IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 May 2024, accepted 7 May 2024, date of publication 13 May 2024, date of current version 21 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3399767

RESEARCH ARTICLE

"Paper, Meet Code": A Deep Learning Approach to Linking Scholarly Articles With GitHub Repositories

PRAHYAT PUANGJAKTHA, MORAKOT CHOETKIERTIKUL, AND SUPPAWONG TUAROB, (Member, IEEE)
Faculty of Information and Communication Technology, Mahidol University, Salaya 73170, Thailand
Corresponding author: Suppawong Tuarob (suppawong.tua@mahidol.edu)

ABSTRACT Computer scientists often publish their source code accompanying their publications, prominently using code repositories across various domains. Despite the concurrent existence of scholarly articles and their associated official code repositories, explicit references linking the two are often missing. Traditionally, identifying whether scholarly content and code repositories pertain to the same research project requires manual inspection, a time-consuming task. This paper proposes a deep learning-based algorithm for automatically matching scholarly articles with their corresponding official code repositories. Our findings indicate that the most common linking information includes the paper title and BibTeX entries, typically found in the repository's readme document. In this study, we employed SPECTER for vector embedding of paper and repository metadata. Utilizing these embedding representations with the Light Gradient Boosting Machine (LGBM), our method achieved an F1 score of 0.94. Moreover, combining our best model with a rule-based approach improved performance by 5.31%. This study successfully delineates a connection between academic papers and associated official code repositories, minimizing reliance on explicit bibliographic information in repositories.

INDEX TERMS Academic code repository mining, paper-repository relationship, text representation, machine learning.

I. INTRODUCTION

The issue of reproducing scientific research, especially achieving consistent results, has been a persistent challenge, extending into the realm of AI research. As pointed out by [1] and [2], replicating AI scientific studies is often hindered by the insufficient detail in AI papers essential for successful replication. Compounding this challenge is the frequent omission of research code in AI studies. Addressing these concerns is vital, given the extensive application of AI research in various fields. To enhance reproducibility, researchers are increasingly utilizing online platforms. Zenodo[1] [3] and GitHub,[2] among others, are preferred for disseminating research materials. In particular, GitHub has gained prominence for sharing and reproducing research findings due to its widespread use among researchers and developers. If the authors of an academic paper publish their implementation in a source code repository, such a paper-repository pair is said to form an *official* relationship [4]. The official relationship indicates that the authors of the corresponding paper publish the GitHub repository, often to distribute the source code used in their published research. The impact of the official relationship extends to individuals seeking to reference authoritative works, as official GitHub repositories often offer more information, reproducibility, reliability, and professionalism [5], [6], [7]. Previously, the traditional way to determine the relationship between a paper and its corresponding GitHub repository was to painstakingly examine the content of both. However, with the advent of platforms like Papers With Code,[3] which explicitly catalogs some papers and their official GitHub repositories, the problem has been partially alleviated. Nevertheless,

The associate editor coordinating the review of this manuscript and approving it for publication was Fu Lee Wang.

[1]https://zenodo.org/
[2]https://github.com/
[3]https://paperswithcode.com/

challenges still persist since such platforms often rely on manual entries by the paper's authors or members, foregoing those paper-repository pairs that have been neglected or simply too new. Furthermore, Papers With Code only houses a subset of papers, particularly focusing on the artificial intelligence topic [8], limiting the coverage to computing research in other domains. However, GitHub serves as the prominent software repository for academic articles within the Paper with Code platform.

GitHub is one of the large-scale code repositories that has been popularized by research communities, both seeking to share their implementation [9] or use its data for software engineering research [10]. In the latter case, researchers have extensively studied GitHub repositories in various contexts. Gousios and Spinellis [11] as well as Chatziasimidis and Stamelos [12] analyzed methodologies for downloading and examining GitHub data, uncovering successful repository rules and behaviors of users that might cause biases in the data. Storey et al. [13] and Cosentino et al. [10] focused on the evolution of software development practices in the context of GitHub. Yu et al. [14] identified four patterns of social behavior on GitHub, including Independence, Group, Star, and Hub. A series of studies [12], [15], [16], [17], [18], [19], [20], [21] explored GitHub's metadata and its influence on repository popularity, delineating four growth patterns which are slow, moderate, fast, and viral. They also sought to understand the factors behind a repository's popularity. Moreover, Al-Rubaye and Sukthankar [22] as well as Zerouali et al. [23] conducted studies focusing on evaluation metrics used on GitHub, where they came to the same conclusion that all these evaluation metrics exhibit a high positive correlation. Aggarwal et al. [24] discovered that the documentation available on GitHub plays a significant role in influencing the popularity of the repositories. Furthermore, the study conducted by Prana et al. [25] delved into the content of readme files, categorizing it into seven distinct types.

In addition to its significance in software development, GitHub's importance in sharing code for research reproducibility has also grown, prompting various inquiries into the dynamics and relationships between scholarly research and its implementation deposited in repositories. Vandewalle [26] and Kang et al. [8] noted that software, as a research artifact, enhances the citation impact of academic papers, leading many researchers to host their software on GitHub and other repositories. Several of these early discoveries have triggered a growing interest in exploring the relationship between academic papers and their official GitHub repositories. For example, recent studies [27], [28], [29] investigated the trend and direction of research software hosted on GitHub, finding a predominance of Python as a programming language and computer science as the main field of study. Wattanakriengkrai et al. [4] delved into how research papers and GitHub repositories are linked, focusing on public accessibility, traceability, and evolution. Their findings indicated that most papers linked to GitHub repositories are publicly accessible, and GitHub remains the preferred platform for online research collaboration. Additionally, they observed that GitHub repositories usually do not experience significant changes after publishing the linked paper. While numerous studies have examined various aspects of GitHub repositories, limited attention has been given to investigating the characteristics of official relationships between papers and these source code archives. Traditional ways to verify the official relationship between a paper and its corresponding official GitHub repository usually require manual inspection, which can become tedious and time-consuming. Therefore, this research aims to automatically identify the official relationship between papers and their corresponding official GitHub repositories using text features extracted from both sources. The main implication of our findings is the ability to discover more pairs of papers and their official GitHub repositories in an automated manner. This is even possible when the paper's or repository's identifiable information is not explicitly stated in both sources. These newly discovered pairs of papers and their published code repositories would be useful in generalizing the findings of the aforementioned studies, covering a broader range of temporal and thematic aspects of computing research.

In this study, we focus on pairs of academic papers and code repositories on GitHub, as the presence of research code repositories on other platforms is limited. This study gathered 19,543 pairs of academic papers and their corresponding official GitHub repositories, as cataloged by Papers With Code. We investigated how academic papers and their official GitHub repositories are interconnected, discovering that roughly 40% of these repositories contain the papers' titles for linking purposes. We also found that roughly 20% of the repositories incorporate the paper's identifiable information in the BibTeX format in their readme files. To determine the official connection between the papers and repositories, we initially examined the similarity of the content in the papers and GitHub documentation as a possible sign of an official link. While a straightforward approach might involve using text similarity metrics like Cosine similarity as detailed by Schutze et al. [30] to compare the textual content of papers and repositories, we found this method alone was insufficient for accurately establishing the official relationship. This led us to apply deep-learning-based techniques to automatically learn the low-level representations of the papers and GitHub repositories. These low-level projected representations enable framing this problem as a classification problem where the official relationship is identified from a pair of a paper and a GitHub repository. Our findings also revealed that the most effective text feature representation is achieved through text embedding using SPECTER [31]. The best features identified for this task include the academic paper's title and a combination of the repository's title and readme file. Our best-performing model scored 0.94 in both F1 and accuracy. Finally, we proposed a hybrid strategy that integrates our

proposed machine-learning-based method with a rule-based method, which yielded a notable 5.31% improvement in performance compared to using the rule-based method alone.

The ability to automatically identify the official GitHub repository for a paper and vice versa could prove valuable in various applications in digital libraries and software engineering domains. For example, researchers seeking the implementation of the algorithms proposed in a paper for extension or baselines can use the system to locate the corresponding repository conveniently, provided that it exists and is publicly accessible, without having to explicitly contact the paper's authors. Furthermore, software developers aiming to adopt a published algorithmic solution in a given GitHub repository could also use the system to find the corresponding paper to further study and quote the relevant experimental results. Our proposed algorithm is highly generalized as it only utilizes publicly available and common information from both the paper and software repository domains, allowing the adoption in various scholarly databases as long as indexed papers have titles and abstracts and software repository services that enable each project to have a textual title and a project-level readme file.

Concretely, this paper presents the following key contributions:

- We recognized the real-world need for the ability to identify official GitHub repositories corresponding to their published academic papers and established a novel research problem of automatic identification of official paper-repository pairs. To the best of our knowledge, we are the first to explore solutions to such a crucial real-world problem.
- We presented an exploratory data analysis (EDA) to examine the relationship between academic papers and their official GitHub repositories.
- We proposed a set of machine learning algorithms to identify official pairs of paper and GitHub repositories by framing the problem as a binary text classification problem where various text representations and classification algorithms were validated via rigorous evaluations.
- We integrated our proposed methodology with a rule-based approach to enhance its performance and discovered that this combination yielded a 5.31% improvement in F1 score.

The rest of this paper is organized as follows. Section II discusses the background and previous related works. Section III describes the details of our dataset. Section IV explains the methodology of this research. Findings and limitations are detailed in Sections V and VII, respectively. Finally, Section VIII concludes this paper.

II. RELATED WORK

Automatically and accurately identifying the official code repository of a scholarly paper can be crucial for researchers and software practitioners alike. Researchers must reproduce proposed algorithmic solutions, while software practitioners require cutting-edge implementations for real-world problems. However, this problem has not been extensively studied; therefore, no solutions have been established. This section will first explore related research that utilizes the relationship between papers and their official code repositories. Furthermore, our proposed solution relies on the ability to intelligently extract representations of papers and GitHub repositories' textual information. Therefore, later in this section, we will provide relevant background on text data representation to help readers better grasp our approach.

A. RELEVANT RESEARCH ON PAPER-REPOSITORY RELATIONSHIPS

Certain areas of study have focused on investigating the connections between GitHub repositories and their associations with research progress. Hasselbring et al. [29] conducted a study on GitHub repositories that serve as placeholders for research software. Their findings revealed that the primary research areas cited by publications referencing GitHub repositories were life science and general science. Moreover, within the field of computer science, the major sub-areas associated with GitHub citations were computer vision and pattern recognition. Similarly, Färber [28] conducted an analysis parallel to Hasselbring et al. [29], focusing on GitHub repositories associated with research papers and examining approximately 3,000 observations. Their analysis revealed that Python was the most commonly used programming language in this context, while the primary fields of study were computer science. Additionally, the TensorFlow framework emerged as a successful machine learning framework in the analyzed GitHub repositories. Another study by Wattanakriengkrai et al. [4] explored GitHub repositories linked to academic papers, specifically investigating aspects of public access, traceability, and evolution. Their investigation provided insights into the relationship between papers and GitHub repositories through explicit linking information, revealing that a majority of papers linked to GitHub were publicly accessible, but it also highlighted that more than 50% of the papers lacked reference information to their corresponding code repositories. This issue aligns directly with the focus of our research, and our proposed approach holds the potential to address this situation effectively. Furthermore, the study unveiled a strong connection between academic communities and GitHub repositories associated with research papers, with approximately 72% of these repositories being developed using Python. It is worth noting that highly cited papers also tended to attract a significant number of citations from other GitHub repositories.

Another study by Rokon et al. [32] focused on distinguishing GitHub repositories through an embedding approach. Specifically, they introduced Repo2vec, a method that utilizes three primary sources of information: GitHub's metadata, repository structure, and source code. By leveraging these inputs, Repo2vec achieved superior performance compared to their baseline models. In a separate investigation,

Shao et al. [33] studied the realm of recommendation tasks, specifically aiming to recommend GitHub repositories based on given paper information. Their proposed method, paper2repo, effectively matches paper information with corresponding GitHub repositories. This approach takes into account both the textual content and the citation graph connecting papers and GitHub repositories. The paper2repo approach demonstrated improved recommendation performance compared to previous methods employed in this context. Therefore, most research in this field focuses on data analysis related to the relationship between academic papers and software repositories. We have summarized the software repository domain, academic article domain, dataset, dataset size, and problem type from existing studies in Table 1.

Most of the aforementioned studies on the relationship between academic papers and their corresponding GitHub repositories were conducted on datasets comprising official paper-repository pairs that were manually verified. Such a dataset is also made publicly available by *Papers With Code*[4] which provides a platform for its users to catalog academic publications along with their relevant artifacts, including GitHub repositories, datasets, and benchmarks. However, the Papers With Code dataset has the following drawbacks that could limit the coverage and generalizability. First, the established linked entities in such a dataset must be cataloged and verified manually by human experts. This manually tedious process would forgo a vast amount of research software repositories that are less popular or simply too new. Second, the Papers With Code dataset only focuses on publications and their corresponding repositories in the machine learning domain,[5] thereby not directly applicable to other study fields that also repose source code as research artifacts [34]. Such needs call for an automated system that is capable of recognizing an official software repository of a given paper and vice versa. This research, therefore, formulates such a problem as a binary classification problem where a pair of a paper and a repository is classified as official or not, where various feature extraction and representation methods, as well as machine learning classification models, could be applied. To the best of our knowledge, we are the first to address this challenging and exciting research question that could give rise to multiple applications in various research domains, including digital libraries and software engineering.

B. TEXT REPRESENTATION

Text representation is a technique for converting textual data into numerical vectors suitable for analysis and computation. Various methods of text representation have been surveyed [35], [36], [37], which could broadly be divided into two categories. The first category, weighted words, involves determining the weight of each word in a document and using these weights as the words' values. Key algorithms in this category include Bag of Words and Term Frequency-Inverse Document Frequency (TF-IDF) [30], which are often used as baselines in research. The second category is text or word embedding. A limitation of the weighted words approach is its inability to capture the semantics of words. To address this, Mikolov et al. [38] introduced word2vec, which uses neural networks to transform words into vectors, capturing their semantic meanings. This area of word embedding has seen substantial growth recently. One notable advancement is the attention mechanism proposed by Vaswani et al. [39], which enables models to focus on specific parts of the input. This mechanism has become highly influential in natural language processing and serves as the foundation for many text embedding models. In the field of academic papers, Cohan et al. [31] introduced the SPECTER model, a text embedding approach based on SciBERT [40]. SPECTER features incorporating academic paper citations to discern paper similarities. Its performance has been notably superior compared to others when used in scholarly contexts. Therefore, in our research, we opt to use the SPECTER model for our text representation due to its effectiveness in representing academic text. In summary, Table 2 presents a comprehensive overview of the existing studies.

III. DATASETS

The main dataset comprises the ground-truth mapping between papers and their official repositories, retrieved from Papers With Code,[6] a platform for curating data about academic papers paired with their corresponding GitHub repositories in the artificial intelligence (AI) area. Each mapping includes identifiable information about a paper (i.e., title, year, and list of authors) and its official GitHub repository (i.e., GitHub URL). This dataset significantly simplifies the process of acquiring academic papers and GitHub repositories that share an official relationship. Our data acquisition took place on June 6, 2021.

Since our methodology requires textual information such as the title and readme file extracted from each GitHub repository, we directly accessed and downloaded such repository metadata using the GitHub API. Furthermore, since Papers With Code does not provide complete metadata of a paper, we utilized the S2AG dataset [41], which offers additional details on academic papers necessary for exploratory data analysis. The paper titles were used as keys to link the paper entries from Papers With Code and S2AG datasets.

The dataset utilized in this study comprises data linking academic papers with their corresponding GitHub repositories. Following pre-processing, detailed in Section IV-B, we collected a total of 19,543 academic research records that exhibit an official relationship between academic papers and their associated GitHub repositories. This official relationship indicates that the author or organization associated with the academic paper is the same as that linked to the GitHub repositories. To utilize machine learning to

[4]https://github.com/paperswithcode/paperswithcode-data

[5]https://paperswithcode.com/about

[6]https://paperswithcode.com

TABLE 1. Comparative summary of the literature on the paper-repository relationship.

Reference	Software Repository Domain	Academic Article Domain	Dataset	Dataset Size	Problem Type
Färber et al. [28]	GitHub	Microsoft Academic	GitHub, Microsoft Academic Graph	2,955	Exploratory Data Analysis
Hasselbring et al. [29]	GitHub	ArXiv, ACM	GitHub	5,000	Exploratory Data Analysis
Shao et al. [33]	GitHub	Microsoft Academic	GitHub, Microsoft Academic API	2,107	Recommendation System
Wattanakriengkrai et al. [4]	GitHub	IEEE, ACM, ICSE, JSS, IST, EMSE	GHTorrent	20,278	Exploratory Data Analysis

TABLE 2. Comparative summary of the literature on text representation.

Reference	Approach	Technique	Advantage	Disadvantage
Manning et al. [30]	Word Weighting	TF-IDF	Effectively mitigates the impact of frequently occurring terms within documents.	Fails to reflect word-level synonymy and polysemys
Mikolov et al. [38]	Word Embedding	Word2Vec	Enables the measurement of semantic similarity between words.	High complexity and computation
Cohan et al. [31]	Document Embedding	SPECTER	Enables document-level embedding, offering similarity measures and seamless integration with downstream applications.	High complexity and computation

identify official relationships between academic papers and GitHub repositories, the dataset was segmented into training and testing sets. The testing set is further divided into two subsets. The initial subset supports research questions 2 and 3, while the second subset is dedicated to the hybrid methodology outlined in research question 4. The first testing subset includes 233 academic research entries missing linking information, such as titles, URLs, or BibTeX entries, in either the papers or GitHub repositories. The testing subset for the hybrid methodology merges this group with 20% of the residual data, totaling 3,862 observations, which may include reference information linking scholarly articles to their official code repositories. The training set consists of the remaining 15,448 observations.

TABLE 3. Descriptive statistics of the paper and GitHub repository datasets.

Number of Words	Min	Median	Max	Average	Std.
Paper Title	1	9	29	9.33	3.13
Paper Abstract	68	166	346	169.8	47.27
Repository Title	1	2	15	1.94	1.33
Repository Readme	33	401	29,602	586.08	763.66

Furthermore, we conducted a statistical analysis of the word size for each feature in our dataset. The titles of the papers have a minimum length of one word, with median and mean lengths of approximately nine words. The maximum length observed in paper titles is 29 words, with a standard deviation of approximately three words. Regarding abstracts, the dataset shows a minimum of 68 words and a maximum of 346 words, with median and average lengths of 166 and 169.8 words, respectively. The standard deviation for abstracts, at 47.27, exceeds that of paper titles. Moreover, our analysis revealed that the word size of academic papers' titles and abstracts follows a normal distribution.

Considering the GitHub repositories, their titles show a distribution similar to that of paper titles, with the same minimum length. However, the median and mean lengths of GitHub titles are around two words, with a maximum length of 15 words and a standard deviation of 1.33 words. On the other hand, the readme sections do not follow a normal distribution. They have a minimum, median, mean, maximum, and standard deviation of 33, 401, 586.08, 29,602, and 763.66 words, respectively. Consequently, GitHub titles typically have shorter lengths than paper titles. Researchers generally use around two words for GitHub titles. However, the readme sections vary widely in length due to the absence of specific writing rules, resulting in a high standard deviation of approximately 763 words. This indicates that some readme sections contain extensive text while others contain relatively little text. Ultimately, readme sections tend to contain more text than the abstracts of corresponding academic papers. The descriptive statistics for this dataset are presented in Table 3.

Additionally, we examined the most frequently occurring tasks in academic papers. Given that Paper With Code lists over 4,000 tasks, we focused on the 12 most prevalent ones in the dataset. Figure 1 displays the proportion of the top 12 tasks in academic papers across both the training and testing sets. Within our dataset, Name Entity Recognition and Classification emerge as the most common tasks in scholarly articles, followed by Translation, Semantic Segmentation, and Reinforcement Learning as secondary preferences. Additionally, Object Detection, Representation Learning, Question Answering, Language Modeling, and Transfer Learning exhibit comparable distribution. The two least common tasks are Frame, which concerns frame studies

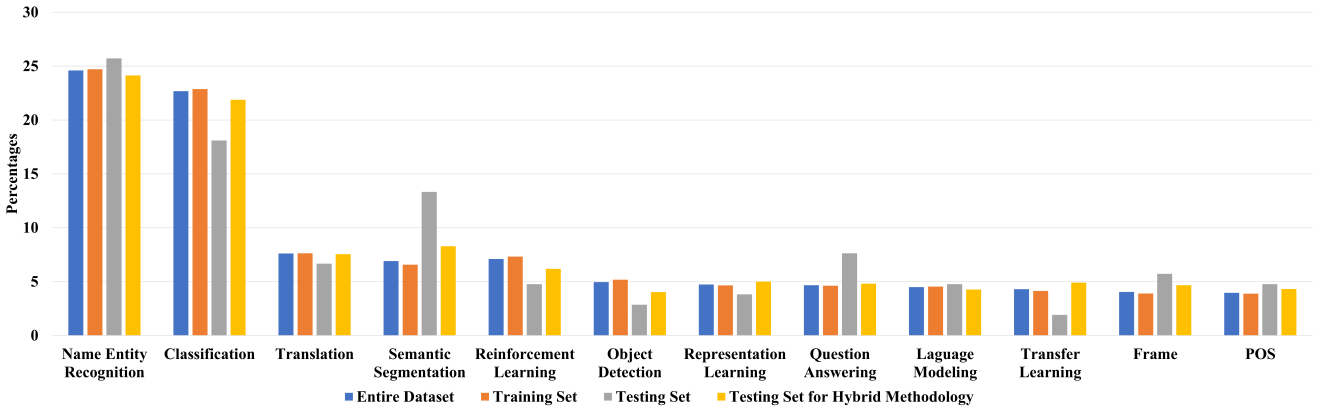


FIGURE 1. Diversity and representation of the dataset.

in videos, and POS, which is related to parts of speech in Natural Language Processing (NLP). After partitioning the data into training and testing sets, the distribution of these 12 tasks stays consistently comparable and closely resembles that of the complete dataset obtained from Paper With Code. This indicates that our dataset is representative of the overall distribution.

IV. METHODOLOGY

This section provides a detailed explanation of the methodology employed to address our research problem. Figure 2 illustrates the step-by-step process used in this study.

A. RESEARCH QUESTIONS

The problem of automated identification of official code repositories for academic papers is novel. Solving such a challenging problem could also open doors to various downstream novel contributions. This research particularly emphasizes systematically answering the following research questions.

1) Can the similarity of documents imply an official connection between papers and corresponding GitHub repositories? This research question investigates whether the similarity between a paper and a GitHub repository alone is sufficient for identifying their official relationship.
2) What is the optimal representation for repositories and papers in order to identify their official relationships? The textual information from the papers and repositories must be represented as numerical vectors so that traditional machine-learning classification methods can be applied. This research question aims to empirically determine the optimal representation techniques for such textual data. These techniques include TF-IDF term weights, Binary term weights, and context embedding using SPECTER.
3) How do different scopes of repository and paper information impact the ability to identify their official relationships? As we have four types of features to consider (i.e., paper title, paper abstract, repository title, and repository readme), these features may contain both valuable information and noise. Utilizing the full combination of these features may not be an optimal approach. Hence, this research question aims to identify the most suitable combination of feature components that yields optimal prediction performance.
4) Can our proposed machine learning-based approach be integrated with the traditional rule-based method to improve the overall performance? A simple approach, such as checking the official relationship between a paper's title and a repository's readme file, would seem reasonable. However, we discovered that such a rule-based method, though highly precise, misses out on some repositories that do not mention their corresponding papers' titles in the readme files, resulting in low recall. Therefore, we aim to investigate whether our approach can enhance the detection performance between official academic papers and corresponding GitHub repository pairs when integrated with the traditional rule-based approach.

B. DATA PRE-PROCESSING

In this study, data pre-processing was divided into three primary stages. The initial step involves the removal of linking information. Subsequent steps encompass the elimination of missing values and noisy data. Finally, a sequence of text-processing steps was applied.

We aim for our approach to perform effectively even in cases where explicit linking information is absent from a paper or a repository. As a result, we removed explicit linking information from our dataset, specifically targeting two features where this information is commonly found: abstracts and readme files. The linking information typically found in abstracts may include GitHub repository titles, while readme files may contain paper titles, paper URLs, and BibTeX entries. Initially, we eliminated GitHub repository titles from the abstracts and removed paper titles and paper URLs from

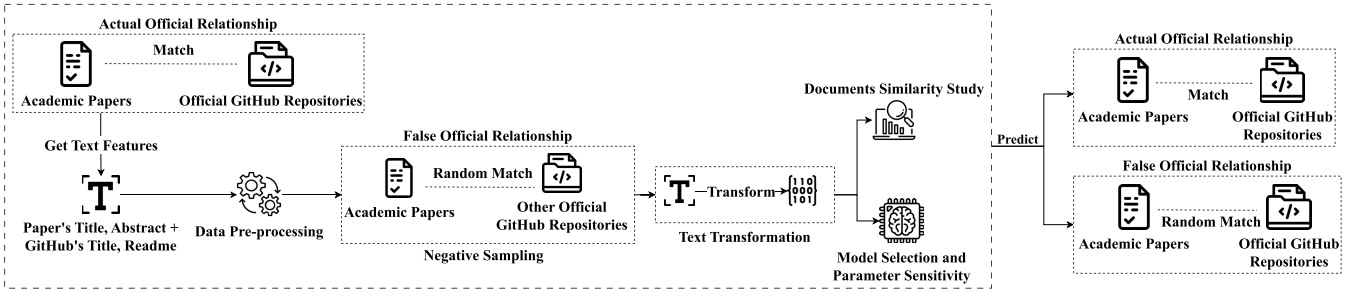


FIGURE 2. High-level methodology of this research.

the readme files. Notably, BibTeX entries are bibliographic references used to cite academic papers, and researchers often include them in readme files to reference their associated academic papers within GitHub repositories. To eradicate this information, we employed regular expressions matching the BibTeX pattern and subsequently removed them from the readme files. This meticulous process enables our approach to consider the genuine context between academic papers and their associated GitHub repositories. Consequently, our approach is robust and accommodates scenarios where the explicit interrelationship between academic papers and GitHub repositories is absent.

Another aspect of data pre-processes involves the elimination of missing values and noise. A major cause of missing values came from the limitations of Papers With Code's data, where some papers' information is missing. To handle missing values in our dataset, we simply remove entries with missing data. Furthermore, the removal of noise is a crucial step in this process. To achieve this, we employ box plots to identify noisy values for each feature. Subsequently, we eliminate data points with values falling below the defined threshold as determined by the box plots. In cases where the features do not follow a normal distribution, we apply a log transformation before calculating the box plots. This transformation allows us to identify and remove data points with values less than the threshold.

Finally, the data underwent several text pre-processing steps to prepare it for analysis and experimentation. Firstly, we removed special characters from the dataset that were present in readme files. Then, we eliminated stop words such as ''a'', ''an'', and ''the'', as they do not carry significant meaning. Finally, we performed lemmatization, which involves converting words into their root forms. For instance, ''is'' would be converted to ''be''. By performing these steps, we could render our text data suitable for further analysis and experimentation.

C. NEGATIVE SAMPLING

The dataset utilized in this research exclusively consists of official (positive) pairs comprising papers and their corresponding GitHub repositories, as mentioned in Section III. In order to perform data analysis and machine learning tasks, it is necessary to have both positive and negative samples. In this context, a positive sample refers to the official pair of a paper and its corresponding GitHub repository. To generate negative data, we employed a random switching technique that involves interchanging the pairs between papers and GitHub repositories. Specifically, for each paper, we randomly associate it with a GitHub repository that is not its official repository.

D. TEXT TRANSFORMATION

This study used textual data from academic papers and GitHub repositories to determine their official relationship. However, the raw textual format of the data is not suitable for effective data analysis or machine learning methods. We implemented a text representation strategy whereby we individually transformed the text from the academic paper and its corresponding GitHub repository. The main goal of this transformation process was to change the text into numerical vectors that could effectively represent the underlying meaning. Following this, given a paper-repository pair, their vector representations are concatenated to jointly represent the pair.

Three text representation techniques were explored: TF-IDF, binary, and SPECTER embedding. Term-Frequency Inverse-Document-Frequency (TF-IDF) [42] is a widely employed method in text classification that represents a document as a vector of term weights, each of which is computed from both the word's frequency and meaningfulness. In contrast, the binary representation is a simplified version of TF-IDF, where if a term appears in the document, its weight is 1, and 0 otherwise. The binary representation indicates the presence or absence of words in the text. The other approach involves using deep learning techniques to compute contextual embedding for each document. This study used a pre-trained model called SPECTER [31] to embed documents. This SPECTER model was chosen since it was pre-trained on a vast corpus of academic papers, which is similar to our data.

Considering the dataset's extensive vocabulary, all the previously mentioned text transformation techniques generate large vectors. These large vectors can lead to the curse of dimensionality when used with machine learning

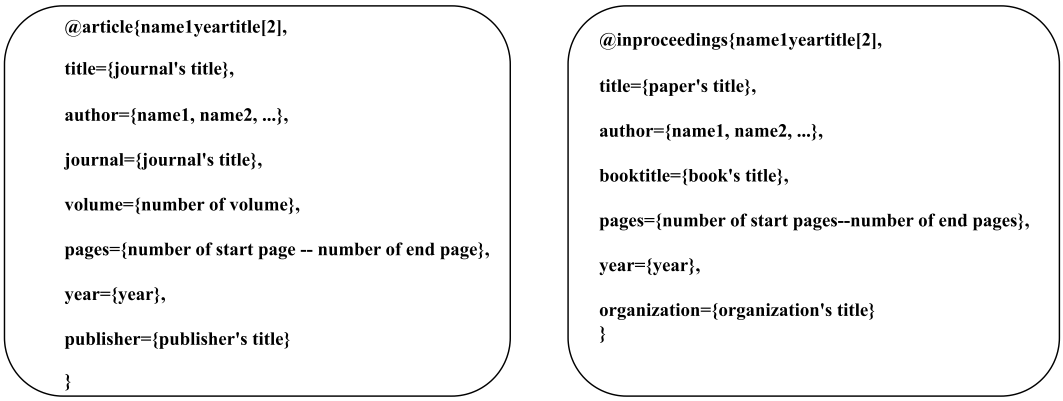


FIGURE 3. Example BibTeX entries in academic research.

models [43]. To tackle this issue, we employed Principal Component Analysis (PCA) [44] to summarize and reduce the dimensionality of the text representations. However, multiple configurations of PCA were evaluated to determine the optimal number of dimensions for our subsequent analyses.

E. TEXT SIMILARITY MEASUREMENT

Our method aims to determine the official relationships between academic papers and GitHub repositories. The textual information from both the papers and repositories is extracted and used to represent them. A natural question may arise as to whether the similarity between a paper's and a repository's textual representations can sufficiently infer their official relationship. Investigating in this direction also answers the first research question raised.

$$Cosine(x, y) = \frac{x \cdot y}{|x||y|} \tag{1}$$

To accomplish this, we utilize cosine similarity [30], illustrated in Equation 1, to compute the similarity score between a paper and a GitHub repository. Subsequently, we analyzed the distribution of similarity scores between the positive and negative sample groups. If the distributions exhibit clear separation, it suggests that document similarity can effectively discern the official relationship between academic papers and their associated GitHub repositories. Conversely, if the distributions overlap, it implies that document similarity may not be discriminative enough to identify the official relationship between academic papers and GitHub repositories.

F. MODEL SELECTION

Due to the intricate nature of the problem, creating solutions based on rules is difficult, requiring the use of machine learning techniques. In order to accomplish this, it is necessary to partition the data into separate sets for training and testing purposes. First, we identified pairs of scholarly articles and corresponding GitHub repositories that lacked any explicit connections. Such a dataset was preserved as the testing set and utilized to assess the machine learning models. The rationale behind utilizing paper-repository pairings devoid of any connection information as the test set is to assess the efficacy of our machine learning-based techniques on challenging scenarios. If specific linking information, such as paper bibliometric information, is included in the corresponding repository's readme file, it is possible to readily establish the formal relationship between a paper and a repository.

The training set, used to train machine learning models, consists of the remaining samples obtained after removing the testing set. Given that our dataset exclusively consists of positive data, referring to pairs of papers and corresponding GitHub repositories with official connections, we synthesized additional negative pairs to supplement the training set, as explained in Section IV-C.

In this study, we have employed a diverse range of machine learning algorithms [45], encompassing various families of supervised learning techniques, including linear models, decision trees, neural networks, support vector machines [46], and Bayesian classifiers. Specifically, for linear models, Bayesian classifiers, neural networks, and support vector machines, we utilized logistic regression (LR), multinomial naive Bayes (MLB), multi-layer perceptron (MLP), and linear support vector machine (LSVM), respectively. Additionally, we also explored ensemble tree-based algorithms in this research, such as Random Forest (RF) [47], Gradient Boosting (GB) [48], XGBoost (XGB) [49], and LightGBM (LGBM) [50]. These ensemble algorithms have demonstrated their effectiveness across various problem domains [51], [52].

Furthermore, we configured these models as binary classifiers, training them to predict the official relationship between pairs of academic papers and their corresponding GitHub repositories. To evaluate the performance of these models, we employed common evaluation metrics typically used in binary classification problems, including accuracy, precision, recall, and F1 score. Consequently, we conducted a comprehensive comparison of the performance of all these models and selected the best-performing one based on

the aforementioned metrics for further parameter sensitivity analyses.

G. PARAMETERS SENSITIVITY

This section outlines the experimental procedure aimed at understanding how varying certain feature configurations impact its classification performance. This section aims to address research questions 2 and 3. Accordingly, we explored the effects of varying factors, such as different combinations of text representation techniques and inclusions of different textual components of papers and repositories.

Representing textual data in vectorized formats allows machine learning classification algorithms to directly interpret and generate prediction models. However, different text representation techniques are distinct in terms of the semantics of the original text that they can encode. To determine the most effective text representation methods, we evaluated each representation with various machine learning models since different learning algorithms' performance may depend on the representation of the data.

In addition to different algorithms employed to represent textual information, we also investigated the combination of various textual components from papers and GitHub repositories. These components include the paper's title, abstract, GitHub's title, and readme files. While it may seem that combining all of these components could lead to the best results, such complete inclusion could also introduce noise and irrelevant information that could hinder the model's performance. Therefore, it is important to identify the combinations of these textual components that yield the model's optimal performance. To achieve this, we explored all possible combinations between the papers' and repositories' textual components. Moreover, we applied the best text representation method identified previously during this parameter exploration. Finally, we compared the performance of all machine learning models and analyzed the textual combinations that contributed to achieving the optimal performance.

V. EXPERIMENTAL RESULTS

This section first discusses the data exploration followed by experiments conducted particularly to address the research questions raised in Section IV-A. All the experiments were conducted in a Linux machine with a 32-core CPU (64 threads), 128 GB of RAM, and an RTX 3090 graphic card. In addition, Python was used to implement the code and experimental framework for this research.

A. DATA EXPLORATION

Automated discovery of official GitHub repositories for research articles is crucial for various applications in digital libraries and software engineering that involve implementation artifacts of newly proposed algorithmic inventions, and vice versa. This section explores the characteristics of the relationship between academic papers and their official corresponding GitHub repositories.

This study identified various aspects of academic papers and their associated repositories. These aspects include linking information in the form of titles or URLs within academic papers or GitHub repositories. Our initial focus was on exploring the linking information present in academic papers. We discovered that approximately 3%, or roughly 630 academic papers, only included URLs directing to their official GitHub repositories within their abstracts. Furthermore, it was evident that the academic papers' titles often differed from those of their corresponding official GitHub repositories. Additionally, we delved into the linking information within GitHub repositories that referred back to their academic papers. The linking practices in GitHub repositories can exhibit significant variation, as GitHub does not impose strict rules governing the inclusion of such information. This freedom allows researchers to present this information in diverse ways. Our findings revealed that around 2.59%, or roughly 506 GitHub repositories, only incorporated the titles of their papers in their descriptions. However, this feature was not extensively utilized, possibly due to the brevity of GitHub descriptions. Consequently, most linking information within GitHub repositories was concentrated in the readme files, which serve as the primary documentation for repositories. Typically, researchers include three pieces of linking information within the readme files: BibTeX, the paper's title, and URLs to the paper's PDF version. In this study, we observed that 40.52%, corresponding to 7,977 of the official GitHub repositories, solely featured the titles of their associated papers within the readme files. In contrast, the inclusion of only BibTeX information was observed in about 2% or 414 repositories. Furthermore, only 0.03% of the repositories contain URLs to their papers. Table 4 provides a comprehensive breakdown of the percentages for each type of linking information between academic papers and their associated official GitHub repositories.

TABLE 4. Types of linking information between academic papers and the corresponding GitHub repositories.

Types of Linking Information	Percent	#Rows
Repository URL in paper abstract	3.22%	630
Paper title in repository readme	**40.82%**	**7,977**
Paper BibTeX information in repository readme	2.12%	414
Paper URL in repository readme	0.03%	5
Paper title in repository description	2.59%	506
Total	**48.77%**	**9,532**

In practice, researchers often include multiple forms of linking information to explicitly show the connection between their papers and the repositories reposing their implementation. Therefore, we conducted an exploration of academic papers and GitHub repositories that featured more than one type of linking information, which we refer to as "bi-linking information." Figure 4 visually represents the prevalence of bi-linking information between academic papers and GitHub repositories. Our experiments

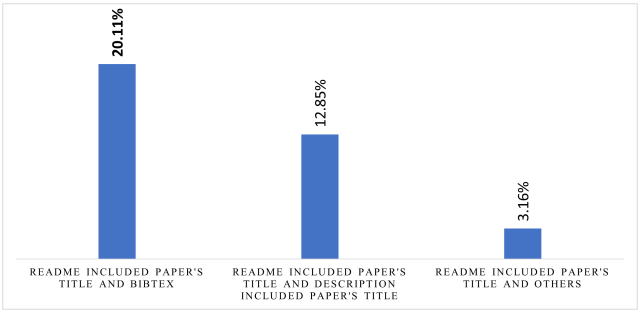


FIGURE 4. Statistics of different bi-linking information between academic papers and the corresponding GitHub repositories.

revealed several patterns in bi-linking information. Notably, we found that readme files frequently contained both BibTeX information and the paper's title, constituting the highest percentage at 20.11%. Another common scenario involved the paper's title appearing in both the readme and the description, accounting for 12.85% of the repositories. Additionally, merging the paper's title with other forms of linking information was observed in 3.16% of the cases.

During our investigation, we determined that the paper's title stands out as the most significant form of linking information, frequently present in readme files. Another noteworthy source of linking information is BibTeX, which houses critical academic paper details. Both of these types of information can typically be found within the readme files of official GitHub repositories. Our analysis revealed that approximately 48% of GitHub repositories employed a single type of linking information, while around 36% utilized bi-linking information, including the paper's title and BibTeX. Intriguingly, we also identified 233 repositories that lack any form of linking information. These repositories will serve as our test set in our experiments, allowing us to evaluate the performance of our approach in detecting paper-repository pairs that lack any linking information—a crucial step in enhancing the capabilities of automated systems.

Furthermore, we analyzed the relationship between academic papers' titles and their corresponding official repositories. This comparison is detailed in Table 5. Our research identified five common practices adopted by researchers when naming their code repositories. Firstly, some repositories directly adopt the title of their academic papers, facilitating immediate identification of the associated scholarly work (Type = Same). Secondly, researchers may devise a model name to serve as the repository title, enhancing the memorability of their project (Type = Model Name). Additionally, authors may use the initial character from each word in the paper's title to form an abbreviation for the repository title (Type = Paper Title Abbreviation). Another observed practice is naming the repository by using the paper's publication venue (Type = Venue). Finally, some academic research exhibits inconsistency between the paper title and the repository name, making it challenging to establish an explicit connection without examining the content of both artifacts. In such cases, repositories are often named using abbreviations derived from the paper's research task (Type = Paper's Tasks).

B. EXPERIMENT RESULTS

This section reports the experiment results and provides relevant analyses and discussions as guided by the raised research questions.

1) RESEARCH QUESTION 1: CAN THE SIMILARITY OF DOCUMENTS IMPLY AN OFFICIAL CONNECTION BETWEEN PAPERS AND CORRESPONDING GITHUB REPOSITORIES?

Our goal in this section is to examine the correlation between the similarity of academic paper titles and abstracts with GitHub repository titles and readme contents. Visualizing plots of the similarity distribution for each text representation used in our research were used to explain the results. If a significant difference in similarity between official and non-official pairs is observed, it would imply that the similarity between academic papers and GitHub repositories alone is sufficient to indicate their official relationship.

After calculating the similarity scores between paper-GitHub pairs using cosine similarity, we compared the distribution of these scores across different text representations and relationships, as illustrated in Figure 5. In the figure, the orange color represents the similarity distribution of the official relationship (positive samples), while the blue color represents the distribution for the non-official relationship (negative samples).

Both TF-IDF and binary representations exhibit a similar distribution trend, with values mostly in the low range from 0 to 0.2. On the other hand, SPECTER's text embedding displays a right-skewed distribution, with values ranging from 0.85 to 0.95. However, regardless of text representation methods, all these distributions have a significant overlap between official and non-official relationships. This overlapping area makes it difficult to determine an optimal threshold that can effectively distinguish the relationship between academic papers and GitHub repositories based solely on similarity scores. As a result, utilizing solely text similarity is not a reliable method to identify official code repositories for papers. Therefore, this limitation highlights the need for statistical techniques to learn the papers' and repositories' representations to determine their official relationship.

2) RESEARCH QUESTION 2: WHAT IS THE OPTIMAL REPRESENTATION FOR REPOSITORIES AND PAPERS IN ORDER TO IDENTIFY THEIR OFFICIAL RELATIONSHIPS?

Upon discovering that merely quantifying the similarity between academic publications and their related GitHub repositories was insufficient, we investigated machine-learning-based text classification as an alternate method to differentiate between official and non-official connections by framing this problem as a binary classification task.

Different text representations, i.e., TF-IDF, binary, and embedding, were experimented with textual information

TABLE 5. Examples of academic papers and corresponding official repositories.

Academic Paper Title	GitHub Title	GitHub URL	Type
HexaConv	HexaConv	https://github.com/ehoogeboom/hexaconv	Same
Meta-Q-Learning	Meta-Q-Learning	https://github.com/amazon-research/meta-q-learning	Same
AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles	AirSim	https://github.com/Microsoft/AirSim	Model Name
XLNet: Generalized Autoregressive Pretraining for Language Understanding	xlnet	https://github.com/zihangdai/xlnet	Model Name
Deep Transferring Quantization	dtq	https://github.com/xiezheng-cs/dtq	Paper Title Abbreviation
Collaborative Deep Reinforcement Learning	cdrl	https://github.com/illidanlab/cdrl	Paper Title Abbreviation
When can Multi-Site Datasets be Pooled for Regression?	ICML2017	https://github.com/hzhoustat/icml2017	Venue
Evolving Spatially Aggregated Features From Satellite Imagery for Regional Modeling	ppsn_2016	https://github.com/skriegman/ppsn_2016	Venue
Exploiting Structure for Fast Kernel Learning	gp_grid	https://github.com/treforevans/gp_grid	Paper's Tasks
Off-Policy Deep Reinforcement Learning without Exploration	bcq	https://github.com/sfujim/bcq	Paper's Tasks

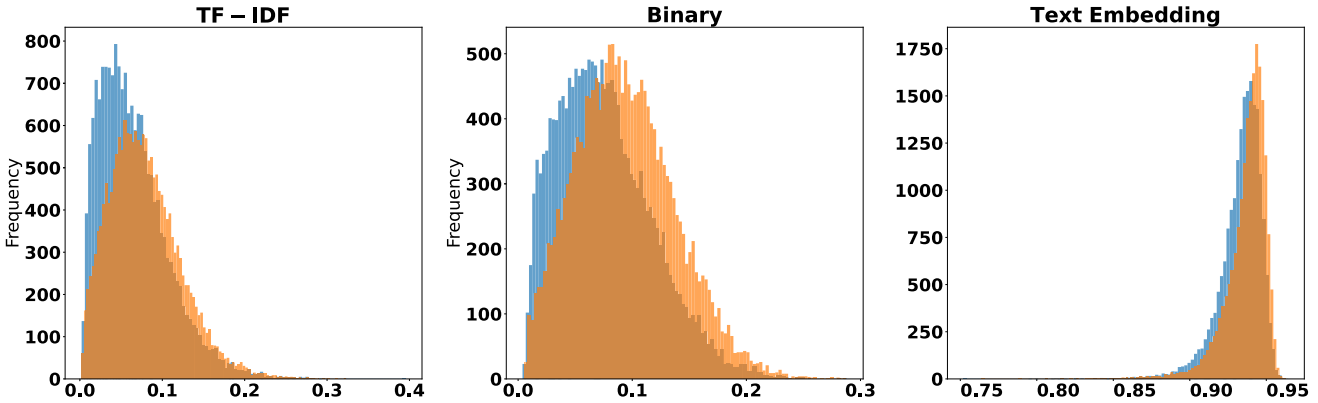


FIGURE 5. Comparison of text similarity distribution, using different representations, between official and non-official paper-repository relationships.

TABLE 6. Comparison of model performance when varying different text representations of papers and repositories.

Paper Representation	GitHub Representation	Classifier	Accuracy	Recall	Precision	F1
TF-IDF	TF-IDF	ALL	0.5	0.5	0.25	0.33
Binary	Binary	ALL	0.5	0.5	0.25	0.33
Text Embedding	Text Embedding	MLB	0.85	0.85	0.87	0.85
TF-IDF + PCA	TF-IDF + PCA	ALL	0.5	0.5	0.25	0.33
Binary + PCA	Binary + PCA	ALL	0.5	0.5	0.25	0.33
Text Embedding + PCA	**Text Embedding + PCA**	**RF and LGBM**	**0.92**	**0.92**	**0.93**	**0.92**

retrieved from both papers and repositories. Each combination of paper and repository representations was validated with a set of machine-learning classification algorithms. Table 6 highlights the best results for each combination of text representations. Note that "ALL" signifies that all of the machine learning models produced identical results.

The results in Table 6 are divided into two parts which are with and without PCA. In the first part, we observed that both TF-IDF and Binary text representations yielded identical results in terms of accuracy, recall, precision, and F1 score. This outcome was due to the limitation of these representations in capturing the essential characteristics of the text in this problem. Additionally, all models employing these text representations assumed that all inputs corresponded to official relationships, yielding an accuracy, recall, precision, and F1 score of 0.5, 0.5, 0.25, 0.33, respectively. These results indicate that the model struggles to accurately distinguish between true and false official relationships, as evidenced by a significant number of false negatives and false positives, as reflected in the recall and precision metrics. However, representing papers and repositories with text embedding and using Multinomial Naive Bayes (MLB) as the classifier yielded the best performance, with an accuracy, precision, recall, and F1 score of 0.85, 0.87, 0.85, and 0.85, respectively. This demonstrates the enhancement in true positives and true negatives achieved by our model, indicating that text embeddings more effectively capture the semantics of text features than TF-IDF and binary representations.

The bottom three rows in Table 6 show the results where Principal Component Analysis (PCA) was applied to all text representations to reduce dimensionality. We found that TF-IDF and Binary struggled to effectively learn the text representations. Consequently, the results of these representations after applying PCA remained the same as

in the previous section. Text embedding with PCA showed significantly improved performance compared to individual models. We identified two machine learning models, Random Forest (RF) and LightGBM (LGBM), as achieving the highest performance, with both models reaching an accuracy and an F1 score of 0.92 and a precision of 0.93. Notably, the integration of PCA with the optimal text representation method resulted in an 8.2% improvement in the F1 score for text embedding techniques. Utilizing PCA enhanced the capability of the text representation to capture features, thereby improving the model's prediction of true positives and true negatives more effectively. This improvement is attributed to the reduction in the embeddings' dimensionality, making them more suitable for machine learning algorithms. Therefore, the combination of text embedding representation with PCA was selected for further experiments.

Furthermore, we employed t-SNE [53] to visualize all the projections of positive and negative samples using different text representation approaches. When dealing with high-dimensional vectors, it can be difficult to visualize them. However, t-SNE can be useful when interpreting features' discriminative power. Figure 6 presents the t-SNE visualizations of both the positive (yellow) and negative (purple) samples in two dimensions. Notably, TF-IDF representation appears to be the most scattered among the data points, making it challenging to differentiate between official and non-official relationships. The binary representation performs better than TF-IDF, with some data points forming distinct groups, yet most of the samples still do not form clear clusters. This lack of distinction is a key factor in the poor performance of both representations.

In contrast, when using SPECTER for text embedding, positive and negative data points seem to establish more well-defined cluster boundaries, as we can observe that certain portions exhibit a clear separation between purple and yellow data points. Consequently, these t-SNE visualizations could serve as illustrative evidence supporting the previous experiment's conclusion, where SPECTER embedding is the optimal method to represent the papers and repositories. In addition, after applying PCA to the text embedding vectors generated by SPECTER, we reduced noise and improved our approach's performance to 0.9 for all evaluation metrics.

3) RESEARCH QUESTION 3: HOW DO DIFFERENT SCOPES OF REPOSITORY AND PAPER INFORMATION IMPACT THE ABILITY TO IDENTIFY THEIR OFFICIAL RELATIONSHIPS?

In the previous section, we conducted experiments by incorporating textual components from papers (i.e., titles and abstracts) and repositories (i.e., titles and readme files) into the feature vectors. However, using all these components might not be optimal, as it can introduce noise and useless information into the model. Hence, we explored all possible feature combinations of papers' and repositories' text components using the best text representation from the previous section (i.e., SPECTER embedding). Each combination was evaluated with various classification algorithms, where the best algorithm is reported. Table 7 highlights these results.

Considering features from the GitHub repositories (the 2nd column), we observed that repository titles are of less significance when compared to the readme files. This can be attributed to the fact that GitHub repositories' titles are typically short and may utilize abbreviated versions of the paper's title or contain specific words not directly related to the paper's content. Consequently, using only GitHub's titles for text embedding may not accurately represent the data, resulting in scores of less than 0.6 in all evaluation metrics, regardless of the papers' features.

Higher performance was achieved when using the readme files as a text embedding feature. This is because readme files typically contain a wealth of valuable information that succinctly describes the project, some of which can resonate with the title and abstract of its corresponding paper. Therefore, using only the readme file to represent a repository yields scores exceeding 0.9 for all evaluation metrics. While the information from the readme files alone suffices to train an accurate model, we found that the combination of repository title and readme file (i.e., `Title + Readme`) yielded the best performance when paired with the paper title (i.e., `Title`).

Considering different combinations of paper features (i.e., `Title`, `Abstract`, and `Title + Abstract`), we observed that the academic paper's title holds more significance than the abstract. Typically, researchers choose the title carefully, including only essential words directly related to the academic paper's content, such as methodology, problem, or solution. Some of these specific words may be utilized in the corresponding repository's readme files. However, when we used the academic paper title and GitHub's title as text embedding features, our machine learning model struggled to establish a clear relationship between the two. Regarding the abstract, it serves as a concise summary of an academic paper and often contains more valuable information compared to the academic paper title, which is akin to a readme file; therefore, combining the abstract with any GitHub readme files resulted in excellent performance, with scores exceeding 0.9 for all evaluation metrics. Surprisingly, we found that using only the abstract as a text embedding feature from the paper domain led to lower performance compared to using only the academic paper title. Furthermore, combining both the academic paper title and abstract resulted in lower performance than using only the academic paper titles alone. This discrepancy arises because the abstract, while informative, may also contain noise and superficial information summarized from the paper's full content, which could explain why using only academic paper titles yielded slightly superior performance compared to the combined (`Title + Abstract`) approach.

As a result, we determined that the optimal feature combination for text embedding, encompassing both paper and GitHub domains, involves exclusively using the academic paper title and combining both the GitHub repository's title

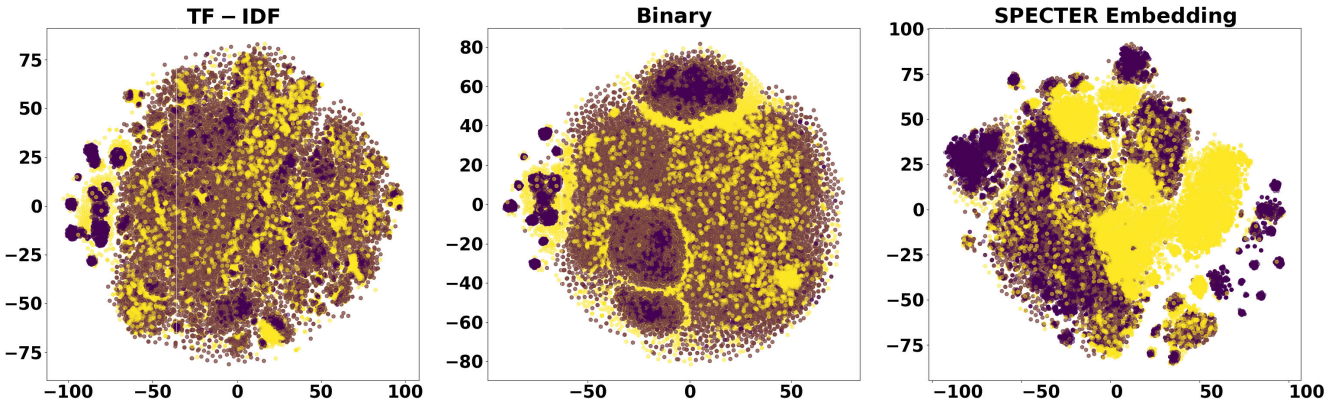


FIGURE 6. t-SNE visualization of positive and negative data points using different text representation methods.

TABLE 7. Comparison of the model performance when trained with different combinations of papers' and repositories' text components.

Paper's Features	GitHub's Features	Classsifier	Accuracy	Recall	Precision	F1
Title	Title	LR	0.57	0.57	0.59	0.55
Title	Readme	Linear SVC	0.9	0.9	0.91	0.9
Title	**Title + Readme**	**LGBM**	**0.94**	**0.94**	**0.94**	**0.94**
Abstract	Title	LR	0.54	0.54	0.56	0.48
Abstract	Readme	MLP	0.93	0.93	0.93	0.93
Abstract	Title + Readme	LGBM	0.92	0.92	0.93	0.92
Title + Abstract	Title	LR	0.55	0.55	0.57	0.52
Title + Abstract	Readme	MLP	0.93	0.93	0.93	0.93
Title + Abstract	Title + Readme	RF and LGBM	0.92	0.92	0.93	0.92

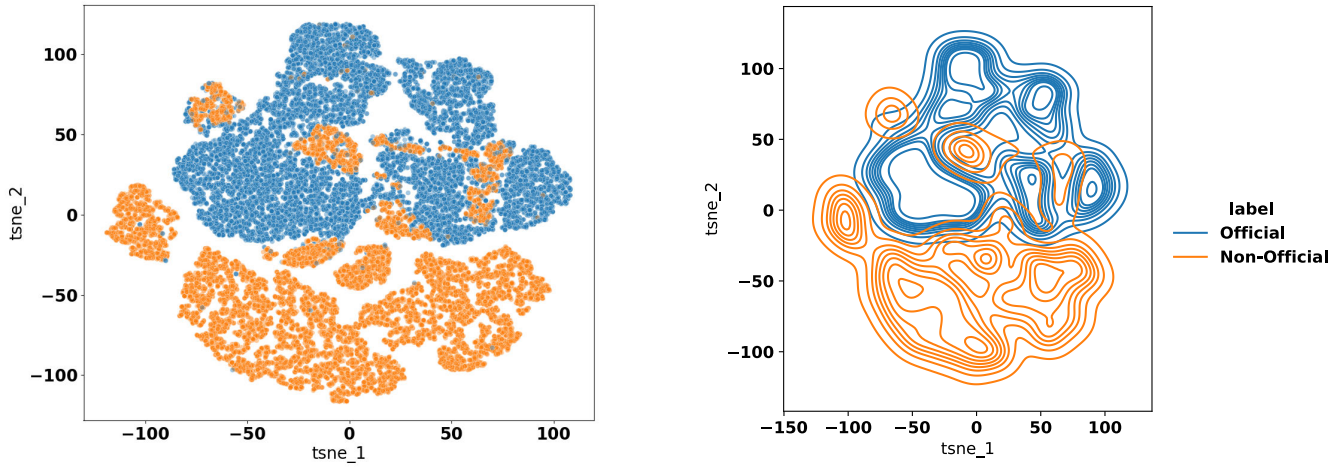
and readme content. When trained with the LGBM algorithm, this particular combination achieved a remarkable score of 0.94 for all evaluation metrics. In this combination of features, recall, precision, and F1 scores achieved identical values, indicating that all evaluation metrics uniformly suggest our methodology's capability to accurately predict both positive and negative classes. This outcome is highly significant as it demonstrates the effectiveness of our approach in accurately distinguishing between genuine and spurious official relationships between academic papers and their associated GitHub repositories using merely publicly accessible textual information from both the papers and repositories.

In addition, we visualized the text embedding that combines our best feature combination (i.e., paper title + repository title and readme) and optimal text representation (i.e., SPECTER). Figure 7 displays the embedding of our data through a scatter plot and a kernel density estimation plot, as shown in Figures 7a and 7b, respectively. The scatter plot shows the distribution of our data, while the kernel density estimation plot provides a better view of the differences between positive and negative groups. In both figures, orange represents embedding official (positive) relationships, while blue represents embedding non-official (negative) relationships.

We noticed that more distinct clusters are formed using this optimal feature configuration than all the available features, as demonstrated in Figure 6. The kernel density estimation plot allows us to divide the data points into two main regions, the upper and lower regions. The lower part predominantly displays blue, representing negative data points, while the upper part contains a significant cluster of positive samples. Our observation indicates that our representation combining the best text features and text representation enhances the discriminative power of the text representation, as we observed only a slight mixture of colors in the scatter plot.

4) RESEARCH QUESTION 4: CAN OUR PROPOSED MACHINE LEARNING-BASED APPROACH BE INTEGRATED WITH THE TRADITIONAL RULE-BASED METHOD TO IMPROVE THE OVERALL PERFORMANCE?

In practical scenarios, we have observed that numerous academic research papers mention their titles in their GitHub readme files. The traditional rule-based methodology relies on identifying the presence of a paper's title in a GitHub repository's title and readme file. This method is quick, straightforward, and often yields high precision. However, such a rule-based method does not work when a repository does not mention its official paper's title.



(a) Scatter plot.

(b) Kernel density estimation plot.

FIGURE 7. t-SNE visualization of the positive (orange) and negative (blue) samples using the best text embedding and feature combination.

TABLE 8. Performance improvement after integrating our proposed method with the traditional rule-based approach.

Method	Accuracy	Recall	Precision	F1	Δ F1 (%)
Rule-based	0.89	0.89	1.00	0.94	-
Hybrid (Rule-based + Machine Learning)	**0.98**	**0.98**	**1.00**	**0.99**	**5.31**

To overcome this limitation, we propose a hybrid approach that integrates a machine-learning-based method with the rule-based method while still benefiting from the high precision of the rule-based approach. This approach first employs the rule-based method to check whether 1) a repository's readme file contains the paper's title or 2) a paper's abstract contains a repository's title. If either of the aforementioned criteria is met, then this repository is classified as the official repository of the paper. If not, the proposed machine learning classifier determines the official relationship. By using this cascade approach, we aim to enhance the performance of identifying official relationships among academic papers or academic GitHub repositories by taking advantage of the high precision of the rule-based approach and improving the recall with the machine-learning-based approach. We present the results of this experiment in table 8, where we compare the performance of two approaches. The first method employs only the rule-based approach, while the second method refers to the hybrid method.

In evaluating the hybrid methodology's performance, we observed a notable improvement in the F1 score, with an increase of approximately 5.31%. The key advantage of the hybrid methodology, which integrates the traditional approach, lies in its ability to reduce false negatives in the model's predictions. These false negatives often pertain to official relationships between academic papers and GitHub repositories that lack explicit information, making them challenging for the rule-based approach to capture. Such reduction in false negatives directly improves recall over 10.11% from 0.89 to 0.98. The enhancement in recall and accuracy demonstrates that our model can almost perfectly predict both true positives and true negatives. The slight shortfall in recall, approximately 2%, within the hybrid methodology indicates a reduced number of false negatives. Regarding precision, achieving a score of one in both methodologies signifies our model's capability to predict the positive class accurately.

The hybrid method proves to be effective in improving the detection of official connections between academic research papers and their corresponding GitHub repositories, particularly in cases where there are no explicit references. This technique significantly enhances overall performance as compared to relying solely on finding explicit information. Specifically, this hybrid method is suitable when academic research papers and their corresponding GitHub repositories lack mutual referencing information about each other.

VI. ETHICAL ISSUES AND IMPLICATIONS

Automated paper-repository matching may raise ethical concerns, including the possibility of exposing unintended links between academic papers and their corresponding official repositories. For instance, researchers publishing their preprints on ArXiv and wishing to keep their repositories private but are compelled to make them temporarily public during the peer-review process for code review may not want their repositories to be automatically discovered until the review process is concluded.

In addition, whereas the practice of sharing research code for the sake of reproducibility is common in artificial intelligence (AI) research communities, it may not be as prevalent in other computing disciplines. Consequently, the

presence of biased training data predominantly found in AI research may result in erroneous application of the method in other computing domains that are not adequately represented in the training data. Consequently, when examining academic work in different fields, downstream tasks that employ the proposed method must acknowledge this constraint or mitigate it before application.

Regarding implications, our method allows for the identification of academic articles and their related official code repositories, specifically addressing the issue of pairs that do not have explicit reference information. This challenge frequently hinders the determination of the official repository for an academic work. This approach is designed to function efficiently even when there is missing information, such as references in scholarly articles and their official code repositories. It enables accurate matching even when unambiguous references are not available. Moreover, precisely aligning an academic publication with its official code repository can greatly affect the research's influence. Repositories that accompany scientific publications typically have higher credibility due to being scientifically evaluated and peer-reviewed, making researchers and software engineers more likely to adopt them over other non-peer-reviewed repositories. As a result, this increases the popularity of the official repository for the scholarly publication. In the future, word embeddings will be derived from several deep learning models, including SciBERT. Furthermore, optimizing these models by fine-tuning has the potential to enhance their performance by customizing them to certain datasets. This study largely examined text attributes. However, future research will explore the integration of other information from academic papers and code repositories to improve the matching process between academic articles and their corresponding repositories.

VII. LIMITATIONS

Our research focuses on four text-based features that are informative about the connection between academic papers and GitHub repositories. Previous studies have mostly concentrated on numerical characteristics, but we found that these text-based features are equally important. However, our dataset is limited as we obtained official relationships from Papers With Code, which only covers papers in the AI fields. Therefore, the evaluation results may not be generalized to other fields of study. To validate the generalizability of our approach, future research can establish ground-truth datasets in different domains of study.

Furthermore, it is crucial to note that when utilizing SPECTER for text embedding, the resulting embeddings are restricted to a set of 712 dimensions. This rigid dimensionality represents one of the limitations of our study, as the number of dimensions cannot be expanded or reduced to best capture the data. Nonetheless, we have taken measures to account for this constraint and believe that our study's findings are still highly relevant and informative. In future work, one could explore different variants of text embedding methods whose embedding dimensions can vary.

VIII. CONCLUSION

This study aimed to investigate the connections between academic papers and their corresponding GitHub repositories. The majority of links between papers and GitHub repositories were established through the paper's title, with approximately 20% of these connections being identified in the repository's readme files alongside the paper's BibTeX. However, relying solely on document similarity was inadequate for identifying official relationships. To overcome this challenge, the study experimented with various machine-learning techniques and found that a combination of text embedding by deep learning model and Principal Component Analysis (PCA) was the most effective text representation, with PCA effectively mitigating the issue of high dimensionality in the models. Furthermore, the study identified that the most informative features were the academic paper's title in conjunction with the GitHub repository's title and readme content. The optimal model achieved an impressive performance with scores of 0.94 across all evaluation metrics, including accuracy, recall, precision, and F1. Finally, the study integrated the proposed methodology with a rule-based approach and discovered that this hybrid strategy outperformed the conventional rule-based method alone, achieving a 5.31% improvement in F1 score on the test set. In the future, we aim to broaden our research's scope by incorporating more data across diverse fields of study while also examining other aspects of research, such as investigating the relationship between the papers and their corresponding GitHub repositories and exploring the utilization of identified official paper-repository relationship for recommendation and prediction applications. This research holds significant relevance in the current landscape, as the number of papers linked to GitHub has grown since the platform's inception.

REFERENCES

[1] E. Gibney, "This AI researcher is trying to ward off a reproducibility crisis," *Nature*, vol. 577, no. 7788, p. 14, Jan. 2020.
[2] P. Ball, "Is AI leading to a reproducibility crisis in science?" *Nature*, vol. 624, no. 7990, pp. 22–25, Dec. 2023.
[3] European Organization For Nuclear Research and OpenAIRE. (2013). *Zenodo*. [Online]. Available: https://www.zenodo.org/
[4] S. Wattanakriengkrai, B. Chinthanet, H. Hata, R. G. Kula, C. Treude, J. Guo, and K. Matsumoto, "GitHub repositories with links to academic papers: Public access, traceability, and evolution," *J. Syst. Softw.*, vol. 183, Jan. 2022, Art. no. 111117.
[5] Y. AlNoamany and J. A. Borghi, "Towards computational reproducibility: Researcher perspectives on the use and sharing of software," *PeerJ Comput. Sci.*, vol. 4, p. e163, Sep. 2018.
[6] W. Raghupathi, V. Raghupathi, and J. Ren, "Reproducibility in computing research: An empirical study," *IEEE Access*, vol. 10, pp. 29207–29223, 2022.
[7] C. E. Anchundia and C. E. R. Fonseca, "Resources for reproducibility of experiments in empirical software engineering: Topics derived from a secondary study," *IEEE Access*, vol. 8, pp. 8992–9004, 2020.
[8] D. Kang, T. Kang, and J. Jang, "Papers with code or without code? Impact of GitHub repository usability on the diffusion of machine learning research," *Inf. Process. Manage.*, vol. 60, no. 6, Nov. 2023, Art. no. 103477.

[9] P. H. P. Braga, K. Hébert, E. J. Hudgins, E. R. Scott, B. P. M. Edwards, L. L. Sánchez Reyes, M. J. Grainger, V. Foroughirad, F. Hillemann, A. D. Binley, C. B. Brookson, K. M. Gaynor, S. Shafiei Sabet, A. Güncan, H. Weierbach, D. G. E. Gomes, and R. Crystal-Ornelas, "Not just for programmers: How GitHub can accelerate collaborative and reproducible research in ecology and evolution," *Methods Ecol. Evol.*, vol. 14, no. 6, pp. 1364–1380, Jun. 2023.

[10] V. Cosentino, J. L. C. Izquierdo, and J. Cabot, "A systematic mapping study of software development with GitHub," *IEEE Access*, vol. 5, pp. 7173–7192, 2017.

[11] G. Gousios and D. Spinellis, "Mining software engineering data from GitHub," in *Proc. IEEE/ACM 39th Int. Conf. Softw. Eng. Companion (ICSE-C)*, May 2017, pp. 501–502.

[12] F. Chatziasimidis and I. Stamelos, "Data collection and analysis of GitHub repositories and users," in *Proc. 6th Int. Conf. Inf., Intell., Syst. Appl. (IISA)*, Jul. 2015, pp. 1–6.

[13] M.-A. Storey, C. Treude, A. van Deursen, and L.-T. Cheng, "The impact of social media on software engineering practices and tools," in *Proc. FSE/SDP Workshop Future Softw. Eng. Res.*, Nov. 2010, pp. 359–364.

[14] Y. Yu, G. Yin, H. Wang, and T. Wang, "Exploring the patterns of social behavior in GitHub," in *Proc. 1st Int. Workshop Crowd-Based Softw. Develop. Methods Technol.*, Nov. 2014, pp. 31–36.

[15] H. Borges, A. Hora, and M. T. Valente, "Understanding the factors that impact the popularity of GitHub repositories," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol. (ICSME)*, Oct. 2016, pp. 334–344.

[16] H. Borges, A. Hora, and M. T. Valente, "Predicting the popularity of GitHub repositories," in *Proc. 12th Int. Conf. Predictive Models Data Anal. Softw. Eng.*, Sep. 2016, pp. 1–10.

[17] H. Borges and M. Tulio Valente, "What's in a GitHub star? Understanding repository starring practices in a social coding platform," *J. Syst. Softw.*, vol. 146, pp. 112–129, Dec. 2018.

[18] H. S. Borges and M. T. Valente, "How do developers promote open source projects?" *Computer*, vol. 52, no. 8, pp. 27–33, Aug. 2019.

[19] Y. Fan, X. Xia, D. Lo, A. E. Hassan, and S. Li, "What makes a popular academic AI repository?" *Empirical Softw. Eng.*, vol. 26, no. 1, pp. 1–35, Jan. 2021.

[20] J. Han, S. Deng, X. Xia, D. Wang, and J. Yin, "Characterization and prediction of popular projects on GitHub," in *Proc. IEEE 43rd Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 1, Jul. 2019, pp. 21–26.

[21] S. E. Sahin, K. Karpat, and A. Tosun, "Predicting popularity of open source projects using recurrent neural networks," in *Proc. IFIP Int. Conf. Open Source Syst. (OSS)*, Montreal, QC, Canada. Cham, Switzerland: Springer, May 2019, pp. 80–90.

[22] A. Al-Rubaye and G. Sukthankar, "Scoring popularity in GitHub," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2020, pp. 217–223.

[23] A. Zerouali, T. Mens, G. Robles, and J. M. Gonzalez-Barahona, "On the diversity of software package popularity metrics: An empirical study of npm," in *Proc. IEEE 26th Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Feb. 2019, pp. 589–593.

[24] K. Aggarwal, A. Hindle, and E. Stroulia, "Co-evolution of project documentation and popularity within GitHub," in *Proc. 11th Work. Conf. Mining Softw. Repositories*, May 2014, pp. 360–363.

[25] G. A. A. Prana, C. Treude, F. Thung, T. Atapattu, and D. Lo, "Categorizing the content of GitHub README files," *Empirical Softw. Eng.*, vol. 24, no. 3, pp. 1296–1327, Jun. 2019.

[26] P. Vandewalle, "Code availability for image processing papers: A status update," in *Proc. WIC IEEE SP Symp. Inf. Theory signal Process.*, May 2019.

[27] K.-J. Stol, M. A. Babar, B. Russo, and B. Fitzgerald, "The use of empirical methods in open source software research: Facts, trends and future directions," in *Proc. ICSE Workshop Emerg. Trends Free/Libre/Open Source Softw. Res. Develop.*, May 2009, pp. 19–24.

[28] M. Färber, "Analyzing the GitHub repositories of research papers," in *Proc. ACM/IEEE Joint Conf. Digit. Libraries*, Aug. 2020, pp. 491–492.

[29] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, and T. Tiropanis, "Open source research software," *Computer*, vol. 53, no. 8, pp. 84–88, Aug. 2020.

[30] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[31] A. Cohan, S. Feldman, I. Beltagy, D. Downey, and D. Weld, "SPECTER: Document-level representation learning using citation-informed transformers," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 2270–2282. [Online]. Available: https://aclanthology.org/2020.acl-main.207

[32] M. O. F. Rokon, P. Yan, R. Islam, and M. Faloutsos, "Repo2 Vec: A comprehensive embedding approach for determining repository similarity," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol. (ICSME)*, Sep. 2021, pp. 355–365.

[33] H. Shao, D. Sun, J. Wu, Z. Zhang, A. Zhang, S. Yao, S. Liu, T. Wang, C. Zhang, and T. Abdelzaher, "paper2repo: GitHub repository recommendation for academic papers," in *Proc. Web Conf.*, Apr. 2020, pp. 629–639.

[34] P. H. Russell, R. L. Johnson, S. Ananthan, B. Harnke, and N. E. Carlson, "A large-scale analysis of bioinformatics code on GitHub," *PLoS ONE*, vol. 13, no. 10, Oct. 2018, Art. no. e0205898.

[35] V. K. Vijayan, K. R. Bindu, and L. Parameswaran, "A comprehensive study of text classification algorithms," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 1109–1113.

[36] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, Apr. 2019.

[37] M. Trivedi, S. Sharma, N. Soni, and S. Nair, "Comparison of text classification algorithms," *Int. J. Eng. Res. Technol.*, vol. 4, no. 2, pp. 334–336, 2015.

[38] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 1–9.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[40] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*. Hong Kong: Association for Computational Linguistics, 2019, pp. 3615–3620. [Online]. Available: https://aclanthology.org/D19-1371

[41] A. D. Wade, "The semantic scholar academic graph (S2AG)," in *Proc. Companion Web Conf.*, Apr. 2022, p. 739.

[42] L. Gomes, R. da Silva Torres, and M. L. Côrtes, "BERT- and TF-IDF-based feature extraction for long-lived bug prediction in FLOSS: A comparative study," *Inf. Softw. Technol.*, vol. 160, Aug. 2023, Art. no. 107217.

[43] T. Yao, Z. Zhai, and B. Gao, "Text classification model based on fastText," in *Proc. IEEE Int. Conf. Artif. Intell. Inf. Syst. (ICAIIS)*, Mar. 2020, pp. 154–157.

[44] T. Kurita, "Principal component analysis (PCA)," in *Computer Vision: A Reference Guide*. Cham, Switzerland: Springer, 2019, pp. 1–4.

[45] Z.-H. Zhou, *Machine Learning*. Cham, Switzerland: Springer, 2021.

[46] S. Suthaharan, "Support vector machine," in *Machine Learning Models and Algorithms for Big Data Classification* (Integrated Series in Information Systems), vol. 36. Berlin, Germany: Springer, 2016, pp. 207–235.

[47] L. Breiman, "Random forests," *Mach. Learning*, vol. 45, pp. 5–32, Oct. 2001.

[48] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers Neurorobotics*, vol. 7, p. 21, Dec. 2013.

[49] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[50] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–12. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[51] H. R. Seireg, Y. M. K. Omar, F. E. A. El-Samie, A. S. El-Fishawy, and A. Elmahalawy, "Ensemble machine learning techniques using computer simulation data for wild blueberry yield prediction," *IEEE Access*, vol. 10, pp. 64671–64687, 2022.

[52] I. D. Mienye and Y. Sun, "A deep learning ensemble with data resampling for credit card fraud detection," *IEEE Access*, vol. 11, pp. 30628–30638, 2023.

[53] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–27, 2008.

PRAHYAT PUANGJAKTHA received the bachelor's degree from Chiang Mai University, Thailand. He is currently pursuing the master's degree in computer science with the Faculty of Information and Communication Technology, Mahidol University, Thailand. His research interest includes applications of machine learning techniques in software repository mining.

MORAKOT CHOETKIERTIKUL received the Ph.D. degree in computer science from the University of Wollongong (UOW), Australia. He is currently a Lecturer with the Faculty of Information and Communication Technology (ICT), Mahidol University, Thailand, where he co-founded the Software Engineering Research Unit (SERU). His research interests include applying AI solutions to improve software quality and software processes. His research has been published at top-tier software engineering venues, such as IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, *Empirical Software Engineering*, and the International Conference on Software Engineering (ICSE).

SUPPAWONG TUAROB (Member, IEEE) received the B.S.E. degree in computer science and the M.S.E. degree in computer science and engineering from the University of Michigan, Ann Arbor, and the M.S. degree in industrial engineering and the Ph.D. degree in computer science and engineering from The Pennsylvania State University. He is currently an Associate Professor of computer science and the Director of the Machine Intelligence and Knowledge Engineering (MIKE) research cluster at the Faculty of Information and Communication Technology, Mahidol University, Thailand. His research interests include data mining in large-scale digital libraries, software engineering, social sciences, healthcare, and the applications of intelligent technologies for the betterment of society.