UDC 65.014.12 **doi: 10.32620/reks.2024.4.19**

Mykyta ROHOVYI, Marina GRINCHENKO

National Technical University "Kharkiv Polytechnic Institute", Ukraine

TOWARDS THE IMPROVEMENT OF PROJECT TEAM PERFORMANCE BASED ON LARGE LANGUAGE MODELS

The subject of *the study is a method for identifying poor quality project sprint task descriptions to improve team performance and reduce project risks.* ***The purpose of*** *the study is to improve the quality of textual descriptions of sprint tasks in tracking systems by implementing models for identifying and improving potentially poor task descriptions.* ***Research Questions****: 1. Can poor quality project sprint task descriptions be identified using clustering? 2. How to utilize the power of large language models (LLMs) to identify and improve textual descriptions of tasks?* ***Objectives: to*** *analyze research on approaches to improving descriptions using clustering and visualization techniques for project tasks, to collect and prepare textual descriptions of sprint tasks, to identify potentially poor task descriptions based on clustering their vector representations, to study the effect of prompts on obtaining vector representations of tasks, to improve task descriptions using LLMs, and to develop a technique for improving project team effectiveness based on LLMs.* ***Methods of*** *vector representation of texts, methods of dimensionality reduction of PCA and t-SNE data space, methods of agglomerative clustering, methods of prompting were used. The following* ***results*** *were obtained. An approach to improving the performance of the project team based on the use of LLM was proposed. Answering the first research question, it was found that there are no linguistic features affecting the perception of textual descriptions of project sprint tasks. In response to the second research question, a model for identifying potentially poor task descriptions is proposed to reduce project risks associated with misunderstanding of task context.* ***Conclusions.*** *The results suggest that project sprint task descriptions can be improved by using large-scale language models for project team understanding. Future research recommends using project source documentation and project context as a vector repository and source of context for LLM. The next step is to integrate the LLM into the project task tracking system.*

Keywords: *project; project team; task description; project task management system; model; neural network; large language model.*

1. Introduction

1.1. Motivation

In modern times, where the IT market is constantly changing and technologies are developing rapidly, project teams need approaches that allow them to quickly adapt to new conditions and ensure high quality and productivity. 7

Agile approaches contribute to improving the efficiency and effectiveness of a project team's work, ensuring flexibility and transparency, high product quality, and customer satisfaction. One of the most effective and widespread project management approaches is the Scrum methodology [1]. When applying it, the work is divided into short iterations (sprints). During sprint execution, priorities and tasks may change. Scrum allows for quick delivery of functionality and feedback from customers during the early stages of development. During the execution of sprint tasks, errors are detected and corrected quickly, and focus is maintained on important features. This helps the team better understand the scope of work and determine who and how each task will be performed.

The team holds daily short meetings in which the progress of task execution, obstacles, and plans for the near future are discussed. When planning a project sprint, the team discussed the tasks' contents and performed their descriptions. This procedure does not take much time; however, unclear and inaccurate descriptions can lead to misunderstandings by the project team about the tasks, which can result in poor-quality results when creating the product increment [2, 3]. Therefore, task descriptions that are clearly understood by all members of the project team are necessary.

When tracking the execution of the project tasks, their descriptions are recorded using task management systems (tracking systems). Tracking systems [4] provide convenient tools for creating, organizing, and tracking all tasks, allowing all project participants to see who is working on which tasks, their status, and the progress achieved. Thanks to tracking systems, project

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

managers can easily assign and control task deadlines. This helps to avoid delays in project execution and to plan resources more effectively. Tracking systems provide the ability to comment, discuss, and collaborate on tasks, which improves communication and teamwork.

Assessment of the quality and clarity of task description formulations plays a key role in the project management process for software development. Accurate and clear task formulations allow the project manager and team to better understand the scope of work and evaluate resources and risks for each task, which impacts the achievement of the overall project goals.

A popular method for developing project tasks used by IT companies is the 3C method (clear, concrete, concise) [5]. It is a widely accepted standard for project task formulation that is intended to enhance the clarity, concreteness, and conciseness of descriptions. This method is used to ensure the quality and success of projects because it promotes clarity, agreement, and responsibility among project team members. The rules of this method stipulate that each task must be [5]:

1. Clear: The task description should be understandable for all participants. It should be written to avoid misunderstandings about what exactly needs to be done.

2. Concrete: The task should include all the necessary information required for its completion, including the scope of work, requirements, success conditions, and other important details.

3. Concise: The task description should be short enough to be easily understood and quickly comprehended by all team members.

The 3C method for describing project tasks in Scrum helps ensure a systematic and effective approach to project management, makes the development process transparent, and promotes rapid adaptation to changes. Using these rules helps avoid misunderstandings and deficiencies in the task execution process, improves communication among project participants, and ensures clarity in task execution.

However, the project team may follow these rules at the level of creating user stories; yet, when describing sprint tasks, these rules are not always adhered to.

The project team must understand which tasks are described qualitatively, meaning that they are clear to any team member and require additional explanations for successful execution. It is necessary to identify and cluster poorly developed sprint tasks as early as possible to minimize the risk of project task failure.

Currently, the use of Large Language Models (LLMs) is gaining momentum, and they can be useful for project management in various fields. LLMs represent a class of scalable, pre-trained language models that are characterized by significant size and training data volume, allowing them to perform a wide range of natural language processing tasks with high accuracy [6]. These models, due to their large size and pre-training on large volumes of textual data, exhibit special capabilities that enable them to achieve excellent results without any specific training in many natural language processing tasks.

LLMs can be used to create various types of documentation, such as project plans, project status reports, task descriptions, requirement specifications, and more [7, 8, 9]. With LLMs, it is possible to analyze textual information, such as emails, reports, comments, and so on, to identify key information and issues that require the project manager's attention [7, 9].

The use of various LLM capabilities with tracking system data helps to avoid misunderstandings, establish priorities, and determine the progress of each task during the sprint. Short sprint task entries in natural language allow team members to quickly grasp the essence of the tasks [10]. These entries include keywords and phrases that identify the main aspects of the tasks.

1.2. State of the art

The task distribution process by the team is lengthy and complex, performed specifically depending on the project stage and team composition, based on the skill set of each participant. Sometimes, task names and descriptions include specific terminology, which complicates task distribution over a long period and makes the automation of this process nearly impossible using traditional natural language processing methods.

This process involves a detailed analysis of existing experience aimed at evaluating the quality of task descriptions in projects that use Scrum. In this section, we discuss studies that present approaches to project task clustering.

In the work [11], a method for requirements detection was developed and empirically tested on numerous examples. This study is based on existing evidence regarding natural language quality defects in requirement artifacts. The evaluation of this approach revealed that automatic detection features have an average precision of 59%, with an average recall of 82%, and significantly vary. The proposed method enhances the identification of quality defects but does not always allow clear differentiation of some features. Some features require more precise definitions; however, overall, the detection of features is a useful tool for supporting quality in the project requirements development process.

The approach proposed in [12] combines data visualization with two natural language processing methods: conceptual model extraction and semantic similarity assessment. The conducted experiments confirmed that identifying terminological ambiguities requires significant effort, even with technical tools, and it is chal-

lenging to determine whether the use of synonyms in task formulation can impact the correctness of software development.

The visual analysis method DeepNLPV is [13] is aimed at understanding and analyzing model behavior in text classification tasks and studying the reasons for success or failure in specific cases. This study is based on information that provides quantitative explanations of how each layer of the model supports the information of the words in the sample. The results demonstrate that task descriptions and comments can be effectively used to predict task success with high accuracy.

In the work [14], a method for ticket classification was presented through clustering and visualization of each category's characteristics using a heatmap and principal component analysis (PCA). It automatically assesses the number of categories and classifies issue tickets. This study examines the visualization of the request term, showing changes in the time series for quick review, while the visualization feature displays the relationships between the request categories and their key category keywords using a heatmap and PCA. The results confirm that the method supports experts and project stakeholders in understanding its features by studying the lifespan of issue ticket categories and keywords.

The language models RoBERTa and CodeBERT-MLM [15] are used to create embedded source codes that capture semantic and contextual features, aiding in language understanding tasks such as SDP. The experiments were conducted in a cross-version SDP scenario for Apache Calcite, which is an open-source data management framework. The evaluation results of defect classifiers demonstrate statistically significant improvement when code representations are learned by pre-trained models compared to semantic representations provided by other natural language-based models, such as doc2vec and LSI.

In [16], a study was conducted on deep learning approaches for the task of Software Defect Prediction (SDP) using the semantic features of source code to predict software defects based on semantic functions. This study employed deep learning to create SDP models based on semantic features. The obtained results will help software developers choose deeper learning methods and code representation techniques for different scenarios. This will enable the identification of relationships between project types, code representation methods, deep learning techniques, and the necessary measures for evaluating such situations.

In [17], a method was proposed to visualize Convolutional Neural Networks (ConvNet) for text processing, allowing observation of how each word influences the final classification. The Adam optimizer and stochastic gradient descent were used with a learning rate of 0.001 for all models. The proposed method can be used for defect detection and improving pre-trained models and classifiers.

In the study presented in [18], key quality factors of requirements were identified using data from interviews with experts, report analysis, and observations of the development process. The empirical research results confirmed that identifying and implementing defined factors, including clarity of formulation, completeness, compatibility, and changeability, can significantly improve the quality of requirements in industrial projects. This, in turn, positively impacts the overall software quality.

The TABASCO tool [19] detects and resolves ambiguities in software requirements and other documents. The authors applied a K-means clustering algorithm with a cosine similarity metric, which allows creating multiple clusters for a candidate noun such that each cluster contains all instances used in a similar context. The experimental results demonstrate that TABASCO can be useful for identifying intra-domain ambiguities in software requirements and other project-related documents written in natural language.

The automated analysis of syntax requirements and pattern detection [20] employs natural language processing techniques to extract syntactic features of requirement formulations and employs a community detection algorithm to group them into logical classes based on syntactic similarity. The experimental results demonstrate that the proposed method effectively recognizes known standard requirement patterns with 90% accuracy.

In the study [21], a methodology based on machine learning (ML) was proposed for classifying and prioritizing requirement defects (requirements smells) that require attention. The authors accumulated 3100 requirements and obtained expert annotations. Common algorithms, such as Logistic Regression (LR), Naive Bayes (NB), Support Vector Machine (SVM), Decision Tree (DT), and K-Nearest Neighbors (KNN), were used to construct classification and prioritization models, and their effectiveness was compared. The results demonstrate that the LR method using Term Frequency-Inverse Document Frequency (TF-IDF) achieved the highest accuracy of 94% in defect classification. For prioritizing requirements, the SVM method outperformed the other algorithms, achieving an accuracy of 99%.

The analysis of studies has shown that there are approaches to cluster the descriptions of project tasks and requirements. Many approaches focus on clustering the descriptions of user stories; however, qualitatively describing sprint tasks is insufficiently researched. However, poorly developed or unclear sprint task descriptions can lead to misunderstandings and errors in

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

their execution, as well as a risk of disrupting the effective implementation of the project. Therefore, clustering sprint task descriptions are a relevant task.

Determining an approach to task clustering will allow the project team to properly focus on refining and improving the descriptions of these tasks, which will help reduce the likelihood of errors and misunderstandings. The quality and clarity of the textual descriptions of tasks in a sprint are important factors that affect the effective execution of tasks, which is a key aspect of the project team's activities.

1.3. Objective and Approach

In a tracking system, the project team describes tasks to ensure that their descriptions clearly convey the essence of the required actions. It is critically important that all team members have a consistent understanding of the tasks to minimize the risks of ambiguities and misunderstandings. Practice demonstrates that executors often supplement task descriptions with comments, videos, images, etc., enriching the context. Storing this information in the tracking system provides the team with a convenient tool for collaboration, simplifies communication, and helps ensure the accuracy and completeness of task understanding by all project participants.

One approach to solving this problem is the use of large language models.

The goal of this study was to improve the quality of textual descriptions of sprint tasks in tracking systems by implementing models to identify and improve potentially poor task descriptions.

To achieve this goal, the following tasks must be addressed:

– analyze research on task description approaches using clustering and visualization methods for project tasks;

– to collect and prepare textual descriptions of sprint tasks and group vector representations of textual descriptions of sprint tasks;

– the impact of prompting on obtaining vector representations of textual tasks and improving task descriptions using LLMs;

– develop technology to improve the textual descriptions of sprint tasks based on LLMs to increase the project team productivity.

The remainder of this paper is organized as follows. The next section presents the research methodology and describes the datasets used in the experimental study. The section 3 is devoted to group vector representations of textual descriptions of sprint tasks and studies the impact of prompting on obtaining vector representations of tasks to improve task descriptions using LLM. As a result, we suggest a pipeline to increase the productivity of the project team based on LLM. Finally, we discuss the obtained results and conclude the study.

2. Materials and methods of research

The following research questions are related to the aim of the study.

RQ1. Can low-quality sprint task descriptions be identified using clustering?

RQ2. How can we use the capabilities of large language models to identify and improve textual task descriptions?

We suggest experimental studies to answer the research questions. The first stage involves collecting and processing text descriptions of tasks from real projects. This study considers open-source projects that are carried out iteratively by the project team. This means that agreed-upon and approved overall tasks are broken down into clearly defined tasks that need to be completed during the project sprint.

The formulation of task descriptions and recording of the execution process during the sprint are expressed in natural language and stored in text format in the Jira task tracking system [22]. Data were collected from 16 public repositories of the Jira tracking system, which contained 1822 projects and 2.7 million releases. The data include historical records of 32 million changes, 9 million comments, and 1 million issue links. The artifact repository stores data in the form of MongoDB databases saved on disk, as well as scripts used for loading the data, interpreting it, and making the data more accessible [15].

For this study, projects that include over 5,000 tasks were selected to ensure acceptable representativeness of the data for analyzing key elements affecting the "understandability" of task descriptions. Each project in this dataset was studied separately to preserve its technological context.

The methodology used to investigate the linguistic features of task texts consisted of several stages, as presented in Fig. 1.

Stage 1: Using "The Public Jira Dataset" involves selecting its relevant parts.

Stage 2: Conducting three series of experiments related to applying natural language processing methods and LLM to clustering and ensure the accuracy of results.

1. First Experiment: A neural network was trained based on the vectorized representation of text, for which the pretrained BERT model (bert-base-uncased) was chosen.

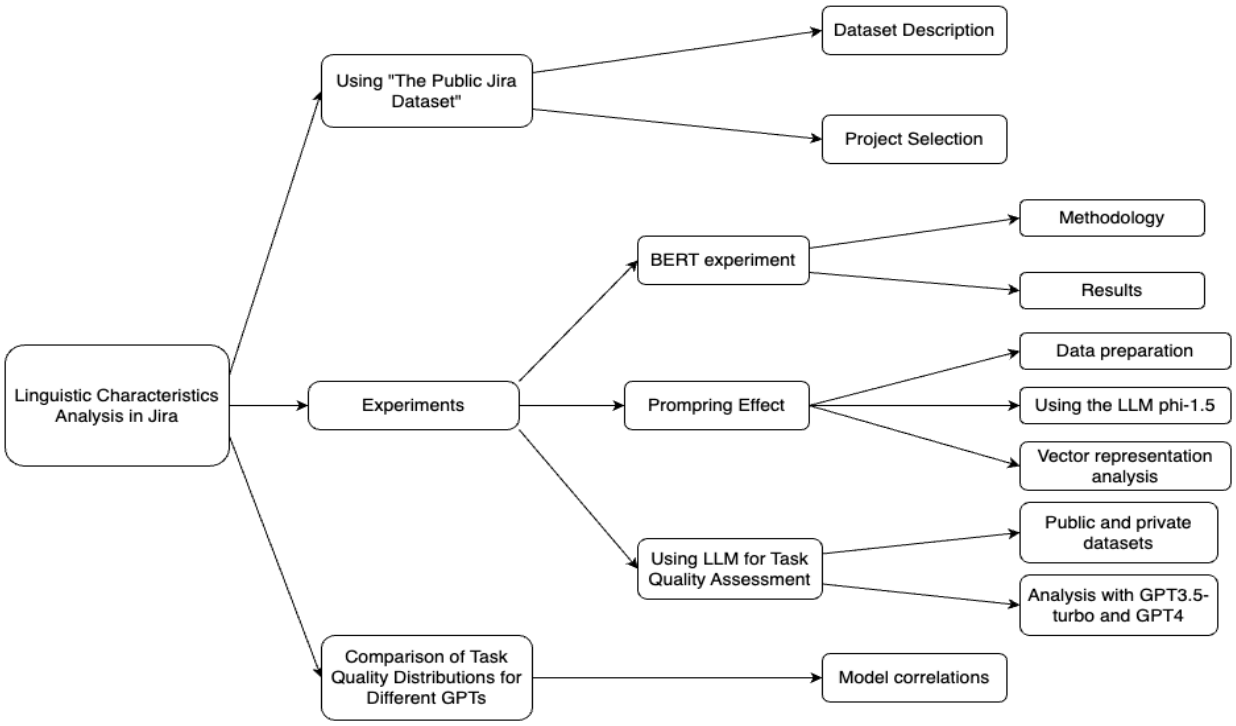

Fig. 1. Methodology for Researching Linguistic Characteristics

2. Second Experiment: We investigated the impact of prompting on obtaining vector representations of textual tasks using large language models.

3. Third Experiment: We explored the results of using LLMs to assess the quality of task descriptions across different datasets.

Stage 3: Comparing model evaluations with expert assessments.

Let's consider the implementation of the proposed technology for researching linguistic characteristics in more detail. In the first stage, the dataset "The Public Jira Dataset" was selected for its representativeness of various projects in the Jira tracking system [22]. A set of text data from real projects was reviewed and described (Using "The Public Jira Dataset"), and relevant parts of it were selected (Dataset description, Project Selection).

The analysis of "The Public Jira Dataset" allowed for a detailed examination of linguistic aspects, including morphological and semantic characteristics that influence developers' interpretation and clustering of tasks.

The dataset comprises publicly available projects in Jira, obtained via Jira API V2, and includes various records, including historical changes to comments and task links [22].

In analyzing the composition of 30 selected open-source projects (The Public Jira Dataset), their quantitative characteristics were determined as follows:

– The average number of words in the task title;

– The average number of words in the task description;

– The average number of words with spaces in the task title;

– The average number of characters in the task description.

The data selection results are presented in Figures 2-5. Fig. 2 and Fig. 3 shows the distribution of the average number of words in the task titles and descriptions, respectively. On average, the number of words in the task title was 8.27, and the average number of words in the task description was 87.38.

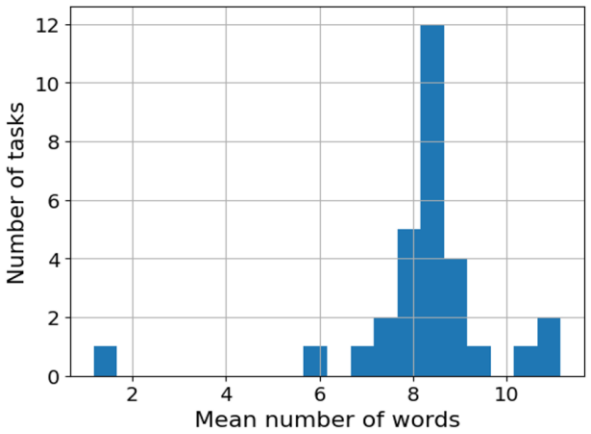

Fig. 2. Distribution of the Number of Words in Task Titles

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

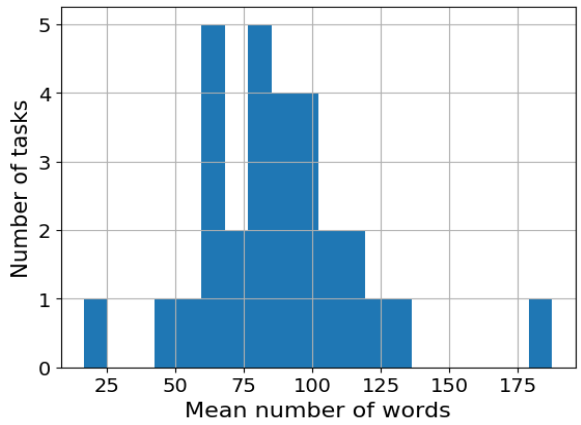

Fig.3. Distribution of the Average Number of Words in Task Descriptions

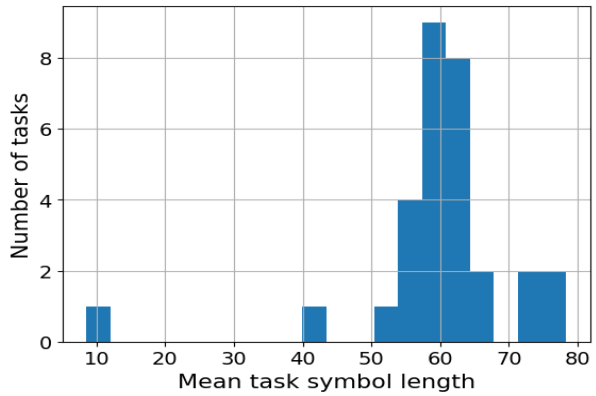

Fig. 4. Distribution of the Number of Characters with Spaces in Task Titles

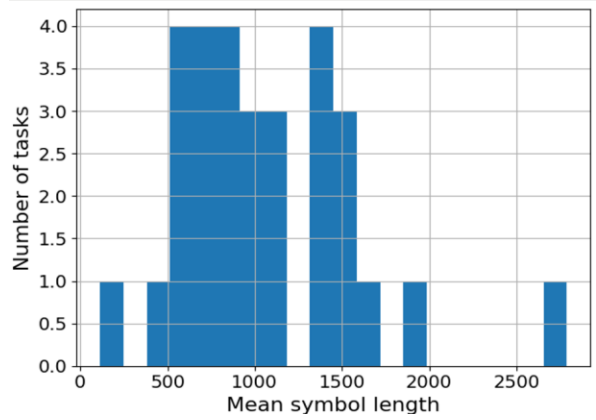

Fig. 5. Distribution of the Number of Characters with Spaces in Task Descriptions

The distribution of the average number of characters in the titles and descriptions of the project tasks is illustrated in Fig. 4 and Fig. 5. The average number of characters in the title and description of each project task was 59.56 and 1051.27, respectively.

Two additional characteristics were identified as essential for each project: task types and priorities. The priority distribution is presented in Fig. 6. It can be observed that most tasks have Major and Minor priorities, and a significant number of tasks also have Critical and Blocker priorities. This step is crucial for accurately estimating task completion times. In Fig. 7, it can be seen that the absolute majority of tasks are of the Bug type, indicating that the projects have already undergone the testing phase, meaning that the tasks have been effectively allocated and completed.

After completing the analysis and preparation stage of the input data for sprint task descriptions, we proceed to the second stage of the study-conducting experiments.

3. Experiments and results

The first experiment was related to training a neural network based on the vectorized representation of text. In this experiment, the pre-trained BERT model (bert-base-uncased) from the Python package for transformer models was selected [23].

The results of the first series of experiments demonstrate that developing a classification model to identify the quality of sprint task descriptions requires the involvement of experts who are familiar with the project's content. Including the formulation of evaluations during the sprint retrospective for training the classifier model and involving project team members as experts is entirely justified. This approach can help collect data for regular model training and improve the consistency of evaluations provided by different experts [24]. According to the proposed methodology described in [24], tasks were transformed into vector representations, which allowed for the identification of clusters and the analysis of their linguistic features.

Using the agglomerative clustering algorithm, which is based on hierarchical merging of data to detect natural groupings in text data, we group vector representations of tasks. This allowed us to identify natural categories and patterns in the data, revealing potential linguistic patterns and structures that characterize different types of tasks.

For clarity, task representations were transformed into a two-dimensional space, allowing visualization of the distribution and connections between task categories. These representations and clusters, visualized in different colors, demonstrate the distribution and interconnections between tasks, as shown in Fig. 8 and Fig. 9.

For clarity, we divided the tasks into two and three clusters.

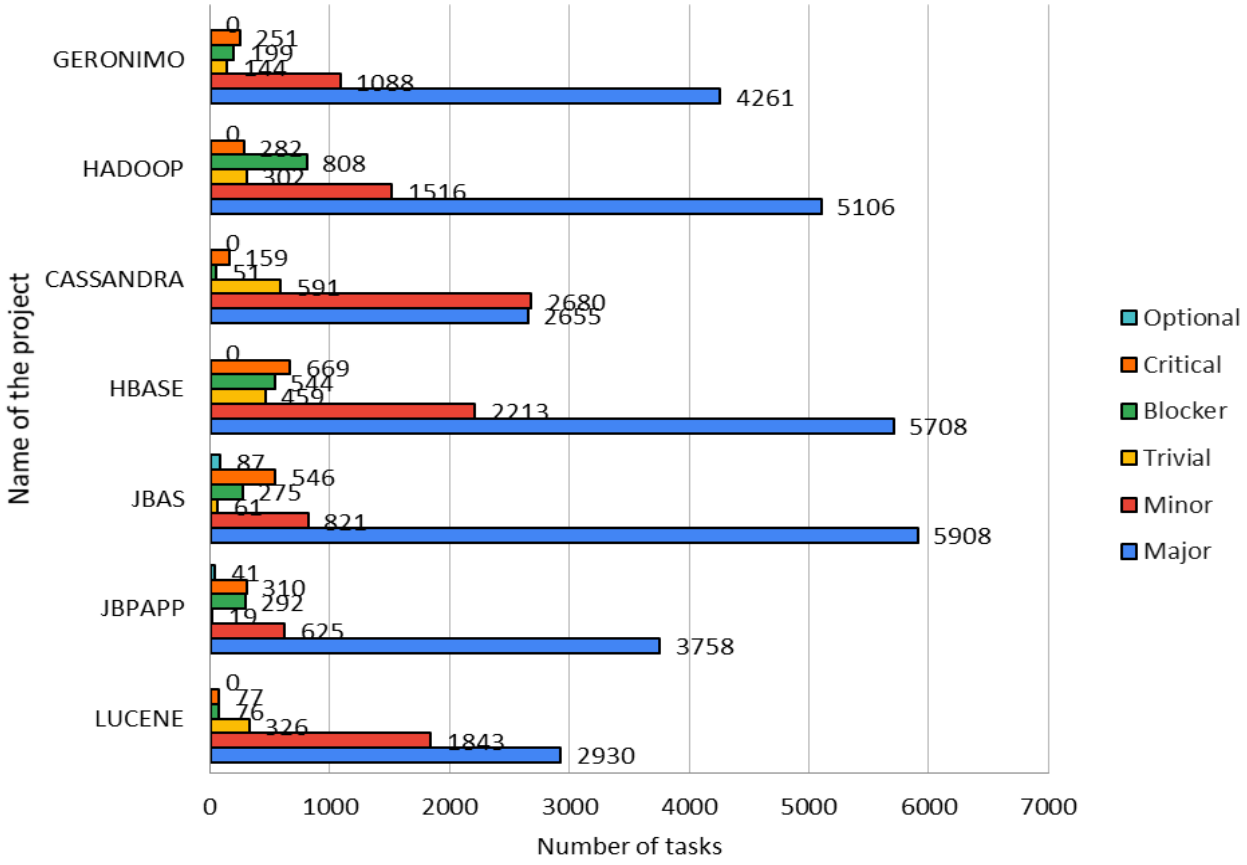

Fig. 6. Distribution of Project Tasks by Priority

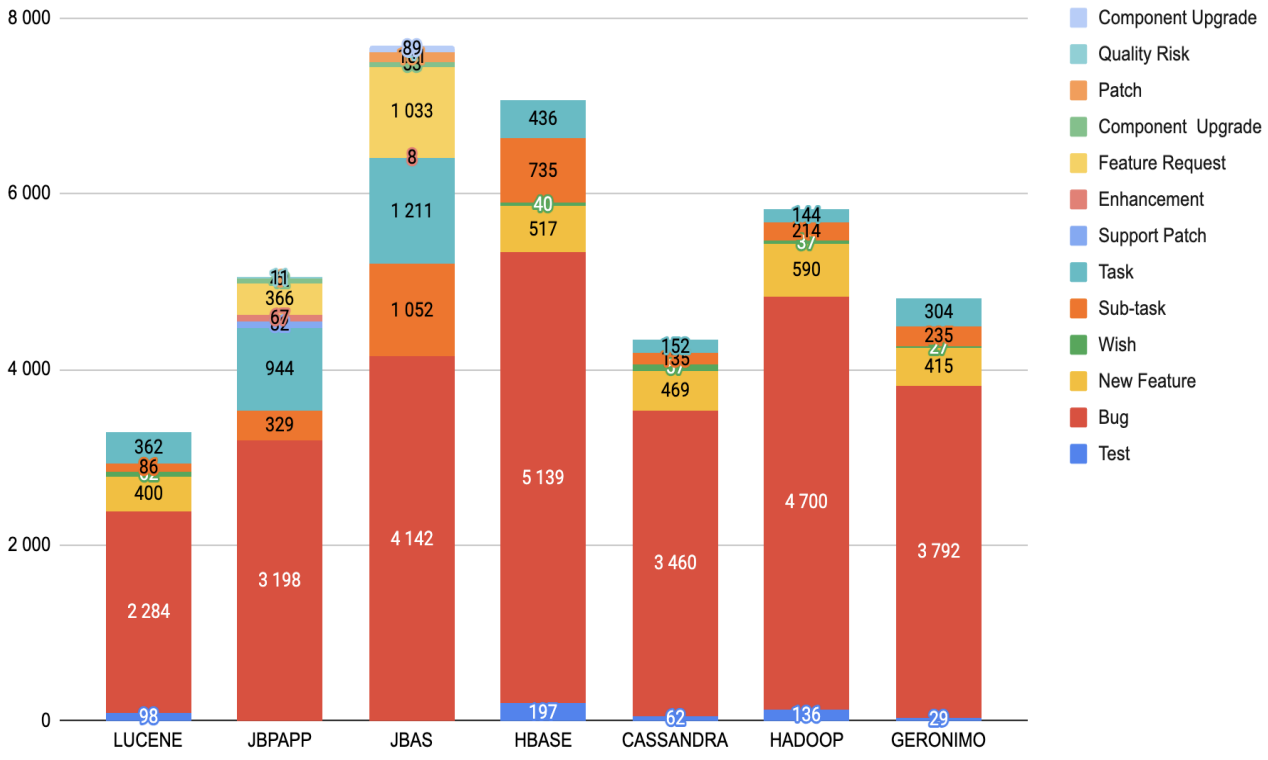

Fig. 7. Distribution of Tasks by Task Type

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

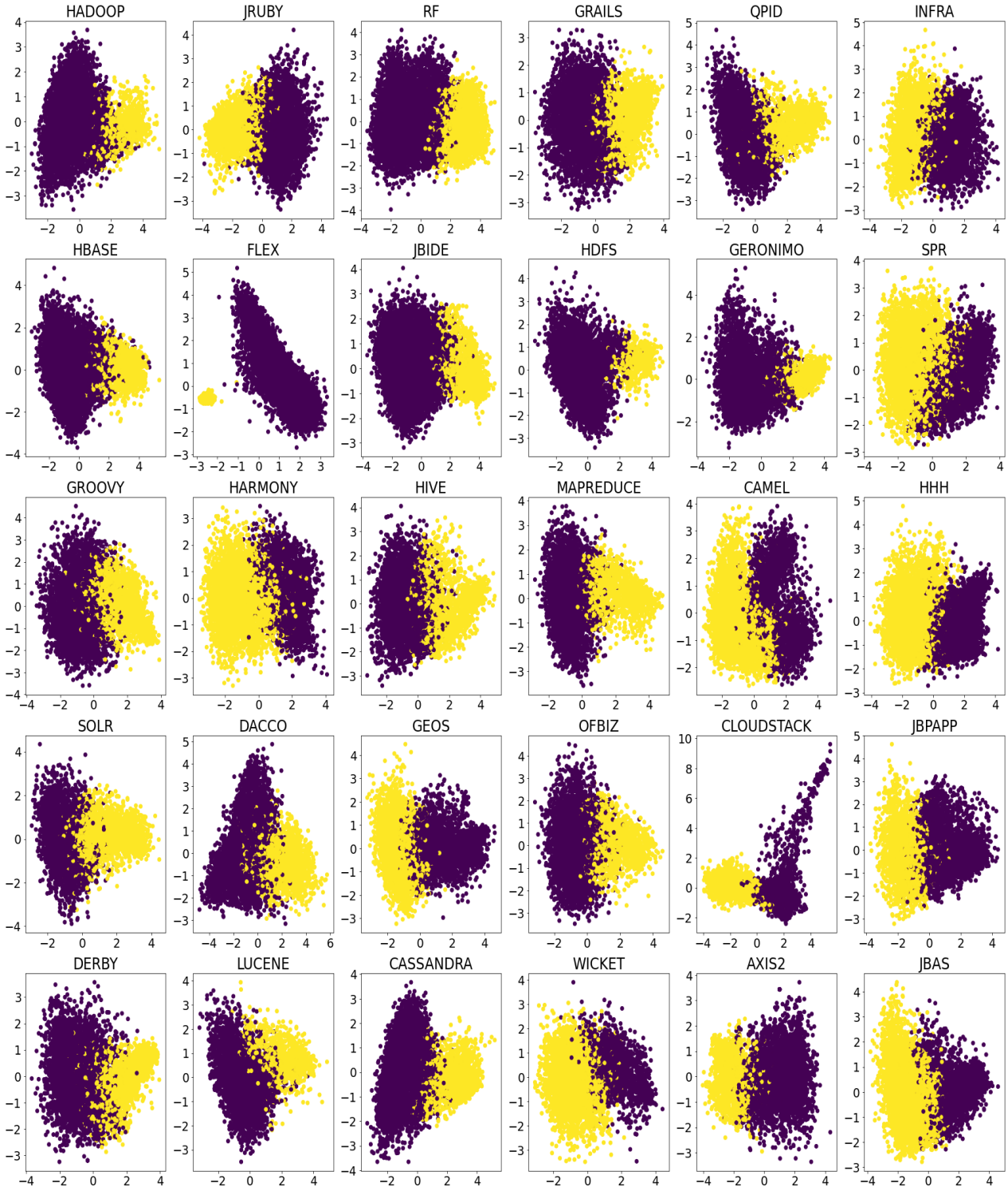

Fig. 8. Project Task Clustering Using BERT Method (Two-Dimensional)

The two clusters represent idealistic separation into good and bad tasks. However, in practice, there is often several data elements that are outliers and do not belong to either of the two clusters based on their characteristics. Therefore, these elements were grouped into a third cluster (unknown).

By clustering the tasks and analyzing the visual and linguistic representation of the results, we found that it is not always possible to compactly separate tasks into clear, understandable clusters. The only aspect by which some projects were clearly divided, considering their linguistic features, was indicating their belonging to specific types (e.g., 'Bug' or 'Feature/Improvement'). The analysis of the obtained representations and corresponding clusters revealed several possible reasons for this clustering.

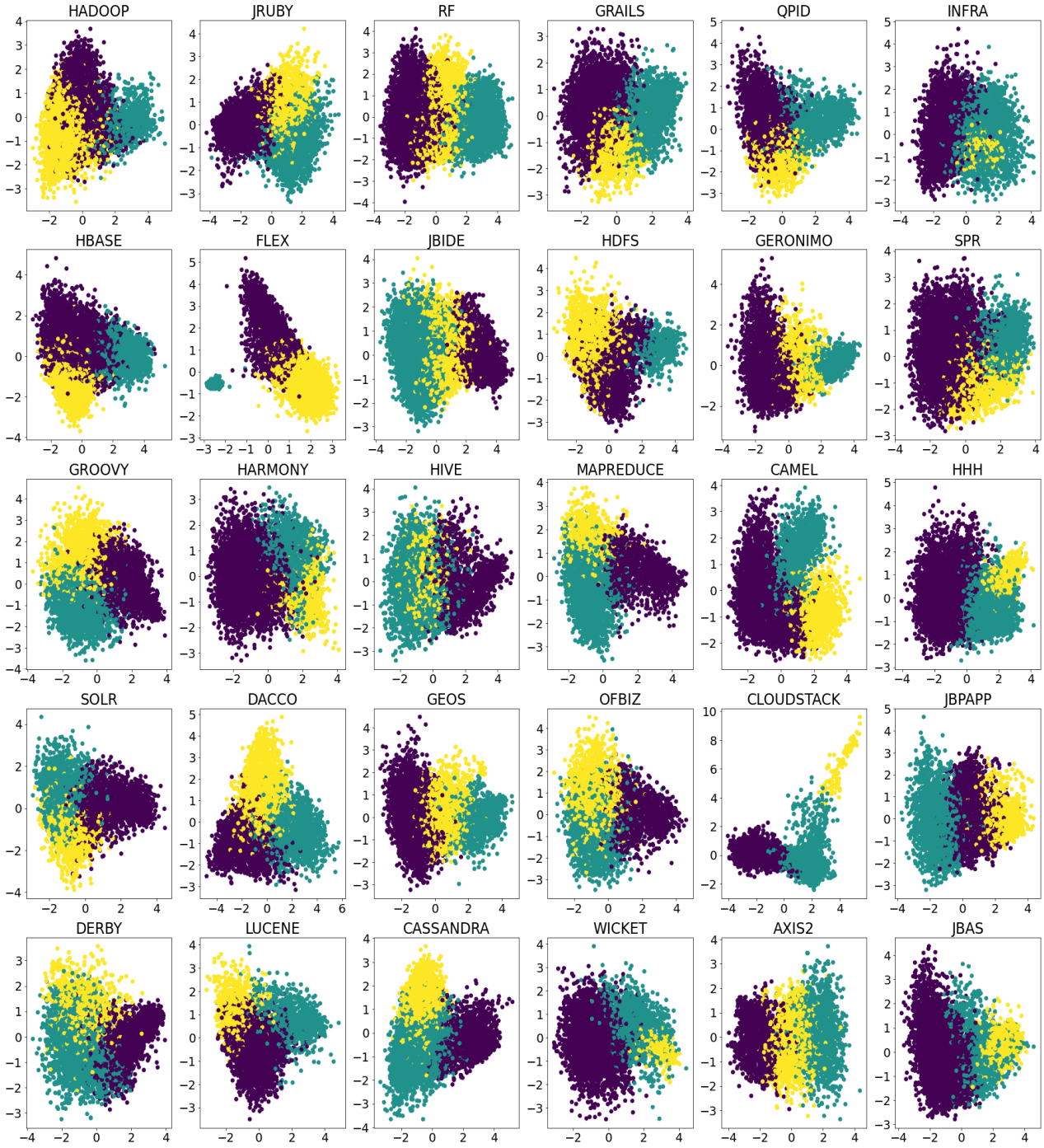

Fig. 9. Project Task Clustering Using BERT Method (Three-Dimensional)

It was found that tasks of different types, such as Bug (a problem in existing code) and Feature/Improvement (tasks for improving the project product), fell into different classes. This finding was confirmed by experts during manual analysis [23].

The experimental results did not allow for clear clustering of sprint task descriptions. We conclude that linguistic features cannot be identified in task quality contexts. Therefore, we conducted a second series of experiments to investigate the impact of prompting using LLMs on obtaining vector representations of textual tasks.

From project management experience, additional files, images, links, code snippets, etc., significantly affect task understanding. For the second stage of the experiments, the same data used in the first experiment were used. Each task in this dataset was pre-processed. The title and description of each task were used. All data were checked for parameter relevance, and special notes in the task descriptions were identified, such as the presence of images, links, code snippets, files, piec-

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

es of XML and JSON, phone numbers, emails, and IP addresses. Thus, each task received a binary indicator of each category.

Since standard language models do not have sufficient internal information, and simply adding a small number of binary attributes usually does not provide enough understanding for algorithms or even give a chance to choose just one for clustering, in this experiment, binary attributes were added as the context for the LLM. To obtain vector representations, LLM phi-1.5 [25] was used. During the experiment, the model and tokenizer were fixed, and only the input text was changed. (Fig. 10).

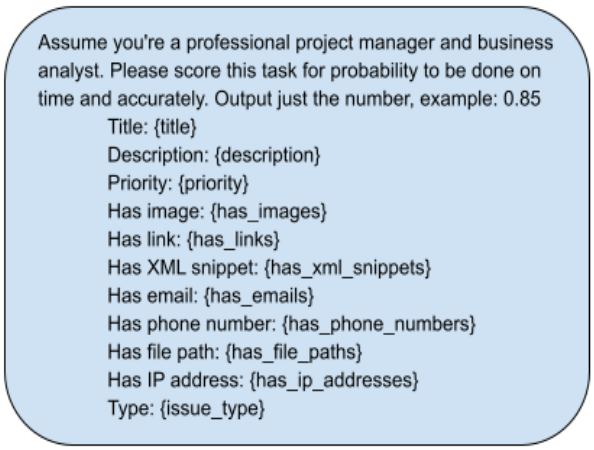

Fig. 10. Prompt for a large language model phi-1.5

In the first case, the input consisted of only the title and description, separated by a period and a space. In the second case, the neural network was given the task by the following prompt (Fig. 10). This setup allowed us to compare the impact of adding contextual binary attributes on the quality and clarity of task descriptions when processed by the LLM.

For vector representation, tasks presented in two text formats were analyzed. Each format was converted into a set of tokens using a specialized tokenizer, after which the tokens were processed by the phi-1.5 neural network.

The tensors obtained from the last layer of the network ('hidden states' of size 2048 elements) served as vector representations of the tasks. Based on these representations, they were clustered into 2 or 3 groups to classify "good", "bad", and "unknown" task types using agglomerative clustering [26].

The number of clusters was selected based on the logic of "good" and "bad" tasks, and "unknown". The number of each attributes in the 30 projects is given in Table 1.

The results demonstrate that file paths and images are the most frequently encountered attributes in tasks.

Thus, class indices were obtained for each project. For clear visualization 2048-dimensional vector representations can be transformed into a two-dimensional space using PCA [27] and t-SNE [28] methods, which allows them to be represented visually.

Table 1

Main Attributes of the Project Tasks Under Consideration

Project	Attributes						
	has images	has links	has xml snippets	has emails	has phone numbers	has file paths	has ip addresses
LUCENE	6	679	86	89	16	1992	15
JBPAPP	11	1432	304	12	20	2580	272
JBAS	5	1110	664	75	35	3458	307
HBASE	7	823	100	69	264	3061	399
CASSANDRA	8	581	29	90	140	2071	307
HADOOP	7	609	93	29	47	2720	159
GERONIMO	7	887	352	74	13	2533	225
HIVE	0	517	40	45	50	1672	40
AXIS2	11	1519	1073	113	26	2719	154
HHH	10	1319	843	61	10	2887	89
HDFS	2	364	46	53	77	1436	185
DACCO	6	76	99	0	13	1483	0
OFBIZ	35	869	313	31	13	2002	130
HARMONY	24	838	70	74	50	2308	63

Continuation of Table 1

Main Attributes of the Project Tasks Under Consideration

Project	Attributes						
	has images	has links	has xml snippets	has emails	has phone numbers	has file paths	has ip addresses
GEOS	80	1508	589	39	74	2449	101
WICKET	47	1026	410	56	17	2031	31
SPR	17	2032	1891	45	14	5080	163
QPID	1	337	41	38	20	1629	148
GROOVY	11	769	184	38	21	1843	27
SOLR	12	1183	571	80	19	2433	80
DERBY	9	1126	69	91	26	2831	550
JRUBY	19	1184	76	75	73	3717	204
CAMEL	3	1960	477	49	6	3038	89
JBIDE	584	2874	893	50	37	5697	95
FLEX	394	16296	4711	10763	66	21202	237
MAPREDUCE	2	319	50	18	30	1600	91
RF	284	1904	1972	18	12	4130	107
INFRA	74	3380	46	1861	21	4122	250
GRAILS	19	1478	661	58	11	3292	89
CLOUDSTACK	13	944	121	78	533	2602	1387

We consider the principal component analysis (PCA) method. PCA is a statistical technique used for dimensionality reduction by transforming many variables into a smaller number of uncorrelated variables, known as principal components. This method helps highlight the main structure of the data while minimizing information loss.

Mathematically, if X – original dataset with dimensions $m \times n$ (where m – the number of observations and n is the number of variables),

PCA seeks a new orthogonal basis for X, where the first principal component has the maximum variance, the second principal component has the maximum variance among those orthogonal to the first, and so on.

The first principal component (PC1) can be found by maximizing the variance as follows:

$$\max\left\{\mathrm{var}\left(Xw\right)\right\} = \max\left\{\frac{w^T X^T X w}{w^T w}\right\}, \qquad (1)$$

where w – weight vector.

The results of vector representation using PCA and t-SNE methods are presented in Fig. 11. The t-SNE (t-distributed Stochastic Neighbor Embedding) method is a machine learning technique for visualization based on dimensionality reduction, which is particularly effective for visualizing large datasets in two- or three-dimensional space. The t-SNE method first computes the pairwise similarity probabilities between objects in high-dimensional space such that closer objects have higher probabilities and distant objects have lower probabilities. Then, the t-SNE algorithm seeks a low-dimensional representation of the data that best reflects these probabilities.

Mathematically, the probability $p_{j|i}$, which indicates that object j is a neighbor of object i in the high-dimensional space, is defined as:

$$p_{j|i} = \frac{\exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\left\|x_i - x_k\right\|^2 / 2\sigma_i^2\right)}, \qquad (2)$$

where x_i, x_j are the high-dimensional vectors of objects and σ_i is the variance of the Gaussian distribution centered on the i-th element.

The t-SNE (t-distributed Stochastic Neighbor Embedding) method is a machine learning technique for visualization based on dimensionality reduction, which is particularly effective for visualizing large datasets in two- or three-dimensional space. The t-SNE method first computes the pairwise similarity probabilities between objects in high-dimensional space such that closer objects have higher probabilities and distant objects

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

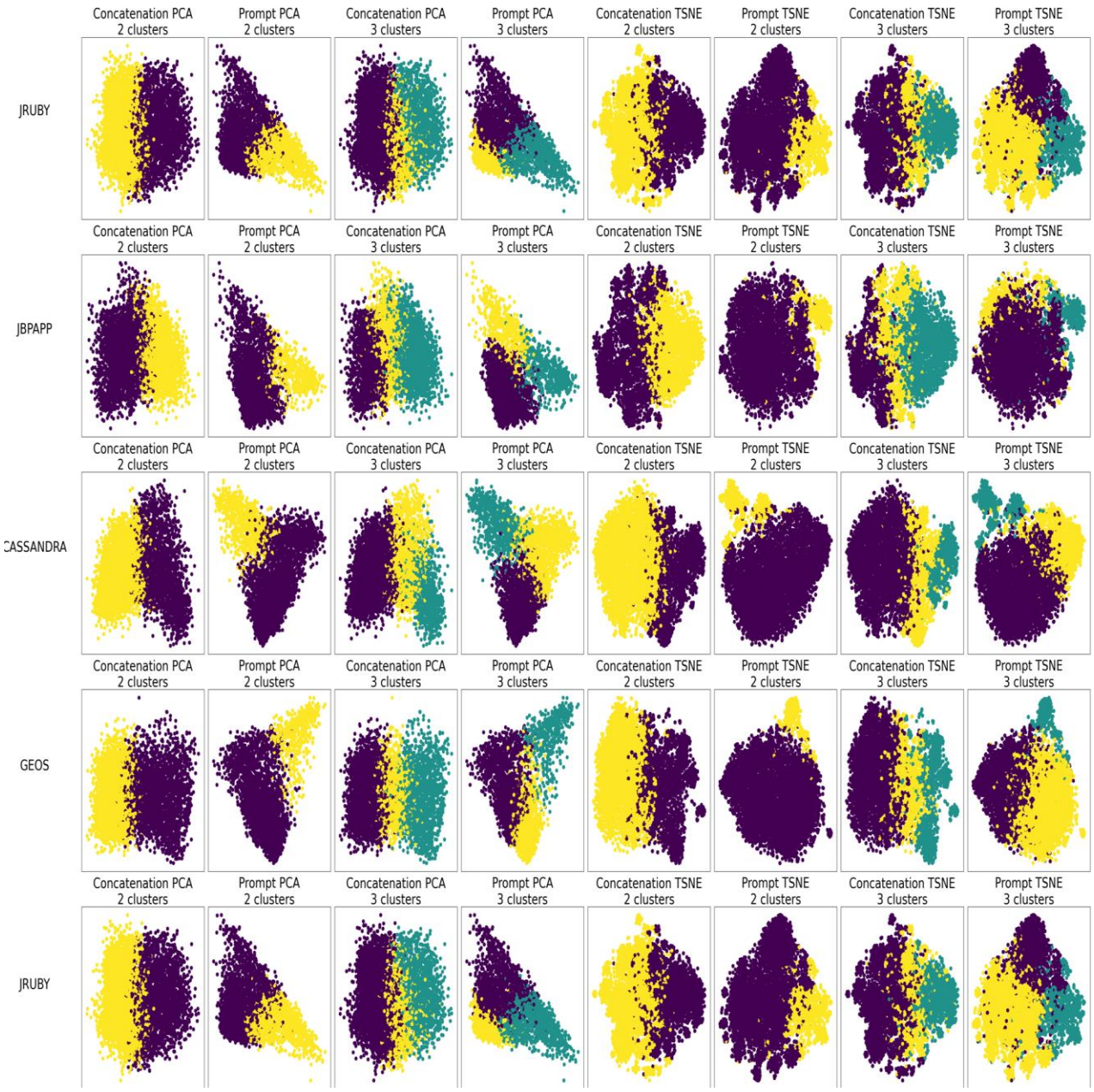

Fig. 11. Clustering of project tasks by the PCA and t-SNE method, part 1

have lower probabilities. Then, the t-SNE algorithm seeks a low-dimensional representation of the data that best reflects these probabilities.

In low-dimensional space, the probabilities q_{ji} are determined similarly, but using the t-distribution with one degree of freedom instead of the normal distribution to prevent the clumping of objects in the center of the map. The optimization involves minimizing the difference between p_{ji} and q_{ji} using the Kullback-Leibler method:

$$C = KL(P \| Q) = \sum_i \sum_j p_{ji} \ln \frac{p_{ji}}{q_{ji}}. \quad (3)$$

The results of vector representation using PCA and t-SNE methods are presented in Fig. 12.

Different clusters are colored depending on their number (2 or 3).

After analyzing the obtained clustering results, we found no linguistic patterns were found, similar to the results obtained with BERT. However, the vector representations obtained through the prompts have a more consistent form, which indicates the potential use of targeted queries in LLM to analyze project tasks.

In the next series of experiments, we explored the use of LLMs to evaluate task quality on various datasets was explored. The experiment identified and cluster factors affecting the quality of textual task descriptions and to test the effectiveness of using LLMs for their evaluation.

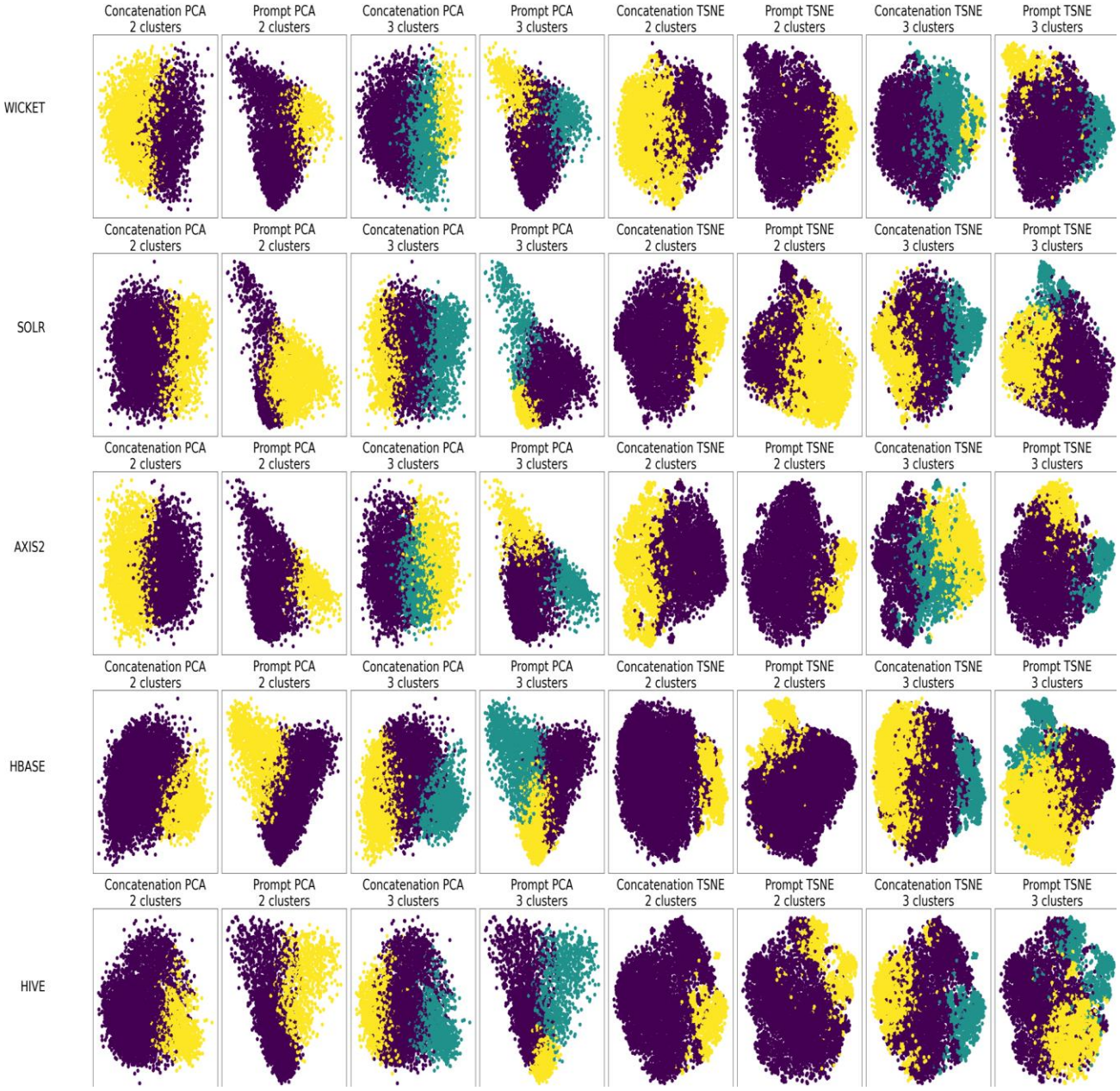

Fig. 12. Clustering of project tasks using the PCA and t-SNE method, part 2

The results obtained using different LLMs were compared. In this experiment, in addition to the public dataset, data from a private project [29] were used. The same search for relevant attributes in task descriptions and the same prompt were performed for these tasks. The query was sent to OpenAI GPT LLM using different versions of this model (3.5-turbo and 4).

During the analysis of 1000 tasks from the private dataset using GPT-3.5-turbo, a distribution of probabilities for timely and accurate task completion was obtained, as shown in Fig. 13.

It can be seen that using the selected threshold value, a task classifier based on their description can be created. However, in the end, there was no correspondence between probability and actual task during the analysis. From this dataset, 100 examples were selected, with 50 from each class marked by experts [29].

These 100 examples were processed in the same manner as in all previous experiments and were analyzed using GPT-3.5-turbo and GPT-4. In Fig. 14, it can be seen that the distributions of these two models do not match at all, which indicates the inconsistency of the models when performing the task.

Similarly, 100 random elements were selected from a single public dataset project, for which the probability distribution is presented in Fig. 15. In this case,

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

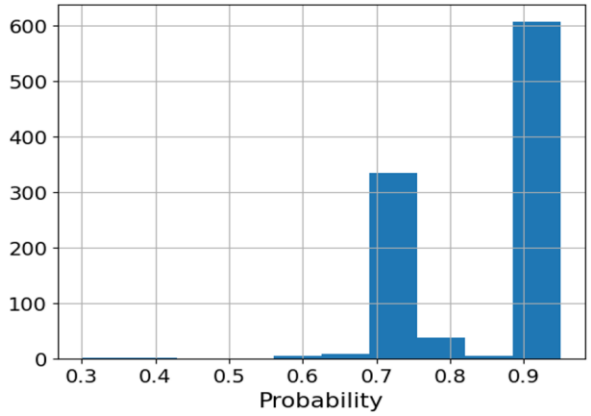

Fig. 13. Probability distribution of tasks

the visual similarity of the data distributions, but after analyzing the correlation between them, a result of 16% was obtained, which confirms the hypothesis about the possibility of using LLM for classification.

The experimental results emphasize the critical necessity of involving experts with a deep understanding of the project to develop an effective classification model that evaluates the quality of sprint task descriptions.

The effectiveness of such a model largely depends on the correct choice of the vectorization method and the adaptation of the classification model to the specifics of the data obtained from the project tracking system.

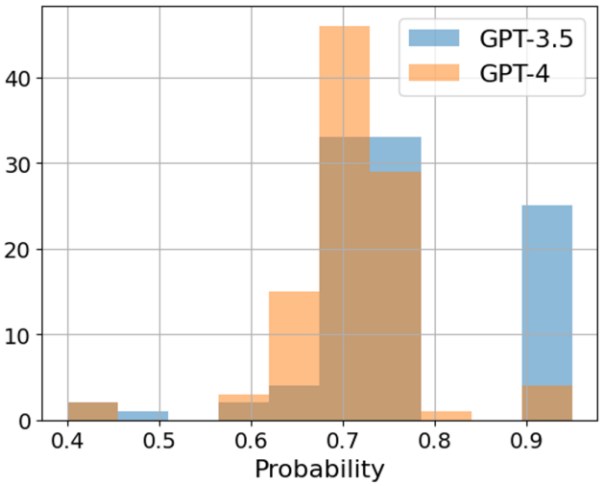

Fig. 14. Probability distribution of tasks using GPT3.5-turbo and GPT4

In the third phase of the research technology, a comparative analysis of the quality of model assessments was conducted with expert assessments. At this stage of the research, according to the proposed technology, attempts were made to evaluate and adjust tasks to assist the project manager.

For this purpose, 274 tasks from a private project were selected based on the uniform distribution of positively and negatively marked tasks by experts.

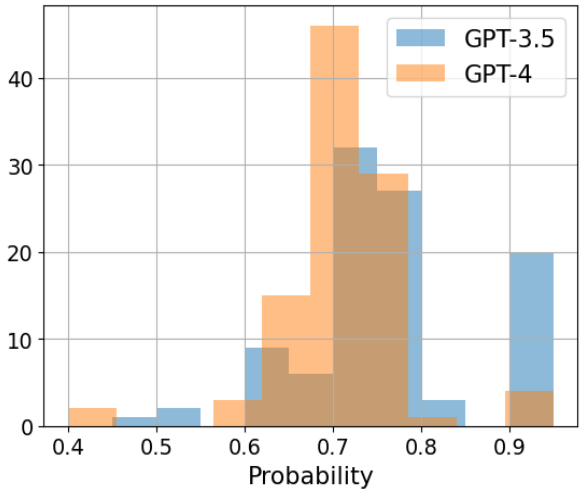

Fig. 15. Comparison of probability values distribution of timely task completion for different GPT models

The GPT-4 model was selected using OpenAI API to accomplish this task. Each task was processed to identify relevant indicators, as in previous experiments. A prompt was created to implement this task, which is presented in Fig. 16.

```
Assume you're a professional project
manager and business analyst. Please
score this task for probability to be done on
time and accurately. Output the number and
how the task can be changed.
Example:
Probability: 0.85 title: 'improved version of
title' description: 'improved version of
description'
Title: {title}
Description: {description}
Priority: {priority}
Has image: {has_images}
Has link: {has_links}
Has XML snippet: {has_xml_snippets}
Has email: {has_emails}
Has phone number: {has_phone_numbers}
Has file path: {has_file_paths}
Has IP address: {has_ip_addresses}
Type: {issue_type}
```

Fig. 16. Prompt for GPT-4 model via OpenAI API

The results of the experiment are given in Table 2.

Private project tasks are submitted sequentially to the model to obtain the probability of task clarity and to improve possible task titles and descriptions.

Table 2

Results of an experiment using the GPT-4 model via the OpenAI API

Input	Output	Probability
Title: Magic wand error on thermal images	Title: “Detailed Error report on magic wand tool for thermal images” Description: “Detailed bug report including error message codes, screen-shots, and steps to reproduce Magic Wand error on thermal images”.	0.6
Title: Research depth map to 3D	Title: “In-depth Research on 3D Mapping”. Description: “Conduct extensive research on 3D mapping techniques and technologies, focusing on latest models, accuracy, efficiency, and operational requirements. The research should include both open-source and commercial solutions available in the market. The research output should involve a well-compiled, comprehensive document presenting comparisons, pros, cons, and practical applications of each technology”.	0.7
Title: FE error on upload image without geo data Description: Error in console in Asset page in image details when image without geo data	Title: “Correct Front-End Error During Uploading Images Lacking Geo-Data Description: “An error is being experienced on the Asset page on the console. This error specifically occurs during the upload of image details that do not contain geo-data. A high level of priority has been assigned to this task. To expedite the resolution of this issue and for accurate diagnosis, please include any unique error messages, steps leading up to the error, screenshots, and any relevant links or snippets of code”.	0.65
Title: Reidentification doesn't work	Title: “Resolve Reidentification Functionality Issues Description: Carefully trace the issue with the reidentification functionality, identify the root cause, and develop an effective solution to fix it. This task must be rigorously tested before being considered complete to ensure that the issue has been fully resolved and no new issues have been created in the process”.	0.6
Title: Icon colors on the map Description: In the map view of the inspection page, asset and image icon colors should be the following: Asset: We should be able to differentiate on the map between assets that are 1. inspected with no defects (green), 2. assets with defects (Red) 3. Assets with images associated but not inspected. (Dark Gray) 4. Assets without images (Light Gray) Images" For images 1. Inspected with defects (Red) 2. Inspected without defects (Green) 3. Not inspected (Dark Gray)	Title: “Enhanced Asset and Image Icon Color Coordination in Map Inspection” Description: “For the mapping interface of the inspection page, we improve the color-coding scheme. This involves using distinct colors to distinguish various asset and image statuses. The color indications for the following situations shall be: For Assets, we use: 1. A “green” to denote assets inspected and found without defects. 2. A “red' for assets with detected defects. 3. Assets with associated images but yet to be inspected should be distinguished with dark gray. 4. “Light gray” denote assets without any images attached. For images, the color identification scheme should be: 1. Red for inspected images containing defects. 2. Green for inspected images that are found to be defect-free. 3. Images yet to be inspected can be distinguished using dark gray”.	0.8

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Continuation of Table 2

Results of an experiment using the GPT-4 model via the OpenAI API

Input	Output	Probability
Title: Add waiting spinner on sign-up request process	Title: "Add waiting spinner on sign-up request process"	0.9
Title: Create report about regenerated models on Prod	Title: Produce Report on Regenerated Models in the Production Environment Description: Create a comprehensive report detailing the regenerated models in the production environment. This report should include information about the model regeneration process and the impact of these changes. This task is considered medium priority.	0.85
Title: Delete images from raptormaps on demo account	Title: Delete images from RaptorMaps on Demo account Description: This task involves deleting images which are no longer needed or relevant from the Demo account. As this task will involve some computing skills and knowledge of the RaptorMaps platform; it's assumed that the team is familiar with it and has access to this account.	0.9

Based on the obtained results, we conclude that understanding some tasks can be improved significantly with the help of large language models. Thus, LLMs can be integrated as an aid to project managers in developing sprint tasks and improving team efficiency by enhancing the linguistic quality of tasks.

4. Discussion

This study focuses on improving the quality of sprint task descriptions to mitigate the risk of project delays or failures. Our experiments revealed that textual data in project tracking systems often include specific abbreviations, slang, and unclear expressions, which complicate the understanding and processing of such descriptions without contextual project information. Despite efforts to use advanced language models like BERT for task clustering, these methods did not yield consistent results in terms of task understandability. This proved that the approach was ineffective.

In answer to the first research question, we determined that there are no linguistic features that affect the perception of text descriptions of project sprint tasks. In response to the second research question, a model for identifying potentially poor task descriptions is proposed, which reduces the project risks associated with misunderstanding the task context.

It is crucial that incorporating expert assessments from the project team during sprint retrospectives is effective for improving task descriptions and training the model. The team members' expertise increases the accuracy of highlighting potentially risk descriptions because of their deep understanding of the project context.

Future research plans to use the project's initial documentation and context as vector storage and source of context for LLMs, as well as to integrate contextual LLMs into the task tracking system as a copilot for the project manager.

It is possible to use the context of already well-formulated tasks and the initial project documentation or requirements combined with the use of LLMs for a better understanding of the project's essence, which, in turn, will improve the formulation of sprint tasks.

5. Conclusions

As a result, we propose an innovative pipeline to leverage LLMs for task management:

1. The project manager (PM) creates a task.

2. The LLM, equipped with the context of all tasks, the overall project objectives, and information about team members, evaluates the task and assigns a score based on its clarity and comprehensibility.

3. If the task received a low score, an alert was triggered for the PM, accompanied by suggestions to improve the task description.

This approach ensures that tasks are created with greater clarity, leading to better team understanding and project execution. In addition, it emphasizes the potential of LLMs to assist in continuously improving task descriptions.

In future research, we recommend incorporating project source documentation and contextual infor-

mation as a vector repository to provide a richer context for LLM. This integration can enhance the model's ability to evaluate and suggest improvements to task descriptions. Expanding the dataset from other tracking systems and involving more experts to assess task description quality may also be beneficial. Furthermore, developing a model for task distribution within the project team and exploring options for task and resource allocation could minimize potential risks.

By incorporating LLM into the project management workflow, we can create a more efficient and effective task management process that improves the performance of the project team. The results of this study highlight the importance of clear and quality textual descriptions for the successful completion of project tasks and the need for improved tools for clustering and analyzing textual data in project management.

Contribution of authors: conceptualization – **Marina Grinchenko**; formulation of tasks – **Marina Grinchenko**; the review and analysis of references - **Marina Grinchenko**; prepared dataset of experiments and analyses results – **Mykyta Rohovyi**; verification – **Mykyta Rohovyi**; analysis of results – **Mykyta Rohovyi, Marina Grinchenko**; visualization – **Mykyta Rohovyi**; writing – original draft preparation – **Mykyta Rohovyi**; writing – review and editing – **Marina Grinchenko**.

Conflict of Interest

The authors declare that they have no conflict of interest concerning this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

This study was conducted without financial support.

Data Availability

The data associated with this work are stored in the data repository.

Use of Artificial Intelligence

The authors have used artificial intelligence technologies within acceptable limits to provide verified data, which is described in the research methodology section.

All the authors have read and agreed to the publication of the finale version of this manuscript.

References

1. The Scrum Guide. *The Definitive Guide to Scrum: The Rules of the Game*. Available at: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf#zoom=100 (accessed 10.05.2024).

2. Cleland, D. L., & Gareis, R. *Learn and apply successful international project management techniques*. McGraw Hill Professional, 2010. 575 p.

3. Strielkina, A., Tetskyi, A., & Krasilshchykova, V. Risk and uncertainty assessment in software project management: integrating decision trees and monte carlo modeling. *Radioelectronic and Computer Systems*, 2023, no. 3(107). pp. 217-225. DOI: 10.32620/reks.2023.3.17.

4. Jira Software. [The IT industry]. Available at: https://www.atlassian.com/software/jira. (accessed 10.05.2024).

5. Covey, S. R. *The 7 Habits of Highly Effective People: Powerful Lessons in Personal Change*. Simon and Schuster, 2004. 372 p.

6. Kalyan, K. S. A survey of GPT-3 family large language models including ChatGPT and GPT-4. *Natural Language Processing Journal*, 2024, vol. 6, pp. 100048. DOI: 10.1016/j.nlp.2023.100048.

7. Agathokleous, E., Rillig, M., Peñuelas, J., & Yu, Z. One hundred important questions facing plant science derived using a large language model. *Trends in Plant Science*, 2023, vol. 29, no. 2, pp. 210-218. DOI: 10.1016/j.tplants.2023.06.008.

8. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., & et al. Language Models are Few-Shot Learners. arxiv, *Computation and Language*, 2020, pp. 1-75. DOI: 10.48550/arXiv.2005.14165.

9. Cherednichenko, O., Ivashchenko, O., Lincényi, M., & Kováč, M. Information technology for intellectual analysis of item descriptions in e-commerce. *Entrepreneurship and Sustainability Issues*, 2023, vol. 11, no. 1, pp. 178-190. DOI: 10.9770/jesi.2023.11.1(10).

10. Peggy, G., Strode, D., Sharp, H., & Barroca, L. An onboarding model for integrating newcomers into agile project teams. *Information and Software Technology*. 2022, vol. 143, article no. 106792. DOI: 10.1016/j.infsof.2021.106792.

11. Femmer, H., Fernández, D., Wagner, S., & Eder, S. Rapid quality assurance with Requirements Smells. *Journal of Systems and Software*, 2017, vol. 123, pp. 190-213. DOI: 10.1016/j.jss.2016.02.047.

12. Dalpiaz, F., Gieske, P., & Sturm, A. On deriving conceptual models from user requirements: An empirical study. *Information and Software Technology*, 2021, vol. 131, article no. 106484. DOI: 10.1016/j.infsof.2020.106484.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

13. Li, Z., Wang, X., Jing, W., Wu, J., Zhang, Z., Liu, Z., Sun, M., Hui, Z., & Liu, S. A Unified Understanding of Deep NLP Models for Text Classification, Available at: https://arxiv.org/abs/2206.09355 (accessed 08.05.2024).

14. Ishizuka, R., Washizaki, H., Tsuda, N., Fukazawa, Y., Ouji, S., Saito, S., & Iimura, Y. Categorization and Visualization of Issue Tickets to Support Understanding of Implemented Features in Software Development Projects, *Applied Sciences*, 2022, no. 12, iss. 7, article no. 3222. DOI: 10.3390/app12073222.

15. Briciu, A., Czibula, G., & Lupea. M. A study on the relevance of semantic features extracted using BERT-based language models for enhancing the performance of software defect classifiers. *Procedia Computer Science*, 2023, vol. 225, pp. 1601-1610. DOI: 10.1016/j.procs.2023.10.149.

16. Abdu, A., Zhai, Z., Algabri, R., Abdo, H., Hamad, K., & Al-antari, M. Deep learning-based software defect prediction via semantic key features of source code systematic survey. *Mathematic*s, 2022, vol. 10, no. 17, article no. 3120. DOI: 10.3390/math10173120.

17. Chawla, P., Hazarika, S., & Shen, H.-W. Token-wise sentiment decomposition for convnet: Visualizing a sentiment classifier. *Visual Informatics*, 2020, vol. 4, iss. 2, pp. 132-141. DOI: 10.1016/j.visinf.2020.04.006.

18. Frattini, J. Identifying Relevant Factors of Requirements Quality: An Industrial Case Study. *Requirements Engineering: Foundation for Software Quality*, 2024. vol. 14588, pp. 20-36. DOI: 10.1007/978-3-031-57327-9_2.

19. Moharil, A., & Sharma, A. TABASCO: A transformer based contextualization toolkit. *Science of Comp*uter *Programming*, 2023, vol. 230, article no. 102994. DOI: 10.1016/j.scico.2023.102994.

20. Sonbol, R., Rebdawi, G., & Ghneim, N. Learning software requirements syntax: An unsupervised approach to recognize templates. *Knowledge-Based Systems*, 2022, vol. 248, article no. 108933. DOI: 10.1016/j.knosys.2022.108933.

21. Berhanu, F., & Alemneh, E. Classification and Prioritization of Requirements Smells Using Machine Learning Techniques. *Conference Proceedings: 2023 International Conference on Information and Communication Technology for Development for Africa* (ICT4DA). IEEE, 2023, pp. 49-54. DOI: 10.1109/ICT4DA59526.2023.10302263.

22. *The Public Jira Dataset*. Available at: https://zenodo.org/records/5901804. (accessed 15.05.2024).

23. Devlin, J., Chang, M., Lee, K., & Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019, vol. 1, pp. 4171-4186. DOI: 10.18653/v1/N19-1423.

24. Grinchenko, M., & Rohovyi, M. A model for identifying project sprint tasks based on their description. *Innovative Technologies and Scientific Solutions for Industries*, 2023, no. 4 (26), pp. 33-44. DOI: 10.30837/ITSSI.2023.26.033.

25. Yuanzhi Li, Bubeck, S., Eldan, R., Del Giorno, A., Gunasekar, S., & Lee, Y. Textbooks Are All You Need II: phi-1.5 technical report. *Microsoft Research*, 2023, pp. 1-16. DOI: 10.48550/arXiv.2309.05463.

26. Nielsen, F. Hierarchical Clustering. *Introduction to HPC with MPI for Data Science, Springer*, Chapter, 2016, ISSN 1863-7310, pp. 221-239. DOI: 10.1007/978-3-319-21903-5_8.

27. Jolliffe, I. T., & Cadima, J. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2016. vol. 374, iss. 2065, pp. 1-16. DOI: 10.1098/rsta.2015.0202.

28. Linderman, G., & Steinerberger, S. Clustering with t-SNE, Provably, *SIAM Journal on Mathematics of Data Science*, 2019, vol. 1, iss. 2, pp. 313-332. DOI: 10.1137/18M1216134.

29. Rohovyi, M., & Grinchenko, M. Project team management model under risk conditions. *Bulletin of the National Technical University "KhPI". Series: Strategic Management, Portfolio, Program and Project Management*, 2023, no. 1(7), pp. 3-11. DOI: 10.20998/2413-3000.2023.7.1.

Received 29.07.2024, Accepted 18.11.2024

НА ШЛЯХУ ДО ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ ПРОЄКТНОЇ КОМАНДИ НА ОСНОВІ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

М. А. Роговий, М. А. Гринченко

Предметом вивчення є метод визначення неякісних описів задач спринту проєкту для зменшення ризику проєкту. **Метою** роботи є зменшення ризиків проєкту, які повязані з якістю опису задач роботи команди проєкту в трекингових системах за допомогою впровадження моделей кластерізації текстових описів

спринту проєкту. **Питання дослідження**: 1. Чи можно за допоомгою кластерзаці визначити неякісні описи задач спринту проєкту? 2. Як використовувати можливоості великих мовних моделей (ВММ) для визначення та покращення текстових описів задач? **Завдання:** аналіз досліджень підходів до опису задач за допомогою методів кластеризації та візуалізації задач проєкту, збір та підготовка текстових описів задач спринту, групування векторних репрезентацій текстових описів задач спринту, дослідження впливу промптінгу на отримання векторних репрезентацій текстових задач, покращення опису задачі за допомогою використання великих мовних моделей, розробка технології підвищення продуктивності проєктної команди на основі ВММ. Використовуваними **методами** є: методи векторної репрезентації текстів, методи зменшення розмірності простору даних PCA та t-SNE, методи агломеративної кластеризації, методи промт інжиніринга. **Отримані такі результати**. Запропоновано підхід до підвищення ефективності проєктної команди на основі використання великих мовних моделей. У відповідь на перше питання дослідження визначено, що немає лингвистичних особливостей, що впливають на сприйняття тестових описів задач спринту проєкту. У відповідь на друге питання дослідження, запропоновано модель визначення потенційно неякісних описів задач, що дозволяє зменшити ризики проєкту, які пов'язані з нерозумінням контексту задачі. **Висновки.** Отримані результати свідчать про те, що описи задач спринту проєкту можна покращити за допомогою великих мовних моделей для розуміння командою проєкту. У подальших дослідженнях рекомендується використовувати вихідну документацію проєкту та контекст проєкту, як векторне сховище та джерело контексту для ВММ. Наступним етапом планується інтеграція ВММ в систему трекінгу задач проєкту.

Ключові слова: проєкт; команда проєкту; опис задач; система управління завданнями проєктів; модель; нейронна мережа; велика мовна модель.

Роговий Микита Антонович – асп. каф. управління проєктами в інформаційних технологіях, Національний технічний університет «Харківський політехнічний інститут», Харків, Україна.

Гринченко Марина Анатоліївна - канд. техн. наук, зав. каф. управління проєктами в інформаційних технологіях, Національний технічний університет Харківський політехнічний інститут, Харків, Україна.

Mykyta Rohovyi – PhD Student of the Department of Project Management in Information Technologies, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine,
e-mail: nikrogovoy@gmail.com, ORCID: 0000-0002-7902-3592.

Marina Grinchenko – PhD of Technical Sciences, Associate Professor, Head of the Department of Project Management in Information Technologies, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine,
e-mail: Marina.Grynchenko@khpi.edu.ua, ORCID: 0000-0002-8383-2675.