Russian Technological Journal.
2025;13(2):7–17

ISSN 2782-3210 (Print)
ISSN 2500-316X (Online)

УДК 004.622
https://doi.org/10.32362/2500-316X-2025-13-2-7-17
EDN OQUHWL

НАУЧНАЯ СТАТЬЯ

Сбор и анализ датасета для задачи автоматической генерации сообщений коммитов

И.А. Косьяненко [@], **Р.Г. Болбаков**

МИРЭА – Российский технологический университет, Москва, 119454 Россия
[@] *Автор для переписки, e-mail: kosyanenko.edu@gmail.com*

Резюме

Цели. Для управления процессом разработки современного программного обеспечения нередко применяются системы контроля версий, которые позволяют фиксировать изменения в программном коде и передавать контекст этих изменений при помощи сообщений коммитов. Релевантное и качественное описание внесенных изменений при помощи таких сообщений требует от разработчика высокой компетенции и времени, но современные методы машинного обучения позволяют решать эту задачу автоматически. Целью работы является статистический и сравнительный анализ собранной выборки данных с наборами изменений в программном коде и их описаниями на естественном языке.

Методы. В исследовании использован комплексный подход, включающий сбор данных с популярных репозиториев на GitHub, предварительную обработку и фильтрацию данных, а также статистический анализ и метод обработки естественного языка (векторизация текста). Для оценки семантической близости между первым предложением и полным текстом сообщений коммитов было использовано косинусное сходство.

Результаты. Проведено исследование структуры и качества сообщений коммитов, включающее сбор данных из репозиториев GitHub и их предварительную очистку. Осуществлена векторизация текста сообщений коммитов и оценка семантической близости между первыми предложениями и полными текстами сообщений с использованием косинусного сходства. Выполнен сравнительный анализ качества сообщений в собранном датасете и в нескольких аналогичных наборах данных с помощью классификации при помощи модели CodeBERT.

Выводы. Проведенный анализ выявил низкий уровень косинусного сходства между первыми предложениями и полными текстами сообщений коммитов (0.0969), что свидетельствует о слабой семантической связи между ними и опровергает гипотезу о том, что первые предложения выступают в качестве обобщения содержания сообщений. Процентная доля пустых сообщений в собранном наборе данных составила лишь 0.0007%, что существенно ниже ожидаемого значения и указывает на высокое качество собранных данных. Классификационный анализ показал, что доля сообщений, отнесенных к категории «плохих», в собранном датасете составляет 16.82%, что значительно ниже аналогичных показателей в других сопоставимых наборах данных, где этот процент варьируется от 34.75% до 54.26%. Данный факт подчеркивает высокое качество собранного набора данных и его адекватность для дальнейшего применения в системах автоматической генерации сообщений коммитов.

Ключевые слова: генерация сообщений коммитов, системы контроля версий, описание изменений в программном коде, косинусное сходство, фильтрация данных, векторизация текста, датасет, машинное обучение

• **Поступила:** 01.03.2024 • **Доработана:** 22.07.2024 • **Принята к опубликованию:** 06.02.2025

Для цитирования: Косьяненко И.А., Болбаков Р.Г. Сбор и анализ датасета для задачи автоматической генерации сообщений коммитов. *Russian Technological Journal*. 2025;13(2):7–17. https://doi.org/10.32362/2500-316X-2025-13-2-7-17, https://elibrary.ru/OQUHWL

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

RESEARCH ARTICLE

Dataset collection for automatic generation of commit messages

Ivan A. Kosyanenko [@], **Roman G. Bolbakov**

MIREA – Russian Technological University, Moscow, 119454 Russia
[@] *Corresponding author, e-mail: kosyanenko.edu@gmail.com*

Abstract

Objectives. In contemporary software development practice, version control systems are often used to manage the development process. Such systems allow developers to track changes in the codebase and convey the context of these changes through commit messages. The use of such messages to provide relevant and high-quality descriptions of the changes generally requires a high level of competence and time commitment from the developer. However, modern machine learning methods can enable the automation of this task. Therefore, the work sets out to provide a statistical and comparative analysis of the collected data sample with sets of changes in the program code and their descriptions in natural language.

Methods. In this study, a comprehensive approach was used, including data collection from popular GitHub repositories, preliminary data processing and filtering, as well as statistical analysis and natural language processing method (text vectorization). Cosine similarity was used as a means of assessing the semantic proximity between the first sentence and the full text of commit messages.

Results. A comprehensive study of the structure and quality of commit messages encompassed data collection from GitHub repositories and preliminary data cleansing. The research involved text vectorization of commit messages and evaluation of semantic similarity between the first sentences and full texts of messages using cosine similarity. The comparative analysis of message quality in the collected dataset and several analogous datasets used classification based on the CodeBERT model.

Conclusions. The analysis revealed a low level of cosine similarity (0.0969) between the first sentences and full texts of commit messages, indicating a weak semantic relationship between them and refuting the hypothesis that first sentences serve as summaries of message content. The low proportion of empty messages in the collected dataset at 0.0007% was significantly lower than expected, indicating high-quality data collection. The results of classification analysis showed that the proportion of messages categorized as “poor” in the collected dataset was 16.82%, substantially lower than comparable figures in other datasets, where this percentage ranged from 34.75% to 54.26%. This fact underscores the high quality of the collected dataset and its suitability for further application in automatic commit message generation systems.

Keywords: commit message generation, version control systems, description of changes in software code, cosine similarity, data filtering, text vectorization, dataset, machine learning

• **Submitted:** 01.03.2024 • **Revised:** 22.07.2024 • **Accepted:** 06.02.2025

For citation: Kosyanenko I.A., Bolbakov R.G. Dataset collection for automatic generation of commit messages. *Russian Technological Journal.* 2025;13(2):7–17. https://doi.org/10.32362/2500-316X-2025-13-2-7-17, https://elibrary.ru/OQUHWL

Financial disclosure: The authors have no financial or proprietary interest in any material or method mentioned.

The authors declare no conflicts of interest.

Глоссарий

Датасет – коллекция (набор) данных, которая обычно обрабатывается и анализируется как единое целое. В контексте машинного обучения датасеты обычно содержат примеры, используемые для обучения модели.

Коммит – в контексте систем контроля версий является записью о конкретном наборе изменений в коде. Каждый коммит обычно сопровождается сообщением, которое описывает внесенные изменения.

ВВЕДЕНИЕ

Системы контроля версий играют ключевую роль в разработке современного программного обеспечения. Они позволяют разработчикам отслеживать и управлять изменениями в коде, обеспечивая эффективное сотрудничество и повышение качества продукта. Одними из основных элементов систем контроля версий являются коммиты – записи о каждом значимом изменении, сделанном в кодовой базе.

Сообщения коммитов играют важную роль, поскольку они предоставляют контекст для каждого изменения, помогают понять, что было сделано и почему. Так, например, при помощи сообщений коммитов предлагается обнаруживать уязвимости в программном продукте[1].

Однако написание эффективных сообщений коммитов является непростой задачей, требующей времени и усилий. Хорошее сообщение должно как описывать изменения в программном коде, так и давать пояснение причины внесенного изменения [1], и, по многим рекомендациям, его объем не должен превышать 30 слов (лексем, токенов).

Для автоматической генерации сообщений коммитов было предложено несколько подходов [2]. Тем не менее, исследования [3] показывают, что около 50% сообщений, полученных при помощи инструментов автоматической генерации, оказываются нерелевантными или некорректными.

В настоящей статье внимание фокусируется на сборе и анализе обучающих данных для алгоритма автоматической генерации сообщений коммитов. Обучающий набор данных (датасет) является одним из важных факторов, влияющих на качество и эффективность моделей машинного обучения [4]. Он представляет собой совокупность примеров, которые используются для обучения и тестирования модели, а также для оценки ее обобщающей способности. Качество обучающего набора данных зависит от его размера, разнообразия, репрезентативности, чистоты и релевантности по отношению к задаче машинного обучения. Некачественный датасет может привести к ряду проблем в процессе обучения модели, включая переобучение [5], недообучение, высокое смещение и дисперсию. Это, в свою очередь, может снизить точность и полноту предсказаний модели. Следовательно, формирование и оптимизация обучающего набора данных являются критически важными этапами в процессе машинного обучения, требующими детального анализа, подготовки и очистки данных для обеспечения качества и эффективности модели.

1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ И ОБЗОР СУЩЕСТВУЮЩИХ НАБОРОВ ДАННЫХ

1.1. Важность набора данных в задачах машинного обучения

В 2001 г. исследователи из Microsoft отметили [6], что, несмотря на наличие обширных текстовых корпусов в Интернете, специалисты в области обработки естественного языка продолжали использовать относительно малые наборы данных (до 1 млн слов) для обучения моделей, сосредотачиваясь в основном на оптимизации алгоритмов. В их работе подчеркивается, что различные, в т.ч. и простые, алгоритмы демонстрируют схожую высокую эффективность в задачах устранения неоднозначности языка при наличии достаточного количества данных. В ходе экспериментов четыре алгоритма обучались на данных с контекстным окном в одно слово, постепенно

[1] Wan L. *Automated vulnerability detection system based on commit messages*: магистерская диссертация. Сингапур: Наньянский технологический университет, 2019. 123 с. [Wan L. *Automated vulnerability detection system based on commit messages*: Master's Thesis. Singapore: Nanyang Technological University, 2019. 123 p.]

увеличивая объемы выборок. Результаты исследования показаны на рис. 1.

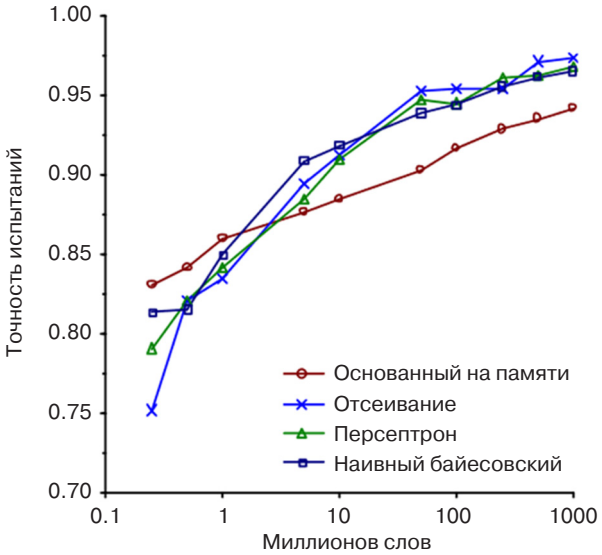

Рис. 1. Результаты экспериментов по увеличению объема обучающего корпуса текста [6]

В экспериментах авторы исследовали:

- алгоритм, основанный на памяти, который сохраняет и использует предыдущую информацию для принятия решений;
- алгоритм отсеивания, применяющий методы отбрасывания ненужных данных или шума для улучшения качества модели;
- персептрон, являющийся простейшей моделью нейронной сети, используемой для бинарной классификации;
- наивный байесовский классификатор, основанный на применении теоремы Байеса с предположением о независимости признаков.

По результатам экспериментов можно заметить, что при увеличении набора данных точность рассматриваемых алгоритмов машинного обучения значительно возрастает. Примечательно, что на малых объемах данных (менее 1 млн слов) алгоритмы показывают различную точность, однако по мере увеличения объема данных различия между ними становятся менее значительными. Авторы отметили, что полученные ими результаты могут навести на мысль о пересмотре компромисса между затратами ресурсов на разработку алгоритмов и на совокупную разработку корпуса текста (набора данных для обучения моделей).

Идея о том, что для сложных задач данные более важны, чем алгоритмы, была популяризирована П. Норвегом и другими исследователями из Google [7]. Экспертная оценка для разметки данных часто сложна и медленна из-за несогласованности оценок. Авторы статьи заключают, что полезные семантические связи могут быть получены статистическими методами, и для обработки текстов следует использовать неразмеченные данные большого объема.

В задаче автоматической генерации сообщений коммитов обучающая выборка играет ключевую роль, т.к. качество и объем данных напрямую влияют на способность модели интерпретировать и описывать изменения в коде. Большой и разнообразный набор данных, содержащий примеры коммитов из различных проектов и написанных разными разработчиками, может улучшить способность модели к обобщению и адаптации к новым данным.

Таким образом, при решении задачи автоматической генерации сообщений коммитов необходимо уделить должное внимание сбору и подготовке обучающего набора данных, что повысит эффективность и точность модели, а также ее практическую ценность для разработчиков.

1.2. Обзор существующих наборов данных

Перейдем к обзору существующих наборов данных с сообщениями коммитов [8]:

- **CommitGen** – один из первых датасетов [9] коммитов, собранный на основе тысячи самых популярных проектов на языке Java. Из датасета исключены коммиты, не несущие смысловой нагрузки для генерации сообщений (например, rollback и merge). Также применен фильтр Verb-Direct Object (V-DO), основанный на морфологическом анализе, показавшем, что сообщения часто начинаются с глагола, за которым идет прямое дополнение [10]. В результате датасет содержит 537000 помеченных diff-файлов.
- **NNGen** [3] – улучшение CommitGen путем удаления «шумных» данных (16% от исходного набора).
- **CoDiSum** [11] – основан на CommitGen, ограничен .java файлами и очищен от специальных символов.
- **PtrGen** [12] – включает 32663 пары <код: сообщение> из 2081 высокооцененных Java-проектов, с заменой специальных символов токенами.
- **MultiLang** [13] мультиязыковой набор из трех популярных репозиториев для Python, Java, JavaScript и C++.
- **ATOM** [14] – содержит 197968 записей после фильтрации шумных сообщений и коммитов без изменений в коде из 56 наиболее популярных Java-проектов.
- **CommitChronicle** [15] – на июль 2024 г. является самым объемным набором данных с коммитами, содержит 10.7 млн коммитов для 20 разных языков программирования.

Ключевая проблема многих датасетов – сосредоточенность на Java, что ограничивает применимость потенциального инструмента для автоматического создания сообщений коммитов. Необходимо собрать набор данных, который позволит инструменту генерации работать с как можно большим количеством языков программирования и содержать информацию о связях между изменениями в коде и сообщениями к ним.

Каждый набор данных имеет свои особенности и ограничения, и выбор подходящего набора зависит от конкретных целей и требований исследования.

В исследовании [9] авторы отмечают, что примерно 14% сообщений коммитов являются пустыми, и данный факт служит одним из обоснований необходимости инструмента генерации сообщений коммитов. Проверка этой информации является одним из исследовательских вопросов данной работы.

B1: какой процент сообщений коммитов из выборки является пустым?

Гипотеза B1: примерно 14% сообщений коммитов в исследуемом датасете являются пустыми.

Результат проверки гипотезы представлен в пункте 3.2.

2. МЕТОДОЛОГИЧЕСКАЯ ОСНОВА

2.1. Планирование сбора датасета

Для преодоления ограничений на количество языков программирования в наборах данных, упомянутых ранее, перед процессом сбора данных необходимо определить перечень актуальных языков. Формируемый набор данных должен содержать изменения в программном коде, написанном на языках из выбранного множества. Таким образом, обученная на таком наборе данных модель сможет обобщать синтаксические и семантические особенности выбранных языков и на их основании синтезировать описания изменений на естественном языке. Оценить актуальность программных языков можно на основании статистических исследований[2].

Выбор источников данных напрямую влияет на качество финального датасета, поэтому отбор репозиториев-доноров происходил в ручном режиме. Источники выбирались на основании популярности репозитория, мерой популярности служили оценки одобрения от пользователей на платформе GitHub[3]. Значительная часть популярных репозиториев состояла из обучающих материалов: такие репозитории не попали в список доноров. Финальный список репозиториев размещен в онлайн-приложении к статье[4].

Исходя из специфики задачи, было принято решение извлекать следующие признаки из каждого репозитория-донора (табл. 1).

Таблица 1. Описание признаков

Признак	Описание
hash	Хеш коммита (hash). Генерируется системой контроля версий и служит идентификатором изменения
author	Идентификатор автора сообщения. Может потребоваться для формирования более обобщенной выборки, где не будет преобладать стиль конкретного автора
commiter_date	Дата и время коммита
timezone	Часовой пояс автора
parents	Список родительских коммитов
message	Сообщение коммита. Описание изменений на естественном языке
language	Язык программирования (language). Основной язык репозитория
changes	Список внесенных в коммите изменений

Два признака – changes (внесенные изменения) и message (описание на естественном языке) – непосредственно являются данными, подаваемыми на вход модели генерации сообщений коммитов. Остальные признаки носят служебный характер и будут применены на этапе фильтрации собранных данных.

2.2. Методы и процедура очистки датасета

Для того, чтобы набор данных был пригоден к применению в алгоритмах машинного обучения, его необходимо подготовить. Под подготовкой понимается фильтрация данных, их очистка, и, если требуется, разработка новых признаков. От указанного выше этапа будет зависеть качество набора данных, и, следовательно, моделей, которые на нем обучаются.

В самых первых датасетах сообщений коммитов содержалось большое количество сообщений, сгенерированных при помощи «ботов» – утилит,

[2] Most used programming languages among developers worldwide as of 2023. Statista. https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/. Дата обращения 10.10.2023. / Accessed October 10, 2023.

[3] https://github.com/. Дата обращения 10.10.2023. / Accessed October 10, 2023.

[4] Онлайн-приложение к статье. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Дата обращения 06.02.2025. [Online appendix to the article. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Accessed February 06, 2025.]

фиксирующих изменения в репозитории программного кода и добавляющих к ним тривиальные описания на естественном языке [9]. Такие сообщения могли описывать внесенные изменения (отвечать на вопрос «что было изменено?»), но не предоставляли достаточно контекста внесения этих изменений (иными словами, не отвечали на вопрос «почему изменения были внесены?») [1], и, следовательно, в качественном датасете количество таких сообщений должно быть сведено к минимуму.

Говоря формально, одной из необходимых процедур очистки собранных данных должна являться классификация сообщений на «естественные» и «сгенерированные» и фильтрация последних. Решить эту задачу можно при помощи метода сопоставления шаблонов [16] – в имени автора сообщения, сгенерированного автоматически, присутствует токен «[bot]». Таким образом, в большинстве случаев запись, в признаке «*author*» которой встречается такой шаблон, можно классифицировать как сгенерированную и удалить в рамках фильтрации.

Помимо сгенерированных сообщений необходимо также подвергнуть фильтрации сообщения тривиальные. Согласно таксономии сообщений коммитов [1], к таковым можно отнести сообщения, генерируемые самой системой контроля версий git (сообщения о слиянии веток в репозитории) и дублирующие сообщения – сообщения, описывающие содержимое изменений, которое легко вывести из различий в коде («Update readme.md», «Add <file name>»). Для классификации и фильтрации таких сообщений можно воспользоваться регулярными выражениями [17], составив шаблоны тривиальных сообщений при помощи синтаксиса регулярных выражений.

Одним из популярных фильтров для наборов данных для задачи автоматической генерации сообщений коммитов является фильтр V-DO, появившийся из наблюдения, что примерно половина сообщений коммитов соответствует структуре «глагол и следующий за ним объект» [10]. Формально, можно описать фильтр V-DO в виде функции:

$$f(m)=\begin{cases}1,\ \exists i:(w_i-\text{глагол и } w_{i+1}-\text{объект}),\\ 0,\ \text{в любом другом случае.}\end{cases} \quad (1)$$

где m – это сообщение коммита, w_i – i-е слово в сообщении m, а $f(m)$ – результат фильтрации. Если $f(m)$ = 1, то сообщение коммита проходит через фильтр, в противном случае – нет. Предположим, у нас есть сообщение коммита «*Minor changes to the database schema*». Фильтр V-DO не пропустит это сообщение, т.к. оно не начинается с глагола, за которым следует прямое дополнение (объект). Однако сообщение коммита «*Modified database schema*» будет пропущено фильтром V-DO, т.к. оно соответствует шаблону.

Одним из важных фильтров является ограничение на количество токенов (длину) в сообщении коммита [8]. Информативное и релевантное сообщение не должно быть слишком коротким или слишком длинным. Большинство исследователей [10–12] фильтровали набор данных по максимальной длине в 30 токенов, хотя более поздние работы [15] сфокусировались на диапазоне от 5 до 600 токенов в сообщении. В большинстве случаев 5 токенов недостаточно для релевантного описания внесенных изменений, а оставление в наборе длинных сообщений (больше 600 токенов) может повлечь за собой увеличение занимаемого набором дискового пространства и вычислительной сложности для его обработки.

Некоторые исследования предлагают оставлять только первое предложение от сообщения коммита [13], объясняя это тем, что первое предложение часто является обобщением всего сообщения. Такой фильтр позволил бы снизить объем дискового пространства, занимаемый набором данных, но вместе с тем потенциально принес бы потери важной семантической информации. Проверка этой гипотезы является еще одним из исследовательских вопросов работы.

B2: являются ли первые предложения в сообщениях коммитов обобщением всего сообщения?

Гипотеза B2: первые предложения в сообщениях коммитов являются обобщением всего сообщения.

Методы проверки данной гипотезы изложены в пункте 1.3, результат проверки представлен в пункте 2.3.

2.3. Методы проверки семантической близости двух текстовых последовательностей

В сфере обработки естественного языка для проверки семантической близости двух текстовых последовательностей применяется мера косинусного сходства [18, 19]. Формально, косинусное сходство отражает косинус угла между векторами предгильбертового пространства и может быть выражено в виде формулы:

$$similarity=\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\cdot\|\mathbf{b}\|}=\frac{\sum_{i=1}^{n}a_i b_i}{\sqrt{\sum_{i=1}^{n}a_i^2}\sqrt{\sum_{i=1}^{n}b_i^2}}, \quad (2)$$

где a_i и b_i – это соответствующие элементы векторов **a** и **b**. Диапазон меры – от 0 до 1; если мера равна 0, то вектора двух последовательностей ортогональны и далеки друг от друга семантически. Если же мера равна 1, то две текстовые последовательности имеют близкое семантическое значение.

Стоит также упомянуть методы отображения символьных последовательностей в векторное (предгильбертово) пространство. Для вычисления меры на собранных данных и проверки гипотезы B2 сообщения коммитов и их первые предложения должны быть преобразованы в численные векторы одинаковой размерности. Такой процесс называется векторизацией текста – это процесс преобразования текста в числовые векторы, которые не только могут быть использованы машинными алгоритмами, но и отражают семантику текста благодаря принципу дистрибутивной семантики.

Одним из методов такой векторизации является векторизация хешированием [20]. Этот инструмент применяет хеш-функцию к словам и преобразует их в числовые индексы в векторном пространстве, сохраняя при этом семантические связи между словами.

3. РЕЗУЛЬТАТЫ

3.1. Процесс сбора данных

Сбор данных велся с 15.10.2023 по 25.11.2023 г. Источником данных послужил сервис GitHub. Необработанный набор данных содержит 3141212 записей и занимает 73 ГБ дискового пространства. Распределение записей по языкам программирования представлено в табл. 2.

Таблица 2. Распределение записей по языкам программирования

Язык	Количество записей в собранном датасете
C	932003
Ruby	253331
TypeScript	251127
C++	251125
Rust	211660
Python	209374
Java	141024
Go	140211
JavaScript	121610
C#	107960
Scala	86929
Dart	86532
Kotlin	81183
Lua	61622
PHP	61399
Groovy	50647
Shell	45687
R	22190
Swift	14396
Objective-C	11200

Важно отметить, что количество записей по различным языкам программирования распределилось неравномерно, и, если того потребует задача, целесообразно было бы сделать равномерную выборку из данного набора.

3.2. Анализ и очистка собранных данных

Всего в собранном наборе данных 22 записи с пустыми сообщениями, что составляет примерно 0.0007% от общего числа всех записей. Записи с пустыми сообщениями были удалены из набора. Важно уточнить, что данные собирались из самых популярных репозиториев, которые часто принадлежат технологическим компаниям, поэтому этот показатель может не отражать реальную статистику по пустым сообщениям.

Гипотеза B1 «*примерно 14% сообщений коммитов в исследуемом датасете являются пустыми*» не подтвердилась.

В собранном наборе данных фильтру V-DO не соответствуют 2952077 сообщений, т.е. приблизительно 94%. Его применение привело бы к значительному сокращению объема набора данных, а вместе с ним и полезных зависимостей между изменениями в коде и естественным языком, поэтому было принято решение не применять данный фильтр.

В неотфильтрованном собранном наборе данных максимальная длина сообщения коммита – 68529. И это явный статистический выброс, который может негативно повлиять на работу алгоритма [21].

После удаления пустых сообщений были рассчитаны следующие статистические показатели по количеству токенов в сообщениях коммитов (табл. 3):

Таблица 3. Статистические показатели по количеству токенов в сообщениях коммитов

Статистические показатели	Значение
Количество записей (count)	3141186
Среднее (mean)	52.19
Стандартное отклонение (std)	129.61
Минимум (min)	1
25-й процентиль	10
Медиана (50-й процентиль)	42
75-й процентиль	174
Максимум (max)	68529

Стандартное отклонение (примерно 129.61) довольно велико, что указывает на значительное разнообразие в длине сообщений. Максимальное значение (68529) гораздо больше, чем 75-й процентиль (174), что указывает на наличие выбросов с очень большим количеством токенов.

Чтобы уменьшить влияние выбросов, было принято решение оставить в наборе только те записи, количество токенов в которых расположено в диапазоне от 5 до 600 (включительно). Такой фильтр сократил количество записей в наборе до 2761945 или на 12.07%. На рис. 2 представлена одномерная диаграмма типа «ящик с усами» (box plot), которая показывает распределение количества токенов в сообщениях коммитов в собранном наборе данных.

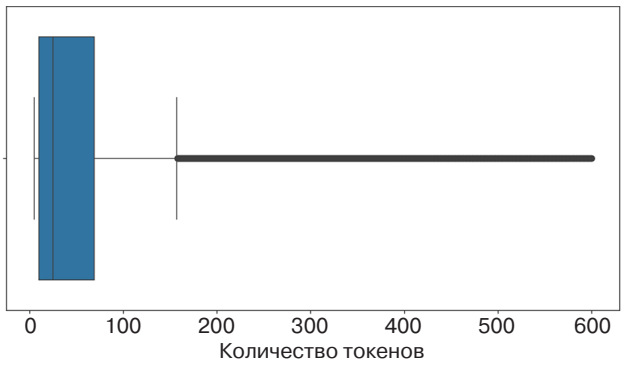

Рис. 2. Распределение количества токенов в сообщениях коммитов

Основная часть данных находится в пределах от 5 до 100 токенов, однако также видны значительные выбросы, указывающие на наличие большого числа сообщений с повышенным количеством токенов.

Для фильтрации тривиальных и сгенерированных сообщений были применены регулярные выражения. В качестве шаблонов использовались ключевые слова из утилит CI-CD, например, «*bump version to*». Помимо этого, были удалены символы, не входящие в ASCII [9] и записи, в признаке «*author*» которых встречался токен «[bot]». В результате применения данного фильтра было выявлено 185540 сообщений, сгенерированных автоматическими утилитами. После применения фильтра количество записей в наборе составило 2576405.

3.3. Оценка семантической близости первого предложения и целого сообщения коммита

После проведения фильтрации была проверена гипотеза [13] о том, что первое предложение в сообщении коммитов является обобщением всего сообщения.

Для проведения экспериментов по проверке этой гипотезы был создан новый признак (колонка в наборе данных), затем первые предложения и сообщения целиком были отображены в численные векторы при помощи утилиты HashingVectorizer из библиотеки scikit-learn с размерностью 1048576. Таким образом, первые предложения и сообщения целиком были преобразованы в векторы, состоящие из 1048576 чисел каждый. Программный код реализации эксперимента доступен в онлайн-приложении к статье[5].

Среднее значение косинусного сходства по всем сообщениям коммитов составило 0.0969. Это значение отражает степень семантической близости между первым предложением и полным текстом сообщений коммитов. Относительно низкое значение среднего косинусного сходства может указывать на то, что первое предложение часто содержит уникальную информацию, которая не полностью повторяется в остальной части сообщения. В рамках данного исследования было принято решение не сокращать объем сообщений коммитов в наборе данных до первого предложения.

Гипотеза В2 «*первые предложения в сообщениях коммитов являются обобщением всего сообщения*» не подтвердилась.

Важно отметить, что при процедуре определения косинусного сходства измеряется только угол между векторами, а не их длина. Это означает, что косинусное сходство не учитывает объем информации, содержащейся в сообщениях коммитов. Для более глубокого анализа структуры сообщений коммитов могут потребоваться более сложные методы.

3.4. Характеристики финального датасета и его сравнение с аналогами

Очищенный датасет доступен в онлайн-приложении к статье[5]. В результате фильтрации количество записей в датасете составило 2576405. Датасет содержит пары изменений для 20 языков программирования, тем самым расширяя сферу применения потенциальной модели генерации сообщений коммитов, которая будет обучаться на данном наборе. Из датасета на этапе фильтрации были удалены автоматически сгенерированные, тривиальные и пустые сообщения.

Для валидации качества датасета было принято решение провести сравнительный анализ собранного набора данных и наборов, упомянутых ранее. Для выполнения сравнительного анализа качества сообщений коммитов в предлагаемом наборе данных была использована методология классификации с применением предобученной нейронной сети «*commit-message-quality-codebert*[6]». Эта модель, основанная

[5] Онлайн-приложение к статье. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Дата обращения 06.02.2025. [Online appendix to the article. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Accessed February 06, 2025.]

[6] Нейронная сеть-классификатор сообщений коммитов. https://huggingface.co/saridormi/commit-message-quality-codebert. Дата обращения 06.02.2025. [Neural network classifier of messages to the commits. https://huggingface.co/saridormi/commit-message-quality-codebert. Accessed February 06, 2025.]

Таблица 4. Результаты сравнительного анализа датасетов

Название датасета	Всего записей	Классифицированы как «хорошие»	Классифицированы как «плохие»	Процент «плохих» записей
Собранный датасет	2576405	2143000	433405	16.82
NNGen	27144	12415	14729	54.26
CoDiSum	90661	56305	34356	37.90
PtrGen	32663	16826	15837	48.49
MultiLang	126928	82819	44109	34.75

на архитектуре CodeBERT [22], была дообучена [23] для задачи классификации качества сообщений коммитов на основании упомянутой ранее таксономии сообщений [1]. «Плохим» по данной таксономии является сообщение, которое не отвечает на вопросы «Почему изменения были внесены?» и «Что изменилось?», т.е. не передают контекст внесенных изменений.

Методологическая процедура анализа включала предварительную обработку данных, в ходе которой выполнялась нормализация текста (приведение к нижнему регистру) и удаление неинформативных символов. Затем сообщения коммитов из собранного набора данных и из других, доступных для сравнения наборов данных, были пропущены через нейронную сеть для автоматической классификации на основе присвоенных меток «хороший» и «плохой». Результаты классификации по каждому из наборов данных представлены в табл. 4.

Анализ результатов сравнения показывает, что доля «плохих» сообщений в собранном наборе данных составила 16.82%, что является существенно более низким показателем по сравнению с другими наборами данных.

ЗАКЛЮЧЕНИЕ

В ходе исследования собран обширный мультиязычный набор данных (датасет), содержащий изменения в программном коде, их описание на естественном языке, а также дополнительную метаинформацию, важную в контексте фильтрации и очистки набора данных. Очищенный от сгенерированных и тривиальных сообщений датасет размещен на хостинге данных HuggingFace[7].

В рамках исследования были выдвинуты две гипотезы, основанные на ранних работах по тематике исследования. Гипотеза B1 состояла в том, что примерно 14% сообщений коммитов в исследуемом датасете являются пустыми. Однако при анализе собранных данных было выявлено, что всего лишь 0.0007% сообщений коммитов в исследуемой выборке являются пустыми, что значительно меньше предполагаемого значения. Таким образом, гипотеза B1 не нашла подтверждения в ходе исследования.

Гипотеза B2 состояла в том, что первые предложения в сообщениях коммитов являются обобщением всего сообщения. Для проверки этой гипотезы проведена векторизация текста и вычислено косинусное сходство между первым предложением и полным текстом сообщений коммитов. На исследуемой выборке мера косинусного сходства составила 0.0969, что свидетельствует о низкой семантической близости между первым предложением и полным текстом сообщения. Следовательно, гипотеза B2 также не нашла подтверждения в ходе исследования.

Полученные результаты опровергают выдвинутые гипотезы и указывают на то, что пустые сообщения коммитов встречаются крайне редко, а первое предложение не является достаточным обобщением всего сообщения. Эти выводы могут служить основой для дальнейшего изучения структуры и содержания сообщений коммитов, а также для разработки систем автоматической генерации сообщений коммитов.

Разумной представляется идея проведения дальнейших исследований по векторизации файлов diff (представление изменений, генерируемое системой контроля версий). Если diff – это набор добавлений и удалений, то для векторизованного diff должны быть определены операции сложения и вычитания, причем сложение можно интерпретировать как добавление программного кода, а вычитание – как удаление из него фрагментов. Такой потенциальный алгоритм позволил бы решить задачу сведения языков программирования к единой формальной нотации, исходя из предположения о том, что одинаковые изменения программного кода, внесенные в файлы, написанные на различных языках программирования, имели бы близкое косинусное расстояние. Вопрос подлежит дальнейшему изучению.

Стоит также отметить целесообразность выявления сгенерированных сообщений при помощи

[7] Собранный датасет. https://huggingface.co/datasets/Malolmalsky/commit_dataset. Дата обращения 06.02.2025. https://doi.org/10.57967/hf/2216. [Collected dataset. https://huggingface.co/datasets/Malolmalsky/commit_dataset. Accessed February 06, 2025. https://doi.org/10.57967/hf/2216]

нейронных сетей. Такой подход будет гораздо сложнее с вычислительной точки зрения, но при этом должен обеспечить большую точность при классификации сообщений коммитов.

Вклад авторов

И.А. Косьяненко – концептуализация исследования, разработка методологии, сбор и анализ данных, проведение вычислительных экспериментов, подготовка первоначального варианта текста статьи.

Р.Г. Болбаков – научное руководство, проверка и редактирование текста статьи, критический анализ полученных результатов.

Authors' contributions

I.A. Kosyanenko – research conceptualization, methodology development, data collection and analysis, computational experiments, preparation of the original draft of the manuscript.

R.G. Bolbakov – scientific supervision, manuscript review and editing, critical analysis of the obtained results.

СПИСОК ЛИТЕРАТУРЫ / REFERENCES

1. Tian Y., Zhang Y., Stol K., Jiang L., Liu H. What makes a good commit message? *Proceedings of the 44th International Conference on Software Engineerin*g. 2022;44:2389–2401. https://doi.org/10.1145/3510003.3510205
2. Косьяненко И.А., Болбаков Р.Г. Об автоматической генерации сообщений к коммитам в системах контроля версий. *International Journal of Open Information Technologies*. 2022;10(4):55–60.
[Kosyanenko I.A., Bolbakov R.G. About automatic generation of commit messages in version control systems. *International Journal of Open Information Technologies* (*INJOIT*). 2022;10(4):55–60 (in Russ.).]
3. Liu Z., Xia X., Hassan A., Lo D., Xing Z. Neural-machine-translation-based commit message generation: how far are we? In: *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. 2018;33:373–384. https://doi.org/10.1145/3238147.3238190
4. Sun Z., Li L., Liu Y., Du X., Li L. On the importance of building high-quality training datasets for neural code search. In: *Proceedings of the 44th International Conference on Software Engineering*. 2022;44:1609–1620. https://doi.org/10.1145/3510003.3510160
5. Hawkins D.M. The problem of overfitting. *J. Chem. Inf. Comput. Sci*. 2004;44(1):1–12. https://doi.org/10.1021/ci0342472
6. Banko M., Brill E. Scaling to Very Very Large Corpora for Natural Language Disambiguation. In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. 2001;26–33. https://doi.org/10.3115/1073012.1073017
7. Halevy A., Norvig P., Pereira F. The unreasonable effectiveness of data. *IEEE Intell. Syst*. 2009;24(2):8–12. https://doi.org/10.1109/MIS.2009.36
8. Tao W., Wang Y., Shi E., Du L., Han S., Zhang H., Zhang D., Zhang W. On the evaluation of commit message generation models: An experimental study. In: *2021 IEEE International Conference on Software Maintenance and Evolution* (*ICSME*). *IEEE*. 2021;126–136. https://doi.org/10.48550/arXiv.2107.05373
9. Jiang S., McMillan C. Towards automatic generation of short summaries of commits. In: *2017 IEEE/ACM 25th International Conference on Program Comprehension* (*ICPC*). *IEEE*. 2017;320–323. https://doi.org/10.48550/arXiv.1703.09603
10. Мягкова Е.Ю. О «формальной» и «внутренней» грамматике. *Вестник Тверского государственного университета. Серия: Филология*. 2012;24(4):96–102.
[Myagkova E.Yu. To the problem of "formal" and "inner" grammar. *Vestnik Tverskogo gosudarstvennogo universiteta. Seriya: Filologiya = Herald of Tver State University. Series: Philology.* 2012;24(4):96–102 (in Russ.).]
11. Xu S., Yao Y., Xu F., Gu T., Tong H., Lu J. Commit message generation for source code changes. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (*IJCAI-19*). 2019;3975–3981. https://doi.org/10.24963/ijcai.2019/552
12. Liu Q., Liu Z., Zhi H., Fan H., Du B., Qian Y. Generating commit messages from diffs using pointer-generator network. In: *2019 IEEE/ACM 16th International Conference on Mining Software Repositories* (*MSR*). *IEEE*. 2019;299–309. http://doi.org/10.1109/MSR.2019.00056
13. Loyola P., Marrese-Taylor E., Matsuo Y. A neural architecture for generating natural language descriptions from source code changes. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 2017;287–292. https://doi.org/10.18653/v1/P17-2045
14. Liu S., Gao C., Chen S., Yiu L., Liy Y. ATOM: Commit message generation based on abstract syntax tree and hybrid ranking. *IEEE Trans. Software Eng*. 2020;48(5):1800–1817. https://doi.org/10.48550/arXiv.1912.02972
15. Eliseeva A., Sokolov Y., Bogomolov E., Golubev Y., Dig D., Bryskin T. From Commit Message Generation to History-Aware Commit Message Completion. In: *2023 38th IEEE/ACM International Conference on Automated Software Engineering* (*ASE*). *IEEE.* 2023;723–735. https://doi.org/10.48550/arXiv.2308.07655
16. Dey T., Mousavi S., Ponce E. Detecting and characterizing bots that commit code. In: *Proceedings of the 17th international conference on mining software repositories.* 2020;209–219. https://doi.org/10.1145/3379597.3387478
17. Kuchnik M., Smith V., Amvrosiadis G. Validating large language models with ReLM. *Proceedings of Machine Learning and Systems*. 2023;5:457–476. https://doi.org/10.48550/arXiv.2211.15458

18. Haque S., Zachary E. Semantic similarity metrics for evaluating source code summarization. In: *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension.* 2022;36–47. https://doi.org/10.1145/3524610.3527909
19. Rahutomo F., Kitasuka T., Aritsugi M. Semantic cosine similarity. In: *The 7th International Student Conference on Advanced Science and Technology* (*ICAST*). 2012;4(1):1–2.
20. Roshan R., Bhacho I.A., Zai S. Comparative Analysis of TF–IDF and Hashing Vectorizer for Fake News Detection in Sindhi: A Machine Learning and Deep Learning Approach. *Eng. Proc.* 2023;46(1):5. https://doi.org/10.3390/engproc2023046005
21. Aggarwal C.C., Yu P.S. Outlier Detection in High Dimensional Data. In: *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*. 2001;30(2):37–46. http://dx.doi.org/10.1145/376284.375668
22. Feng Z., Guo D., Tang F., et al. CodeBERT: A pre-trained model for programming and natural languages. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. P. 1536–1547. Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.139
23. Qasim R., Bangyal W.H. A fine-tuned BERT-based transfer learning approach for text classification. *J. Healthc. Eng.* 2022;2022:3498123. https://doi.org/10.1155/2022/3498123

Об авторах

Косьяненко Иван Александрович, аспирант, кафедра инструментального и прикладного программного обеспечения, Институт информационных технологий, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: kosyanenko.edu@gmail.com. SPIN-код РИНЦ 2592-5015, https://orcid.org/0009-0009-1804-9412

Болбаков Роман Геннадьевич, к.т.н., доцент, заведующий кафедрой инструментального и прикладного программного обеспечения, Институт информационных технологий, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: bolbakov@mirea.ru. Scopus Author ID 57202836952, SPIN-код РИНЦ 4210-2560, http://orcid.org/0000-0002-4922-7260

About the authors

Ivan A. Kosyanenko, Postgraduate Student, Department of Instrumental and Applied Software, Institute of Information Technologies, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: kosyanenko.edu@gmail.com. RSCI SPIN-code 2592-5015, https://orcid.org/0009-0009-1804-9412

Roman G. Bolbakov, Cand. Sci. (Eng.), Associate Professor, Head of the Department of Instrumental and Applied Software, Institute of Information Technologies, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: bolbakov@mirea.ru. Scopus Author ID 57202836952, RSCI SPIN-code 4210-2560, http://orcid.org/0000-0002-4922-7260