technical University)], 2019, No. 51 (77), pp. 115-119. DOI: 10.36807/1998-9849-2019-51-77-115-119.
21. *Kozina S.A., Kulinchenko I.A., Korobkin D.M., Fomenkov S.A.* Kontseptsiya i arkhitektura parsinga i khraneniya edinoy bazy patentov i nauchnykh zhurnal'nykh publikatsiy [The concept and architecture of parsing and storing a unified database of patents and scientific journal publications], *Modelirovanie, optimizatsiya i informatsionnye tekhnologii* [Modeling, Optimization and Information Technology], 2024, Vol. 12, No. 4, 15 p. DOI: 10.26102/2310-6018/2024.47.4.024. Available at: https://moitvivt.ru/ru/journal/pdf?id=1740.

Бондаренко Артём Геннадьевич – Волгоградский государственный технический университет; e-mail: temdit01@yandex.ru; г. Волгоград, Россия; тел.: +79375596156; кафедра систем автоматизированного проектирования и поискового конструирования; магистрант.

Кравец Алла Григорьевна – Волгоградский государственный технический университет; e-mail: AllaGKravets@yandex.ru; г. Волгоград, Россия; тел.: +79023639186; кафедра систем автоматизированного проектирования и поискового конструирования; д.т.н.; профессор.

Bondarenko Artem Gennadevich – Volgograd State Technical University; e-mail: temdit01@yandex.ru; Volgograd, Russia; phone: +79375596156; the Department of CAD&RD; master student.

Kravets Alla Grigorievna – Volgograd State Technical University; e-mail: AllaGKravets@yandex.ru; Volgograd, Russia; phone: +79023639186; the Department of CAD&RD; dr. of eng. sc.; professor.

УДК 004.89 DOI 10.18522/2311-3103-2025-3-159-171

А.М. Мансур, Ж.Х. Мохаммад, Ю.А. Кравченко

РАЗРАБОТКА ЧАТ-БОТА ДЛЯ КЛАССИФИКАЦИИ И АНАЛИЗА ТЕКСТОВ НА ЕСТЕСТВЕННОМ ЯЗЫКЕ С ИСПОЛЬЗОВАНИЕМ ЛОКАЛЬНЫХ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ

Исследуются локальные большие языковые модели (Local large language models, Local LLM) и их применение в задачах классификации текста, а также проводится сравнение их производительности с традиционными методами. Статья предоставляет всесторонний обзор ряда ключевых локальных LLM, уделяя особое внимание их архитектурным преимуществам, характеристикам и областям применения. В частности, рассматриваются модели с различным количеством параметров, их способность адаптироваться к специализированным доменам, а также требования к вычислительным ресурсам при их развертывании на локальном оборудовании. Особый акцент делается на компромиссах между производительностью и эффективностью использования ресурсов. В качестве практического вклада разработан чат-бот, использующий локальные LLM (такие как DeepSeek, Gemma и Llama2 через Ollama) для классификации входящих текстов по заранее заданным категориям, демонстрируя работу этих моделей без использования облачных вычислений. Система реализована с модульной архитектурой, позволяющей легко интегрировать новые модели и сравнивать их эффективность. Вычислительный эксперимент включает оценку точности и скорости вывода локальных LLM в сравнении с более простыми методами, такими как Sentence-BERT, TF-IDF и BoWC, выделяя сценарии, в которых локальные модели превосходят традиционные подходы или уступают им. Тестирование проводилось на основе эталонного набора данных BBC. Результаты показывают, что языковые модели (включая модели с 7 миллиардами параметров) демонстрируют сильную и логически обоснованную классификационную производительность при обработке текстов на естественном языке, однако их результаты не являются идеальными для эталонных наборов данных. В частности, обнаружены случаи, когда все тестируемые модели, включая традиционные методы, ошибочно классифицировали документы, что указывает на возможные проблемы в разметке данных. Полученные результаты указывают на необходимость пересмотра эталонных меток в стандартных наборах данных. Это особенно

важно для доменов с субъективными категориями, где экспертные оценки могут значительно расходиться. С другой стороны, хотя локальные LLM уступают облачным в скорости, их преимущества в конфиденциальности данных и оффлайн-работе делают их пригодными для специализированных задач.

Локальные большие языковые модели; БЯМ; классификация; Ollama; Sentence-BERT; BoWC; децентрализованный ИИ.

A.M. Mansour, J.H. Mohammad, Yu.A. Kravchenko

DEVELOPMENT OF A CHATBOT FOR CLASSIFICATION AND ANALYSIS OF NATURAL LANGUAGE TEXTS USING LOCAL LARGE LANGUAGE MODELS

This paper explores local large language models (LLMs) and their application in text classification tasks, while also comparing their performance with traditional methods. The paper provides a comprehensive review of several key local LLMs, with particular focus on their architectural advantages, characteristics, and application domains. Specifically, we examine models with varying numbers of parameters, their ability to adapt to specialized domains, and their computational requirements when deployed on local hardware. Special emphasis is placed on the trade-offs between performance and resource efficiency. As a practical contribution, we developed a chatbot that utilizes local LLMs (such as DeepSeek, Gemma, and Llama2 via Ollama) to classify incoming texts into predefined categories, demonstrating the operation of these models without cloud computing. The system features a modular architecture that allows for easy integration of new models and comparison of their effectiveness. The computational experiment involves evaluating the accuracy and inference speed of local LLMs compared to simpler methods such as Sentence-BERT, TF-IDF and BoWC, highlighting scenarios in which local models outperform or underperform traditional approaches. Testing was conducted using the benchmark BBC dataset. The results show that language models (including 7-billion parameter models) demonstrate strong and logically consistent classification performance in natural language text processing. However, their results are not perfect for benchmark datasets. Notably, we identified cases where all tested models, including traditional methods, misclassified documents, suggesting potential issues with data labeling. These findings indicate the need to reconsider benchmark labels in standard datasets, particularly for domains with subjective categories where expert evaluations may vary significantly. On the other hand, while local LLMs lag behind cloud-based solutions in speed, their advantages in data privacy and offline operation make them suitable for specialized tasks. This is particularly valuable in medical and financial institutions where protection of sensitive information is critical, and where local models can be fine-tuned for specific business processes without the constraints of cloud APIs.

Local large language models; LLM; classification; Ollama; SBERT; BoWC; decentralized AI.

Введение. Стремительное развитие больших языковых моделей (LLM) произвело революцию в области обработки естественного языка (NLP), обеспечив значительный прогресс в таких задачах, как классификация текста и разработка диалоговых систем. Хотя облачные модели, такие как GPT-4, в настоящее время доминируют в этой сфере, их зависимость от внешних API порождает серьёзные проблемы, связанные с конфиденциальностью данных, задержками при обработке и эксплуатационными расходами. Это стимулирует растущий интерес к локальным LLM – моделям машинного обучения, способным понимать и генерировать естественно-языковые тексты исключительно на пользовательском оборудовании (например, персональных компьютерах, серверах или периферийных устройствах), без необходимости подключения к облачным сервисам [1, 2].

Классификация текста представляет собой базовую задачу обработки естественного языка (NLP), суть которой заключается в отнесении текстовых данных к заранее определённым категориям на основе их содержания [3]. Данный процесс играет ключевую роль в извлечении знаний и поддержке принятия решений, поскольку позволяет выявлять ценную информацию в массивах текстовых данных. В условиях экспоненциального роста объёмов цифровой информации значимость эффективных методов текстовой классификации существенно возрастает.

Компактные, но эффективные архитектуры, такие как LLaMA2 [4], DeepSeek [5, 6] и Gemma [7], демонстрируют высокую универсальность в классификации неструктурированных данных благодаря использованию методов zero-shot и few-shot обучения.

В этих подходах модели работают либо исключительно на основе инструкций, либо с минимальным количеством размеченных примеров, что позволяет быстро адаптироваться к новым предметным областям. Локальные LLM предлагают существенные преимущества, включая контроль над данными, возможность кастомизации и автономную функциональность.

Несмотря на эти преимущества, существуют серьезные проблемы, связанные с точностью, вычислительной эффективностью и практическим применением, которые требуют дальнейшего совершенствования. Предыдущие исследования подтвердили полезность локальных LLM для задач классификации, но большинство работ сосредоточено на отдельных моделях или узких приложениях, не предлагая всестороннего сравнения различных архитектур. В частности, отсутствуют систематические сравнительные оценки с традиционными методами, такими как Sentence-BERT [8], TF-IDF или Bag-of-Weighted-Concepts (BoWc) [9–11], которые остаются конкурентоспособными благодаря своей проверенной эффективности и оптимальному использованию ресурсов. Это особенно важно для специалистов, которым необходимо учитывать компромиссы между производительностью и аппаратными ограничениями в реальных сценариях.

В данной работе проводится эмпирическое сравнение современных локальных LLM (включая DeepSeek R1, Gemma и LLaMA2, реализованные через Ollama) с традиционными методами при решении задачи классификации текстов. Исследование оценивает:

- эффективность LLM в задачах классификации текстов;
- скорость вывода, объем используемой памяти и требования к оборудованию;
- особенности развертывания LLM в периферийных и серверных средах.

Проведенный анализ предоставляет практические рекомендации по выбору моделей LLM на основе баланса между точностью, скоростью и ресурсозатратностью.

1. Архитектура больших языковых моделей (LLM). Современные LLM используют трансформерную архитектуру [12], которая состоит из:

А. Слой встраивания (англ. embedding layer). Слой встраивания выполняет критически важную функцию преобразования текстовых данных в числовое представление, понятное нейронной сети. На этом этапе исходный текст сначала разбивается на токены (отдельные слова или части слов), которые затем переводятся в плотные векторные представления – встраивания **(**эмбеддинги). Этот процесс включает два основных компонента: токенные встраивания, отображающие семантические свойства языковых единиц в многомерном векторном пространстве, и позиционные встраивания, кодирующие информацию о порядке следования элементов в последовательности, что принципиально важно для понимания контекста и синтаксических отношений.

*Б. Блоки-трансформеры (англ. Transformer Blocks)***.** Основу архитектуры составляют блоки-трансформеры, организованные в виде последовательности идентичных слоёв. Каждый такой блок содержит три ключевых элемента. Во-первых, механизм многоголовой само-внимательности (Self-Attention) анализирует взаимосвязи между всеми словами входной последовательности, используя матрицы Query (запросов), Key (ключей) и Value (значений) для динамического определения значимости каждого элемента контекста. Этот механизм обеспечивает параллельную обработку последовательностей и выявление сложных семантических зависимостей. Во-вторых, прямая нейронная сеть (Feed-Forward Network) выполняет нелинейное преобразование выходов механизма внимания, обычно реализуемое как двухслойная структура с активацией ReLU или GELU. В-третьих, система остаточных связей и слой нормализации (Residual + LayerNorm) стабилизирует процесс обучения: остаточные соединения позволяют передавать исходные данные через несколько слоёв, предотвращая проблему затухающих градиентов, а нормализация активаций способствует более устойчивому обучению.

С. Выходной слой (англ. Output layer). Завершающий этап обработки - выходной слой – преобразует полученные скрытые состояния в вероятностное распределение над словарём модели. Используемая здесь функция активации softmax обеспечивает нормализацию выходных значений, превращая их в вероятности следующего токена, что позволяет модели генерировать осмысленные и когерентные продолжения текста. Эта ар-

хитектурная схема, сочетающая мощные механизмы анализа контекста с эффективными методами нормализации, лежит в основе современных достижений в области обработки естественного языка. В табл. 1 описаны основные компоненты, составляющие каждую большую языковую модель, и функции каждого из них.

Таблица 1

Ключевые компоненты LLM

Компонент	Назначение
Токенизация	Разбивает текст на подслова/символы
Маска внимания	Управляет видимостью токенов (например, для паддинга/авторегрессии)
KV-кэш	Сохраняет предыдущие состояния внимания (оптимизация инференса)
Адаптеры LoRA	Облегчённые слои для тонкой настройки (снижают потребление памяти)

Генерация выходного текста реализуется через авто-регрессивный процесс [13, 14]: модель последовательно предсказывает наиболее вероятные следующие токены, используя различные стратегии декодирования (жадный поиск для максимальной детерминированности, *beam search* для баланса качества и разнообразия, или вероятностное сэмплирование для творческих задач), при этом каждый новый токен рекурсивно включается в контекст для генерации последующих элементов, что обеспечивает когерентность и связность выходного текста.

Математическая основа описывается формулой (1):

$$\text{Attention}(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \quad (1)$$

где d_k – размерность ключевых векторов.

Далее приведены некоторые примеры для локальных больших языковых моделей.

2. Локальные открытые большие языковые модели (Local LLM). Локальные LLM демократизируют ИИ, предоставляя пользователям мощные инструменты прямо в их руки – хотя и с компромиссами в скорости и масштабируемости по сравнению с облачными аналогами. Ниже приводится описание набора распространенных локальных языковых моделей, которые будут использоваться в данном исследовании.

Llama 2 [4] – это большая языковая модель с открытым исходным кодом, разработанная компанией Meta совместно с Microsoft, которая предназначена для генерации и понимания текста на естественном языке. Она использует архитектуру трансформеров с авторегрессией и дополнительно обучается с помощью методов обучения с подкреплением и обратной связью от человека (RLHF), что повышает её безопасность и эффективность. Llama 2 доступна в различных вариантах с количеством параметров до 70 миллиардов и подходит для решения задач от программирования до творческого письма.

DeepSeek R1 [5] – представляет собой серию крупномасштабных языковых моделей (LLM), посвящённых созданию экономически эффективных ИИ-решений с улучшенными возможностями логического и эмерджентного мышления. В рамках серии были разработаны такие модели, как DeepSeek-LLM, DeepSeek-V2, DeepSeek-V3 и DeepSeek-R1, каждая из которых демонстрирует прогресс в области машинного обучения, особенно в крупномасштабном обучении с подкреплением (RL).

DeepSeek-R1 – это передовая LLM, оптимизированная для сложных логических и структурированных рассуждений. В отличие от традиционных подходов, она использует методы RL с минимальным предварительным обучением с учителем (включая режим «Zero Supervised Fine-Tuning»), что позволяет развивать расширенные способности к цепочечным рассуждениям (chain-of-thought) и саморефлексии. Модель применяет архитектуру смешанных экспертов (англ. *Mixture of Experts*, *MoE*), обеспечивая высокую эффективность и масштабируемость при решении задач в математике, программировании и логическом анализе.

DeepSeek-R1 успешно конкурирует с закрытыми коммерческими аналогами (такими как OpenAI o1), превосходя или соответствуя их показателям в тестах на логическое мышление, оставаясь при этом открытой, прозрачной и экономически выгодной альтернативой. Её способность наглядно демонстрировать ход рассуждений делает её особенно ценной для научных исследований и разработки ПО. Серия DeepSeek в целом иллюстрирует, как совместная оптимизация алгоритмов и аппаратного обеспечения может значительно повысить производительность LLM, способствуя демократизации передового ИИ.

Gemma [7] – это крупномасштабная языковая модель (LLM), разработанная для универсального применения в задачах обработки естественного языка. Модель выделяется высокой точностью в генерации текстов, понимании контекста и адаптации к различным стилям общения благодаря использованию современного трансформерного архитектурного подхода. Gemma применяется в широком спектре областей, включая автоматический перевод, создание контента и анализ текста, обеспечивая поддержу многоязычности и устойчивость к шуму данных.

Особенностью *Gemma* является сбалансированное сочетание производительности и ресурсной эффективности, что делает её удобной для интеграции в коммерческие и исследовательские проекты. Благодаря оптимизированным алгоритмам обучения, модель демонстрирует способность быстро обучаться на дополнительной информации и адаптироваться к новым задачам без значительных потерь качества. Это позволяет использовать Gemma как надёжный инструмент для ускорения работы с большими объёмами текстовой информации и улучшением качества взаимодействия с пользователями. Таким образом, Gemma представляет собой мощный и гибкий инструмент в арсенале современных LLM, сочетающий высокие показатели понимания текста с доступностью и удобством эксплуатации.

В целом, разницу между локальными и облачными LLM-моделями можно обобщить в следующих ключевых пунктах, представленных в табл. 2. Локальные модели обеспечивают приватность, оффлайн-работу и кастомизацию, но уступают в производительности и скорости. Облачные решения (например, GPT-4) предлагают высокую точность и минимальные задержки, однако зависят от сторонних серверов, платных API и интернет-соединения. Локальные модели дают полный контроль над данными, но требуют мощного железа, тогда как облачные легко масштабируются – ценой конфиденциальности и регулярных платежей.

Таблица 2

Сравнение: локальные и облачные LLM

Характеристика	**Локальные LLM (LLaMA 2, DeepSeek)**	**Облачные LLM (GPT-4, Claude)**
Конфиденциальность	✅ Данные не покидают устройство	❌ Обработка на сторонних серверах
Оффлайн-работа	✅ Полная функциональность без интернета	❌ Требуется постоянное подключение
Производительность	⚠ Зависит от мощности оборудования	✅ Максимальная производительность
Кастомизация	✅ Полная свобода модификации	❌ Ограничена провайдером
Стоимость	✅ Разовые затраты на оборудование	❌ Плата за использование (подписка)
Масштабируемость	⚠ Ограничена локальными ресурсами	✅ Автоматическое масштабирование
Точность	⚠ Хорошая, но ниже облачных аналогов	✅ Передовые результаты

Для локального запуска больших языковых моделей (LLM) на ПК используется Ollama – это платформа для локального запуска открытых языковых моделей (LLM), таких как Llama2, llama2 и DeepSeek, на macOS, Linux и Windows. Она упрощает развертывание, объединяя веса моделей, конфигурации и данные в готовые оптимизированные пакеты с поддержкой GPU-ускорения.

3. Постановка задачи. Дан набор текстовых документов $\boldsymbol{D} = \{d_1, d_2, ..., d_{|D|}\}$, требующих классификации по заданным категориям (классам) $\boldsymbol{C} = \{\mathbf{c_1, c_2, ..., c_{|C|}}\}$.

Первичная классификация, выполняется с помощью традиционной модели (например, SBERT), которая выявляет подмножество документов с ошибочной классификацией. Для неверно классифицированных документов необходимо оценить точность больших языковых моделей (LLM) в анализе текстового содержания и корректном отнесении документов к истинным категориям.

Задача классификации заключается в построении аппроксимирующей функции $\boldsymbol{\Phi'}$, максимально приближенной к целевой функции $\boldsymbol{\Phi: D \times C \rightarrow \{0, 1\}}$, определяющая принадлежность документа к категории (1 – принадлежит, 0 – не принадлежит). Для решения данной задачи используются методы машинного обучения, которые опираются на наличие коллекции заранее классифицированных документов $\boldsymbol{\Omega} = \{d_1, d_2, ..., d_{|\Omega|}\}$.

Тогда задача состоит в нахождении такой функции $\boldsymbol{\Phi'}$, которая минимизирует суммарные потери на коллекции $\boldsymbol{\Omega}$:

$$\min_{\Phi'} \sum_{\mathbf{d} \in \Omega} \mathbf{L}(\Phi(\mathbf{d}), \Phi'(\mathbf{d})),$$

где $\mathbf{L}(\Phi(\mathbf{d}), \Phi'(\mathbf{d}))$ – функция потерь, оценивающая качество аппроксимации $\boldsymbol{\Phi'}$ относительно $\boldsymbol{\Phi}$ для документа $\mathbf{d}$.

Результаты позволят определить, может ли дополнение традиционных моделей LLM повысить точность классификации, или же требуются гибридные подходы для баланса между точностью, эффективностью и затратами.

4. Архитектура чат-бота для классификации текста с использованием локальных LLM. В данном разделе представлено проектирование чат-бота на основе LLM, предназначенного для классификации пользовательских текстов на одну из $\boldsymbol{N}$ предопределенных категорий с использованием локально размещенных больших языковых моделей (LLM), таких как DeepSeek, Gemma и llama2. Система использует *Ollama* для локального вывода моделей и *FastAPI* [15] для обеспечения структурированного *backend API*, обеспечивающего взаимодействие между пользовательским интерфейсом и языковыми моделями (рис. 2).

1. Пользовательский интерфейс (фронтенд), доступный через веб-приложение или командную строку, обеспечивает ввод текста и отображение результатов классификации.

2. Бэкенд на FastAPI выступает центральным связующим звеном, предоставляя RESTful API для выбора моделей (DeepSeek, Gemma или Llama 2), обработки текстовых запросов и мониторинга состояния системы, с обязательной валидацией данных через Pydantic.

3. Интеграция с Ollama (Слой вывода LLM): Система взаимодействует с локальным сервером вывода Ollama (localhost:11434), отправляя хорошо структурированные промпты выбранной языковой модели для обеспечения согласованного вывода классификации. Полученные сырые ответы модели обрабатываются для извлечения предсказанной категории, что гарантирует согласованный формат вывода.

4. Инженерия промптов (англ. Prompt engineering) – это методика, заключающаяся в тщательном формулировании инструкций для оптимального управления выводом больших языковых моделей (LLM) с целью получения желаемых результатов [16, 17]. В нашем исследовании промпты разрабатываются для работы в условиях zero-shot обучения, где:

1. *Метод Zero-Shot (ZS)*: В данном подходе модель получает исключительно базовую инструкцию для выполнения задачи, без каких-либо дополнительных указаний или примеров. В таких условиях языковая модель опирается исключительно на свои предварительно полученные знания в ходе предобучения (pre-training), чтобы сгенерировать ответ [18, 19].

2. *Метод Few-Shot (FS)*: В отличие от Zero-Shot, этот метод предоставляет модели не только основную инструкцию, но и небольшое количество демонстрационных примеров (обычно от 2 до 5), которые помогают "настроить" её понимание задачи и улучшить качество ответа.

Инженерия промптов для классификации: Входной текст преобразуется в строго формализированный промпт, явно инструктирующий LLM отнести текст к одной из пяти предопределенных категорий. Промпт специально ограничивает ответ модели только названием категории, исключая избыточные объяснения. Пример промпта:

```
22      def generate_answer(self, model, user_input: str, categories) -> str:
23          """Generate answer using Ollama """
24          prompt = f"""
25          Classify the following text into one of these categories: {', '.join(categories)}.
26          Text: {user_input}
27          Respond ONLY with category name.
28          """
```

Рис. 1. Пример промпта для классификации текста

Для повышения эффективности система реализует LRU-кэш (Least Recently Used, наименее недавно использованный), сохраняющий результаты последних классификаций и минимизирующий избыточные запросы к LLM при повторяющихся входах.

На представленной схеме (рис. 2) детально отображен процесс обработки запросов в системе классификации текста. Архитектура реализует последовательный рабочий процесс (англ. workflow), начинающийся с этапа взаимодействия пользователя с фронтенд-интерфейсом, и заканчивая получением результата классификации.

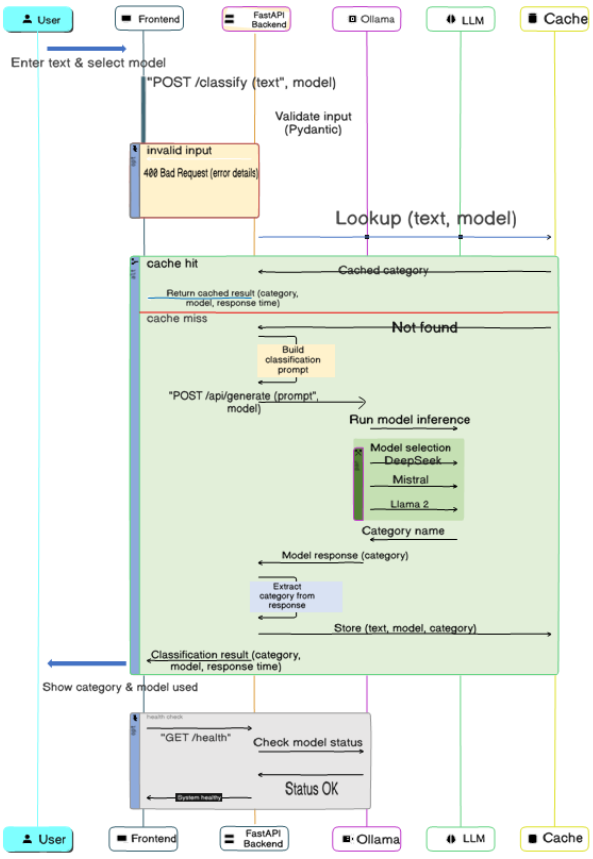

Рис. 2. Процесс обработки запросов в системе классификации текста

Пользователь отправляет текст через фронтенд-интерфейс, указывая предпочитаемую языковую модель. Полученный запрос поступает в FastAPI бэкенд, где происходит валидация входных данных и формирование структурированного промпта для классификации. Сформированный промпт передается в Ollama, которая выполняет локальный вывод выбранной LLM-модели. После обработки запроса Ollama возвращает сырой ответ модели в виде названия категории.

На стороне бэкенда осуществляется извлечение предсказанной категории из ответа модели. При активированном механизме кэширования результат сохраняется для оптимизации обработки идентичных запросов в будущем.

Окончательный результат, содержащий название категории, информацию о используемой модели и времени отклика, отправляется обратно во фронтенд, где отображается пользователю.

Развертывание моделей. Для корректной работы системы необходимо предварительно запустить локальный сервер Ollama с загруженными моделями (DeepSeek, Gemma, Llama 2). Сервер FastAPI должен быть развернут с реализацией механизмов обработки ошибок на случай недоступности Ollama или отсутствия требуемых моделей.

Данная архитектура обеспечивает гибкую и эффективную структуру для развертывания локально размещенного чат-бота для классификации текста на основе LLM. Используя FastAPI для бэкэнд-коммуникации и Ollama для локального вывода, система обеспечивает конфиденциальность, низкую задержку и офлайн-функциональность. Будущие улучшения могут включать тонкую настройку моделей для лучшей точности классификации, интеграцию более сложного фронтенда и сравнительный анализ производительности в различных конфигурациях оборудования.

5. Вычислительный эксперимент и анализ полученных результатов. Для проведения эксперимента и оценки рассматриваемых методов в задаче классификации текстов использовался набор данных BBC [20], который включает 2225 документов, полученных с веб-сайта новостей *BBC* за период 2004-2005 гг. (табл. 3). Набор данных охватывает пять тем: спорт (Sport, S), бизнес (Business, B), развлечения (entertainment, E), политику (Politics, P) и технологии (Tech, T).

Таблица 3

Набор использованных данных BBC

Набор данных	***BBC***
Количество документов	2225
Количество категорий	5
Среднее количество слов в документе	2262
Среднее количество словарных слов в одном документе	207

Для реализации классификатора на основе традиционных методов векторизации были выбраны SBERT, BoWC и TF-IDF для построения векторов документов набора данных BBC. В качестве классификатора был выбран алгоритм опорных векторов (SVM) на основе результатов предыдущих работ авторов [9–11].

Также, реализован классификатор на основе больших языковых моделей, использующий архитектурное решение, подробно описанное в разделе 2. Экспериментальная оценка проводилась для следующих современных больших языковых моделей:

Gemma3:1b – это предпоследняя версия модели Gemma3, представляющая собой наиболее производительный вариант, способный эффективно работать на одной видеокарте (GPU).

DeepSeek-R1:7b (DS-r1:7b) и DeepSeek-R1:1.5b (DS-r1:1.5b) – это версии крупномасштабной языковой модели DeepSeek-R1, разработанные с использованием методов обучения с подкреплением для улучшения рассуждений и решения сложных задач в математике, программировании и логике. DeepSeek-R1 показывает производительность,

сопоставимую с OpenAI-o1, и включает модели с разным числом параметров, в том числе версии 1.5B и 7B, которые являются уменьшенными дистиллятами оригинальной модели, обеспечивая хорошую производительность при меньших вычислительных ресурсах и оставаясь открытыми и доступными для коммерческого использования

Llama2:7B – это одна из моделей серии Llama 2, представляющая собой крупномасштабную языковую модель с 7 миллиардами параметров, разработанную Meta.

Метрика оценки классификации. F-мера – одна из распространенных метрик для оценки успешности классификатора [21]. Это среднее гармоническое между точностью и полнотой, которое определяется следующими формулами:

$$\text{Precision}_{C_i} = \frac{\text{TP}_{C_i}}{\text{TP}_{C_i}+\text{FP}_{C_i}}, \quad (1)$$

$$\text{Recall}_{C_i} = \frac{\text{TP}_{C_i}}{\text{TP}_{C_i}+\text{FN}_{C_i}}, \quad (2)$$

$$\text{мера } F1 = \frac{(1+\beta)\cdot\text{Precision}_{C_i}\cdot\text{Recall}_{C_i}}{\beta\cdot\text{Precision}_{C_i}+\text{Recall}_{C_i}}. \quad (3)$$

Здесь C – это метка класса, True Positive TP – это количество документов, правильно отмеченных классификатором как относящиеся к классу C, а False Positive FP – это количество документов, неправильно отмеченных классификатором как относящиеся к классу C. Между тем, False Negative FN ложноотрицательный – это количество документов, которые относятся к классу C и которые классификатор ошибочно определил как не относящиеся к классу C, а True Negative TN – это количество документов, не принадлежащих к классу C, правильно отмеченных классификатором как не относящиеся к классу C.

Результаты. В табл. 4 представлены сравнительные результаты классификации документов из набора данных BBC, полученные с использованием традиционных методов векторизации текстов. Наибольшую эффективность продемонстрировали методы SBERT и BoWC, что подтверждается метриками точности по мере F1.

Таблица 4

Точность классификации документов по мере F1

Метод (размер вектора)	мера F1 (%)
SBERT (768)	98.2
BoWC (200)	98.2
TF-IDF (26000)	97.89

На основе проведенного анализа были выделены 13 ошибочно классифицированных документов, которые вместе с перечнем пяти возможных категорий были переданы чат-боту для оценки способности больших языковых моделей (LLM) корректно определять категории текстов.

Несмотря на эффективность больших языковых моделей (LLM) в анализе текста, они не смогли корректно классифицировать все документы, используя только названия категорий. В табл. 5 и 6 представлены результаты классификации документов, с которыми традиционные методы обработки допустили ошибки. В табл. 5 показана точность каждой модели, а также объем используемой модели для вычислений. С другой стороны, в таблице 6 показаны подробные сведения о метках классов для каждой модели в сравнении с правильными метками.

Модель DeepSeek-R1:1.5b демонстрирует более высокую точность классификации документов по сравнению с DeepSeek-R1:7b. Однако данное наблюдение отражает не абсолютное превосходство архитектуры, а специфику анализа семантических признаков, существенных для конкретной задачи категоризации.

Таблица 5

Точность классификации выборки документов с использованием локальных LLM, измеренная мерой F1

Модель LLM	Размер модели (ГБ)	F1 (%)
Gemma3:1b	0.82	39.23
Deepseek r1:1.5b	1.1	**41.76**
Deepseek-r1:7b	4.7	35.77
Llama2:7b	4	15.38

Отмечается также, что модель Gemma, несмотря на свой небольшой размер (0,82 ГБ), обеспечивает баланс между точностью и потреблением памяти по сравнению с DeepSeek в обеих версиях.

Интересно, что существуют документы, которые все языковые модели классифицируют ошибочно. Например, в случае с документом 12 четыре модели вместе с традиционными методами (BoWC, SBERT) единогласно отнесли его к категории «политика», тогда как на самом деле он принадлежит к классу «развлечения». Это вызывает сомнения в правильности меток, установленных экспертами.

Таблица 6

Метки классов для каждой модели, по сравнению с правильными метками

	Метка класса	SBERT	BoWC	DS-r1:7b	Gemma	DS-r1:1.5b	Llama2
0	B	T	T	B	T	B	T
1	E	T	T	E	E	E	T
2	P	B	B	P	T	P	P
3	E	T	T	B	**S**	E	T
4	E	**T**	**T**	**B**	**B**	**T**	**T**
5	T	E	E	E	T	E	T
6	B	T	T	T	B	T	T
7	B	**T**	**T**	**P**	**P**	**T**	**T**
8	B	P	P	P	P	T	P
9	P	T	T	P	T	P	P
10	B	P	P	P	B	P	P
11	P	B	B	P	P	B	P
12	**E**	**P**	**P**	**P**	**T**	**P**	**P**

Для анализа ошибочной классификации документа (12) к категории "политика" вместо "развлечения" мы исследовали цепочку рассуждений (Chain-of-Thought) модели DeepSeek-к1:7b. Кроме того, чат-боту был задан следующий аналитический вопрос: «Какие признаки в тексте заставили вас отнести этот контент к политике, а не к бизнесу (business)?» Образец ответа чат-бота (Deepseek-r1: latest) на вопрос представлен на рис. 3.

Проведенный анализ свидетельствует, что языковые модели демонстрируют удовлетворительную, но не оптимальную эффективность классификации. Результаты указывают на необходимость пересмотра эталонной разметки в стандартных наборах данных. Несмотря на способность LLM анализировать категории (что подтверждается chain-of-thought рассуждениями), их точность при работе исключительно с названиями классов остается ограниченной. Для улучшения показателей рекомендуется: Few-Shot обучение с демонстрационными примерами, расширенные описания категорий, включающие семантические маркеры и пограничные случаи.

Что касается времени выполнения, были зафиксированы временные затраты каждой модели на обработку указанных документов. Анализ показал, что все протестированные модели демонстрируют четкую положительную корреляцию между объемом текста и временем выполнения, однако степень этой зависимости значительно варьируется (рис. 4).

[Вот анализ ключевых признаков, которые определили классификацию текста как "политика", а не "бизнес":
1. Государственные институты и политические решения
- Упоминание Pensions Policy Institute (PPI) — аналитического центра, влияющего на госполитику
- Ссылка на отчет Pensions Commission — официального органа по реформе пенсий
- Предложения о "гражданской пенсии" — изменение законодательной системы
2. Фокус на социально-экономической политике
- Обсуждение госгарантий (госпенсия vs. частные накопления)
- Темы неравенства и бедности — классические политические вопросы
- Варианты реформ через налоги или повышение пенсионного возраста — инструменты госрегулирования
3. Отсутствие бизнес-контекста
Нет признаков, характерных для категории "бизнес":
- Не анализируются корпоративные пенсионные программы
- Нет данных о рынке финансовых услуг
- Не упоминаются компании или прибыльность реформ
Пример из текста:
"Замена госпенсии на гражданскую пенсию сократит неравенство" — это политическое предложение, так как:
1. Требует изменений в законах
2. Нацелено на общественное благо
3. Инициируется госорганами
Для отнесения к "бизнесу" текст должен содержать:
- Анализ влияния на частные пенсионные фонды
- Финансовые показатели страховых компаний
- Рыночные механизмы (например, инвестиционные варианты)]

Рис. 3. Ответ чат-бота (Deepseek-r1: latest)

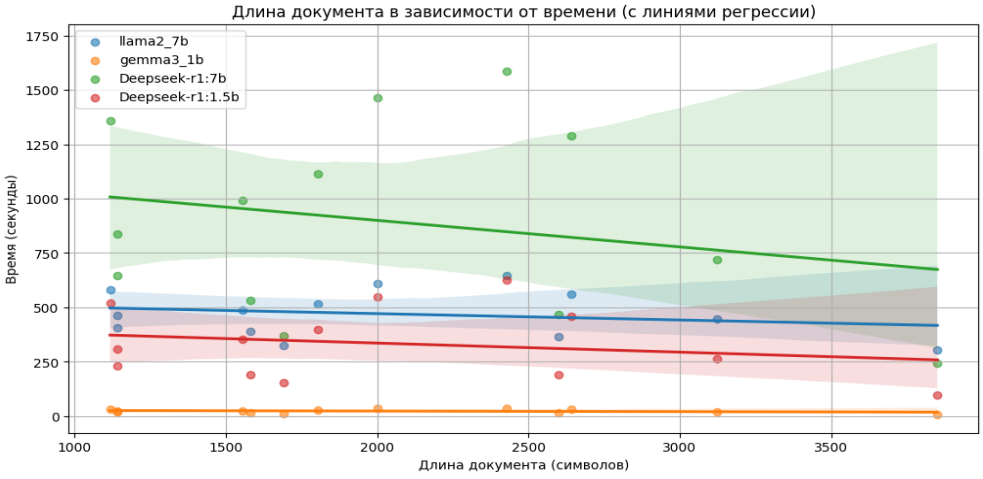

Рис. 4. Зависимость времени выполнения от длины документа

Наиболее выраженная чувствительность наблюдается у Deepseek-r1:7b, где увеличение длины документа приводит к практически линейному росту времени обработки. В противоположность этому, *Gemma* показывает удивительную стабильность работы,

демонстрируя наименьшую зависимость от объема текста. В будущих работах планируется систематически исследовать корреляцию времени обработки с: (1) длиной текста, (2) параметризацией модели и (3) характеристиками аппаратной платформы.

Заключение. В данной статье разработан интеллектуальный чат-бот с комплексным анализом его архитектуры и возможностей обработки текста, а также проведена оценка производительности локальных языковых моделей. Основное внимание уделялось анализу эффективности этих моделей при выполнении традиционных задач, таких как классификация текстов с использованием метода "одиночного примера" (One-Shot). Несмотря на значительный прогресс в аналитических возможностях этих моделей, они не смогли корректно классифицировать все документы. Примечательно, что увеличение размера модели – что предположительно должно повышать точность – не привело к значительному улучшению классификации, а в некоторых случаях даже вызвало снижение производительности. В этом контексте модель *Gemma* выделилась как оптимальный вариант с точки зрения баланса между размером модели и точностью классификации. Исследование также проанализировало взаимосвязь между временем выполнения и длиной текста, выявив положительную корреляцию между этими параметрами. В будущих исследованиях планируется более глубокий анализ этого аспекта, включая влияние других факторов, таких как архитектура модели и возможности тонкой настройки (Fine-Tuning) для улучшения производительности. Также будет проведена проверка достоверности экспертных меток в эталонных наборах данных для задач классификации текстов на естественных языках.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, B. Zoph [et al.]*. OpenAI. GPT-4 Technical Report, 2024.
2. *Baktash J.A., Dawodi M.* Gpt-4: A Review on Advancements and Opportunities in Natural Language Processing, 2023.
3. *Allahyari M., Pouriyeh S., Assefi M., Safaei S., Trippe E.D., Gutierrez J.B., Kochut K.* A brief survey of text mining: Classification, clustering and extraction techniques, 2017.
4. *Roumeliotis K.I., Tselikas N.D., Nasiopoulos D.K.* Llama 2: Early Adopters' Utilization of Meta's New Open-Source Pretrained Model, 2023.
5. *DeepSeek-AI., Guo D., Yang D., Zhang H., Song J., Zhang Z. [et al.]*. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025.
6. *Zhang C., Deng Y., Lin X., Wang B., Ng D., Ye H., Li X., Xiao Y., Mo Z., Zhang Q., Bing L.* 100 Days After DeepSeek-R1: A Survey on Replication Studies and More Directions for Reasoning Language Models, 2025.
7. *Team G., Mesnard T., Hardin C., Dadashi R., Bhupatiraju S., Kenealy K. [et al.]*. Gemma: Open Models Based on Gemini Research and Technology, 2024.
8. *Reimers N., Gurevych I.* Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, *arXiv preprint arXiv,* 2019, Vol. abs/1908.10084.
9. *Mansour A., Mohammad J., Kravchenko Y., Kravchenko D., Silega N.* Harnessing Key Phrases in Constructing a Concept-Based Semantic Representation of Text Using Clustering Techniques, *International Workshop on Artificial Intelligence and Pattern Recognition*. Springer, 2023, pp. 190-201.
10. *Mansour A., Mohammad J., Kravchenko Y.* Text Vectorization Method Based on Concept Mining Using Clustering Techniques, *2022 VI International Conference on Information Technologies in Engineering Education (Inforino)*. IEEE, 2022, pp. 1-10.
11. *Mansour A.M., Mohammad J.H., Kravchenko Y.A.* Text vectorization using data mining methods, *Izvestia SFedU. Technical science,* 2021, No. 2.
12. *Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł., Polosukhin I.* Attention is all you need, *Advances in neural information processing systems,* 2017, Vol. 30.
13. *Franceschelli G., Musolesi M.* Creative Beam Search: LLM-as-a-Judge For Improving Response Generation, 2024.
14. *Pryzant R., Iter D., Li J., Lee Y.T., Zhu C., Zeng M.* Automatic Prompt Optimization with "Gradient Descent" and Beam Search, 2023.
15. *Adeshina A.A.* Building Python Web APIs with FastAPI: A fast-paced guide to building high-performance, robust web APIs with very little boilerplate code. Packt Publishing Ltd, 2022.

16. *Giray L.* Prompt engineering with ChatGPT: a guide for academic writers, *Annals of biomedical engineering,* 2023, Vol. 51, No. 12. pp. 2629-2633.
17. *Marvin G., Hellen N., Jjingo D., Nakatumba-Nabende J.* Prompt Engineering in Large Language Models, *Data Intelligence and Cognitive Informatics: Algorithms for Intelligent Systems,* eds. I.J. Jacob, S. Piramuthu, P. Falkowski-Gilski. Singapore: Springer Nature Singapore, 2024, pp. 387-402. ISBN 978-981-9979-99-8.
18. *Mahmoud Bsharat S., Myrzakhan A., Shen Z.* Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4, *arXiv e-prints,* 2023, pp. arXiv-2312.
19. *Mann B., Ryder N., Subbiah M., Kaplan J., Dhariwal P., Neelakantan A., Shyam P., Sastry G., Askell A., Agarwal S.* Language models are few-shot learners, *arXiv preprint arXiv:2005.14165,* 2020, Vol. 1, pp. 3.
20. *Sabbah T., Selamat A., Selamat M.H., Al-Anzi F.S., Viedma E.H., Krejcar O., Fujita H.* Modified frequency-based term weighting schemes for text classification, *Applied Soft Computing,* 2017, Vol. 58, pp. 193-206.

Мансур Али Махмуд – Южный федеральный университет; e-mail: mansur@sfedu.com; г. Таганрог, Россия; тел.: +79880158697; кафедра систем автоматизированного проектирования им. В.М. Курейчика; программист.

Мохаммад Жуман Хуссайн – Южный федеральный университет; e-mail: zmohammad@sfedu.ru; г. Таганрог, Россия; тел.: +79880158697; кафедра систем автоматизированного проектирования им. В.М. Курейчика; соискатель.

Кравченко Юрий Алексеевич – Южный федеральный университет; e-mail: yakravchenko@sfedu.ru; г. Таганрог, Россия; тел.: +79289080151; кафедра систем автоматизированного проектирования им. В.М. Курейчика; профессор.

Mansour Ali Mahmoud – Southern Federal University; e-mail: mansur@sfedu.com; Taganrog, Russia; phone: +79880158697; the Department of Computer Aided Design named after V.M. Kureichik; programmer.

Mohammad Juman Hussain – Southern Federal University; e-mail: zmohammad@sfedu.ru; Taganrog, Russia; phone: +79880158697; the Department of Computer Aided Design named after V.M. Kureichik; applicant.

Kravchenko Yury Alekseevich – Southern Federal University; e-mail: yakravchenko@sfedu.ru; Taganrog, Russia; phone: +79289080151; the Department of Computer Aided Design named after V.M. Kureichi; professor.

УДК 621.396.969 DOI 10.18522/2311-3103-2025-3-171-180

В.А. Деркачев

КЛАССИФИКАЦИЯ РАДИОЛОКАЦИОННЫХ ИЗОБРАЖЕНИЙ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ МУЛЬТИРОТОРНОГО ТИПА С ПРИМЕНЕНИЕМ АЛГОРИТМА YOLO11

Рассматривается классификатор радиолокационных изображений беспилотных летательных аппаратов, основанный на нейронной сети, построенной на алгоритме YOLO 11 версии. Решение задачи обнаружения и классификации беспилотных летательных аппаратов стало одной из приоритетных задач в настоящее время. Увеличение числа модификаций беспилотных летательных аппаратов сильно усложняет применение статистических методов классификации, что требует применения новых подходов в решении задачи классификации. Развитие нейросетевых методов, одновременно с увеличением производительности вычислителей для обучения, с одной стороны, и встраиваемых решений, с другой, позволяет осуществлять классификацию летательных аппаратов с применением радиолокационных изображений в реальном масштабе времени. Применение алгоритма YOLO11 позволяет, помимо определения класса цели, осуществить оценку дальности до наблюдаемого объекта. Использование радиолокационных изображений оправданно в связи с тем, что визуальное наблюдение не всегда является возможным, из-за сложных погодных условий и темного времени суток. Для обучения нейронной сети предполагается использовать набор радиолокационных изображений, полученный с применением авторской модели генерации данных с произвольной конфигурацией беспилотных летательных аппаратов. Проведено обучение