Information systems. Computer sciences. Issues of information security
Информационные системы. Информатика. Проблемы информационной безопасности

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

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.

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.

Glossary

Dataset—collection (set) of data that is usually processed and analyzed as a whole. In the context of machine learning, datasets usually contain examples used to train a model.

Commit (in the context of version control systems)—record of a specific set of changes to code. Each commit is usually accompanied by a message that describes the changes made.

INTRODUCTION

Version control systems, which play a key role in modern software development, allow developers to track and manage changes to code as a means of ensuring effective collaboration and improving product quality. One of the main elements of version control systems are commits, i.e., records of every significant change made to the code base. The important role played by commit messages consists in the context they provide for each change, helping other developers to understand what was done and why. For example, commit messages can be used to detect vulnerabilities in a software product.[1]

However, writing effective commit messages represents a difficult task requiring time and effort. As well as describing changes in program code, a good message should explain the reason for the change [1]; moreover, according to many recommendations, it should not exceed 30 words (lexemes, tokens).

Although several approaches have been proposed for the automatic generation of commit messages [2], studies show that about 50% of messages generated by automatic generation tools turn out to be irrelevant or incorrect [3].

The present work focuses on the collection and analysis of training data for an automatic commit message algorithm. Representing one of the most important factors affecting the quality and efficiency of machine learning models [4], a training dataset comprises a collection of examples that are used to train and test the model, as well as to evaluate its generalization ability. The quality of a training dataset depends on its size, diversity, representativeness, purity, and relevance to the machine learning task. A poor quality dataset can lead to a number of problems during model training, including overtraining [5], undertraining, high bias, and variance. This, in turn, can reduce the accuracy and completeness of the model predictions. Hence, the generation and optimization of the training dataset are critical steps in the machine learning process, requiring detailed analysis, training and data cleaning to ensure the quality and efficiency of the model.

1. THEORETICAL BACKGROUND AND REVIEW OF EXISTING DATASETS

1.1. Importance of dataset in machine learning tasks

In 2001, researchers from Microsoft noted [6] that, despite the availability of extensive text corpora on the Internet, natural language processing experts continued to use relatively small datasets (up to 1 million words) to train models, which were mainly focused on optimizing algorithms. Their work emphasizes that different algorithms, including simple ones, demonstrate similar high performance in the task of eliminating language ambiguity given sufficient data. In the experiments, four algorithms were trained on data with a one-word context window, gradually increasing the sample sizes. The results of the study are shown in Fig. 1.

In the experiments, the authors studied:

- a memory-based algorithm that stores and uses previous information for decision making;
- a winnow algorithm that applies techniques to discard unnecessary data or noise to improve model quality;
- a perceptron, representing the simplest model of a neural network used for binary classification;
- a naive Bayesian classifier based on the application of Bayes' theorem with the assumption of feature independence.

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

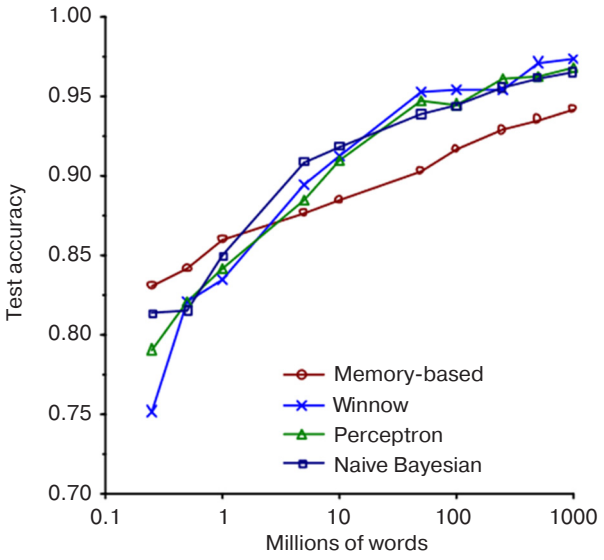

Fig. 1. Experimental results for increasing the size of the training corpus of text [6]

The results of the experiments show that the accuracy of the considered machine learning algorithms increases significantly as the data set increases. Notably, while the algorithms demonstrate varying accuracy on small amounts of data (less than 1 million words), as the amount of data increases, the differences between them become less significant. The authors noted that their results suggest that the trade-off between spending resources on algorithm development and on the aggregate development of the text corpus (dataset for model training) should be reconsidered.

The idea that data is more important than algorithms for complex tasks was popularized by Peter Norvig and other researchers from Google [7]. Conversely, expert judgment for data partitioning is often difficult and slow due to inconsistency in the estimates. The authors of the article conclude that useful semantic relations can be obtained by statistical methods relying on large volumes of unlabeled data used for text processing.

In the task of automatic generation of commit messages, the training sample plays a key role, since the quality and volume of data directly affect the model's ability to interpret and describe changes in the code. A large and diverse dataset containing examples of commits from different projects and written by different developers can improve a model's ability to generalize and adapt to new data.

Thus, when solving the problem of automatic commit message generation, it is necessary to pay due attention to the collection and preparation of the training data set, which will increase the efficiency and accuracy of the model, as well as its practical value for developers.

1.2. Overview of the existing datasets

Let us proceed to review existing datasets with commit messages [8]:

- **CommitGen** is one of the first commit datasets [9] collected from thousands of the most popular Java projects. Commits that do not carry meaningful load for message generation (e.g., rollback and merge) are excluded from the dataset. We also applied the Verb-Direct Object (V-DO) filter based on morphological analysis, which showed that messages often start with a verb followed by a direct complement [10]. As a result, the dataset contains 537000 labeled diff files.
- **NNGen** [3] improves the CommitGen by removing "noisy" data (16% of the original set).
- **CoDiSum** [11] is based on CommitGen, restricted to .java files and cleaned of special characters.
- **PtrGen** [12] includes 32663 <code:message> pairs from 2081 highly-valued Java projects, with special characters replaced by tokens.
- **MultiLang** [13] is a multi-language set of three popular repositories for Python, Java, JavaScript, and C++.
- **ATOM** [14] contains 197968 records after filtering noisy messages and commits without code changes from 56 most popular Java projects.
- **CommitChronicle** [15] is the largest commit dataset (as of July 2024), containing 10.7 mln commits for 20 different programming languages.

A key problem with many datasets is the focus on Java, which limits the applicability of a potential tool for automatically generating commit messages. To solve this problem, a dataset should be assembled that enables the generation tool to work with as many programming languages as possible and contain information about the relationships between code changes and the messages to them.

Each dataset has its own characteristics and limitations, while the selection of an appropriate dataset depends on the specific objectives and requirements of the study.

In the study [9], the authors note that approximately 14% of commit messages are empty. If true, such a fact serves as one of the justifications for the need of a commit message generation tool. Verifying this information is thus one of the research questions of the present work.

B1: What share of commit messages in the sample is empty?

Hypothesis B1: approximately 14% of the commit messages in the dataset under study are empty messages.

The result of hypothesis testing is presented in Paragraph 3.2.

2. METHODOLOGICAL BASIS

2.1. Dataset collection planning

In order to overcome the limitations on the number of programming languages in the datasets, it is necessary to define a list of relevant languages before the data collection process. The generated dataset should contain changes in program code written in languages from the selected set. Thus, a model trained on such a dataset will be able to generalize syntactic and semantic features of the selected languages and synthesize descriptions of changes in natural language on this basis. The relevance of program languages can be assessed on the basis of statistical studies.[2]

Since the choice of data sources directly affects the quality of the final dataset, the selection of donor repositories was done manually. Sources were selected based on the popularity of the repository as assessed by approval ratings from users on the GitHub[3] platform. However, a significant portion of popular repositories consisted of tutorials, which were excluded from the list of sources. The final list of repositories can be found in the online appendix to this article.[4]

Based on the specifics of the task, it was decided to extract the following features from each donor repository (Table 1):

Table 1. Attribute description

Attribute	Description
hash	Commit hash. It is generated by the version control system and serves as a change identifier
author	The identifier of the author of the message. May be required to form a more generalized sample, where the style of a particular author will not be predominant
commiter_date	Date and time of the commit
timezone	Author's timezone
parents	List of parent commits
message	Commit message. Description of changes in natural language
language	Programming language. The main language of the repository
changes	List of changes made in the commit

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

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

[4] Online appendix to the article. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Accessed February 06, 2025.

Two attributes—changes and message (natural language description)—are directly fed to the input of the commit message generation model. The other attributes, which are of a service nature, will be applied at the stage of filtering the collected data.

2.2. Methods and procedure for cleaning the dataset

In order for a dataset to be suitable for use in machine learning algorithms, it must be prepared. Preparation means filtering the data, cleaning it, and developing new features if required. The quality of the dataset—and consequently of the models that are trained on it—will depend on the above step.

The very first datasets of commit messages contained a large number of messages generated by "bots"—utilities that capture changes in the code repository and add trivial natural language descriptions to them [9]. Such messages could describe the changes made (answering the question "what was changed?"), but fail to provide enough context for making these changes (in other words, did not answer the question "why were the changes made?") [1]. Consequently, the number of such messages should be minimized in a high-quality dataset.

Formally speaking, one of the necessary procedures for cleaning the collected data should be the classification of messages into "natural" and "generated" and filtering of the latter. This task can be solved using the pattern matching method [16]—the author's name of an automatically generated message contains the token "[bot]". Thus, in most cases, a record with such a template in the *author* attribute can be classified as generated and deleted within the filtering process.

In addition to generated messages, trivial messages must also be filtered. According to the taxonomy of commit messages [1], these include messages generated by the git version control system itself (messages about merging branches in the repository) and duplicate messages—messages describing the content of changes that can be easily deduced from differences in the code (Update readme.md, Add <file name>). To classify and filter such messages, we can use regular expressions [17] by composing templates of trivial messages using the regular expression syntax.

One popular filter for datasets for the task of automatically generating commit messages is the V-DO filter, which emerged from the observation that about half of the commit messages correspond to the structure 'verb followed by an object' [10]. We can formally describe the V-DO filter in the form of a function:

$$f(m) = \begin{cases} 1, \exists i : (w_i, \text{verb and } w_{i+1}, \text{object}), \\ 0, \text{ in any other case.} \end{cases} \quad (1)$$

where m is the commit message, w_i is the ith word in the message m, and $f(m)$ is the result of the filtering. If $f(m) = 1$, the commit message passes through the filter, otherwise it does not.

Suppose we have a commit message *Minor changes to the database schema.* The V-DO filter will not pass this message because it does not start with a verb followed by a direct object. However, the commit message *Modified database schema* will be skipped by the V-DO filter because it matches the following pattern.

One important filter is the limit on the number of tokens (length) in a commit message [8]. An informative and relevant message should be neither too short nor too long. Most of the researchers [10–12] filtered the dataset by a maximum length of 30 tokens, although more recent work [15] focused on the range of 5 to 600 tokens in a message. In most cases, five tokens are insufficient to describe the changes made in a relevant way; conversely, leaving excessively long messages (more than 600 tokens) in the set may increase the disk space occupied by the set and the computational complexity to process it.

Some studies suggest leaving only the first sentence from a commit message [13] on the basis that the first sentence is often a generalization of the whole message. Such a filter would reduce the amount of disk space occupied by the dataset, but also potentially lead to the loss of important semantic information. Testing this hypothesis is another of the research questions of the work.

B2: do the first sentences in commit messages summarize the whole message?

Hypothesis B2: the first sentences in commit messages are a summary of the whole message.

The methods for verifying this hypothesis are outlined in paragraph 1.3; the result of the test is presented in paragraph 2.3.

2.3. Methods for checking semantic proximity of two text sequences

In the field of natural language processing, the cosine similarity measure [18, 19] is used to test the semantic proximity of two text sequences. Formally, cosine similarity reflects the cosine of the angle between vectors of the pre-Hilbert space and can be expressed as the formula:

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

where a_i and b_i are the corresponding elements of the vectors **a** and **b**.

The range of the measure is from 0 to 1; if the measure is 0, then the vectors of the two sequences are orthogonal and far apart semantically. If the measure is 1, then the two text sequences have close semantic meaning.

It is also worth mentioning the methods of mapping symbolic sequences into vector (pre-Hilbert) space. In order to compute a measure on the collected data and test hypothesis B2, commit messages and their first sentences must be transformed into numerical vectors of the same dimensionality. Such a text vectorization process involves the conversion of text into numerical vectors that not only can be used by machine algorithms, but also reflect the semantics of the text due to the principle of distributive semantics.

One method of such vectorization is vectorization by hashing [20]. This tool applies a hash function to words and converts them into numeric indices in vector space, while preserving semantic relations between words.

3. RESULTS

3.1. Data collection process

The data was collected during the period from 15/10/2023 to 25/11/2023. The data source was the online GitHub service. The raw dataset contains 3141212 records and occupies 73 Gb of disk space. The distribution of records by programming languages is presented in Table 2.

Table 2. Distribution of the records by programming language

Language	Number of records in the collected dataset
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

It is worth noting that the number of records for the different programming languages was not evenly distributed. If required, it would be appropriate to sample evenly from this set.

3.2. Analyzing and cleaning the collected data

The number of records comprised of empty messages in the collected dataset is 22, which is approximately 0.0007% of the total number of all records. The records with empty messages were removed from the dataset. In this connection, it is important to clarify that the data was collected from the most popular repositories, which are often owned by technology companies. Thus, this figure may not reflect the actual statistics on empty messages.

Hypothesis B1, that "approximately 14% of the commit messages in the dataset under study are empty messages," was not confirmed.

In the collected dataset, 2952077 messages do not match the V-DO filter, i.e., approximately 94%. Since its application would have resulted in a significant reduction in the size of the dataset, and along with it the useful dependencies between code changes and natural language, the decision was taken not to apply this filter.

In the unfiltered collected dataset, the maximum commit message length is 68529. This is a clear statistical outlier that can negatively affect the algorithm [21].

After removing empty messages, the following statistics were calculated for the number of tokens in commit messages (Table 3):

Table 3. Statistical indicators on the number of tokens in commit messages

Statistical indicators	Value
Number of records (count)	3141186
Mean value (mean)	52.19
Standard deviation (std)	129.61
Minimum (min)	1
25th percentile	10
Median (50th percentile)	42
75th percentile	174
Maximum (max)	68529

The standard deviation (approximately 129.61) is quite large, indicating a significant diversity in message length. The maximum value (68529) is much larger than the 75th percentile (174), indicating that there are outliers with a very large number of tokens.

In order to reduce the impact of outliers, it was decided to keep in the set only those records with the number of tokens in the range from 5 to 600 (inclusive). This filter reduced the number of records in the set to 2761945 or by 12.07%. Figure 2 presents a one-dimensional box plot showing the distribution of the number of tokens in commit messages in the collected dataset.

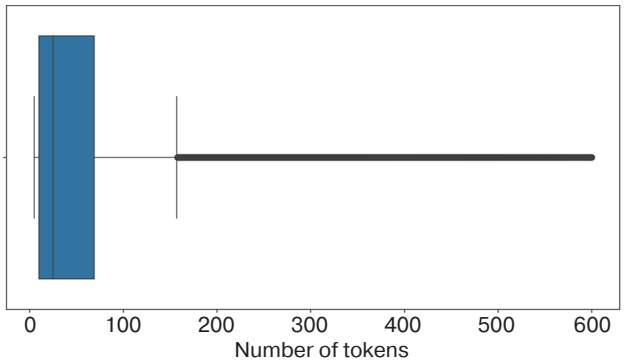

Fig. 2. Distribution of the number of tokens in the commit messages

While the bulk of the data is between 5 and 100 tokens, significant outliers are also visible, indicating a large number of messages with higher token counts.

Regular expressions were applied to filter trivial and generated messages. Keywords from CI-CD utilities were used as templates, e.g., *bump version to*. In addition, non-ASCII characters [9] and records with [bot] token in the *author* sign were removed. As a result of applying this filter, 185540 messages generated by automatic utilities were identified. After applying the filter, the number of records in the set amounted to 2576405.

3.3. Evaluating the semantic proximity of the first sentence and the whole commit message

After filtering, the hypothesis [13] that the first sentence in a commit message is a generalization of the whole message was tested.

In order to conduct experiments to test this hypothesis, a new feature (a column in the dataset) was created, then the first sentences and whole messages were mapped into numerical vectors using the HashingVectorizer utility from the scikit-learn library with a dimensionality of 1048576. Thus, the first sentences and whole messages were converted into vectors consisting of 1048576 numbers each. The program code for implementing the experiment is available in the online supplement to the article.[5]

The mean value of cosine similarity across all commit messages was 0.0969. This value reflects the degree of semantic proximity between the first sentence and the full text of commit messages. The relatively

[5] Online appendix to the article. https://gist.github.com/Malomalsky/a243e43c00adb56fd11c19242a239275. Accessed February 06, 2025.

low mean cosine similarity value may indicate that the first sentence often contains unique information that is not fully repeated in the rest of the message. For the purposes of this study, it was decided not to reduce the commit messages in the dataset to the first sentence.

Hypothesis B2, that "the first sentences in commit messages are a summary of the whole message," was not confirmed.

It is important to note that the cosine similarity procedure only measures the angle between vectors, not their length. This means that cosine similarity does not take into account the amount of information contained in commit messages. More sophisticated methods may be required to analyze the structure of commit messages in more depth.

3.4. Characteristics of the final dataset and its comparison with analogues

The cleaned dataset is available in the online appendix to the paper. The number of records in the dataset as a result of filtering was 2576405. The dataset contains change pairs for 20 programming languages thus extending the scope of a potential commit message generation model to be trained on the dataset. Automatically generated messages, along with trivial and empty messages, were removed from the dataset during the filtering phase.

In order to validate the quality of the dataset, a comparative analysis was performed on the collected dataset and the datasets mentioned earlier. A classification methodology using a pre-trained *commit-message-quality-codebert*[6] neural network was used to perform a comparative analysis of the quality of the commit messages in the proposed dataset. This model, based on the CodeBERT architecture [22], was pre-trained [23] for the task of classifying commit-message quality based on the previously mentioned message taxonomy [1]. According to this taxonomy, a message is "bad" that does not answer the questions 'Why were the changes made?' and 'What changed?', i.e., does not convey the context of the changes made.

The methodological procedure for the analysis included data preprocessing, which involved normalizing the text (lower case) and removing uninformative characters. Commit messages from the collected dataset and from other datasets available for comparison were then passed through a neural network for automatic classification based on the assigned "good" and "bad" labels. The classification results for each of the datasets are presented in Table 4.

From an analysis of the comparison results, the proportion of "bad" messages in the collected dataset was 16.82%, which is significantly lower than the other datasets.

CONCLUSIONS

The study collected an extensive multi-lingual dataset containing changes in program code, their natural language descriptions, and additional meta-information important in the context of filtering and cleaning the dataset. The dataset cleaned of generated and trivialized messages is hosted on the HuggingFace data hosting service.[7]

As part of the study, two hypotheses were proposed based on earlier work on the research topic. Hypothesis B1 was that approximately 14% of the commit messages in the studied dataset are likely to empty. However, when analyzing the collected data, it was found that only 0.0007% of the commit messages in the study sample are empty, which is much less than the expected value. Thus, hypothesis B1 was not confirmed during the study.

Hypothesis B2 was that the first sentences in the commit messages are a generalization of the whole message. To test this hypothesis, we vectorized the text

Table 4. Results of comparative dataset analysis

Dataset name	Total records	Classified as "good"	Classified as "bad"	Share of "bad" records, %
Collected dataset	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

[6] Neural network classifier of messages to the commits. https://huggingface.co/saridormi/commit-message-quality-codebert. Accessed February 06, 2025.

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

and calculated the cosine similarity between the first sentence and the full text of the commit messages. In the study sample, the cosine similarity measure was 0.0969, indicating low semantic similarity between the first sentence and the full message text. Hence, hypothesis B2 was also not confirmed during the study.

The obtained results, which refute both hypotheses, indicate that empty commit messages are extremely rare, and that the first sentence of a commit message is an insufficient summary of the whole message. These findings can serve as a basis for further study of the structure and content of commit messages, as well as for the development of systems for automatic generation of commit messages.

The idea of conducting further research on vectorizing diff files (a representation of changes generated by a version control system) seems reasonable. If diff is a set of additions and deletions, then addition and subtraction operations should be defined for vectorized diff, where addition can be interpreted as adding program code and subtraction as removing fragments from it. Such a potential algorithm would solve the problem of reducing programming languages to a common formal notation under the assumption that identical program code changes made to files written in different programming languages would have close cosine distance. This question will be considered in further study.

It is also worth noting the feasibility of identifying generated messages using neural networks. While this approach will be much more computationally complex, it should provide greater accuracy in classifying commit messages.

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.

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. 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. 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

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

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

Translated from Russian into English by Lyudmila O. Bychkova
Edited for English language and spelling by Thomas A. Beavitt