Cureus Journal of Computer Science

Open Access Review Article

Semantic Graph Analytics for Research Trends in Software Defect Prediction

Taiwo O. Olaleye [1] , Oluwasefunmi Arogundade [1] , Dada A. Aborishade [1] , Olusola J. Adeniran [2] , Oluwatomisin Ajayi [3] , Wilson Ahiara [4]

1. Computer Science, Federal University of Agriculture, Abeokuta, Abeokuta, NGA 2. Mathematics, Federal University of Agriculture, Abeokuta, Abeokuta, NGA 3. Electrical and Computer Engineering, University of New Haven, West Haven, USA 4. System Engineering, Michael Okpara University of Agriculture, Umudike, Umudike, NGA

Corresponding author: Taiwo O. Olaleye, olaleyeto@funaab.edu.ng

Review began 12/13/2024
Review ended 12/21/2024
Published 12/25/2024

DOI: https://doi.org/10.7759/s44389-024-02851-3

Abstract

Software defect prediction (SDP) is a critical area in software engineering quality assurance, leveraging predictive analytics to enhance defect identification and resolution. However, existing approaches often narrowly focus on single methodologies or lack consensus-driven insights, limiting their ability to inspire future research. This study addresses these gaps by exploring semantic relationships among 31 published studies, including journal articles, conference proceedings, and book chapters, using a novel graph analytics framework. Research articles were represented as nodes, and edges were formed based on a minimum threshold of five shared semantic terms. Degree centrality was computed to quantify the influence and connectivity of articles within the research network. Results revealed a significant clustering of studies, with a conference proceeding exhibiting the highest degree centrality (0.309859), indicative of its methodological influence on 59.7% of the network. Conversely, 40.2% of articles were isolated, highlighting unique research directions, including sentiment analysis of defect reports. Findings provide critical insights into the consensus and divergence in SDP research, emphasizing the importance of ensemble methods, severity prediction, and feature selection while revealing gaps in methodological overlap and emerging topics. This work demonstrates the utility of semantic graph analytics in extracting valuable knowledge from scholarly literature to advance the SDP domain.

Categories: Graph Algorithms, Software Quality Assurance (SQA), Data Mining
Keywords: data science, semantic relation, software defect prediction, graph algorithm, degree centrality

Introduction And Background

Knowledge detection through text mining entails extracting meaningful insights from unstructured textual data [1] and has significantly evolved as a key task in natural language processing (NLP). This advancement has facilitated various applications, including opinion mining via sentiment analysis (SA) [2], topic modeling [3], predictive analytics of social texts, automated text document summarization [4,5], semantic relation extraction (SRE) [4], and textual entailment (TE) studies [6], among others. Notably, SRE has gained substantial attention as a research technique for evaluating NLP tasks [7], particularly as it extends beyond basic text summarization by extracting mutual semantic entities across documents. By identifying similarities or differences within a network of documents addressing similar subjects, SRE can function as an automated systematic literature review (SLR) tool [8], helping establish ground truths within datasets to address research questions - this forms the core focus of this study.

Traditional review methodologies often rely on manual annotation and mapping of research questions to findings from primary studies, a process that is both time-intensive and prone to errors [4]. Moreover, SRE has yet to be extensively applied in software defect prediction (SDP) review studies. This study addresses these limitations by constructing a semantic relationship graph from primary studies, enabling TE and clustering studies with similar methodological approaches through measures of degree centrality. Motivated by the need to systematically review methodological approaches for SDP, particularly those adopting data science-based models, this research seeks to identify clusters of studies employing similar techniques. By analyzing abstracts from selected scientific studies, the level of TE between neighboring articles is evaluated. The resulting semantic relationship is modeled as a graph, where each node represents a research abstract and edges indicate semantic relationships between articles, connecting nodes only when a semantic relationship exists.

To enhance the scope and likelihood of identifying semantic relationships, a 5-relation threshold is employed in this study, establishing edges between neighboring nodes that share at least five mutual lexical items. This approach creates a robust framework for knowledge extraction, as demonstrated through the evaluation of 31 selected scientific research abstracts. While traditional information retrieval methods address general queries, the domain of scientific publications requires more precise and context-aware solutions. Unlike general-purpose search engines, which yield broad results for keyword searches, document mining of scientific publications demands focused retrieval [4]. Text mining for relation extraction, as an

NLP evaluation task, provides directional insights into text components, and recent studies have optimized existing frameworks for this purpose. For instance, Olaleye et al. [4] emphasizes the potential of SRE in cybersecurity studies, criticizing SLRs for narrowly focusing on research questions and neglecting potential semantic relationships in primary studies. Techniques such as sequential and tree-structured long short-term memory [9], relation-to-relation paradigms [10], and neural ensemble classifiers [11] have enhanced SRE by incorporating advanced methods like dependency trees, word embeddings, and data-driven approaches devoid of manual rules.

A graph convolutional network framework proposed by Kapanipathi et al. [6] incorporates personalized page rank algorithms to create contextual sub-graphs, reducing noise and improving TE through external knowledge infusion. Similarly, hybrid systems, as explored by Zeng et al. [12], leverage human expertise and secure vector machines to extract relational patterns, particularly within the digitized medical domain. Additionally, Karaa et al. [13] introduce a double-graph GAIN methodology for document-level relation extraction, utilizing a hierarchical multi-grain graph to capture multifaceted interactions within documents. Evaluation on public repositories such as DocRED demonstrates the efficacy.

Relation extraction from biomedical literature was the purpose of Kruiper et al. [14], using machine learning and NLP. Identification of entities related to drugs and diseases from medical texts would assist doctors as recommender systems during consultation, as achieved in the study. A secure vector machine algorithm is deployed for the machine learning phase, while the NLP phase takes advantage of Unified Medical Language System ontology, which is the strength of the study. In Nisar et al. [15], scientific texts were the target for relation extraction in a semi-open approach for guiding readers to the germane information in scientific documents, and the model was evaluated on the FOBIE dataset. Techniques for analyzing graphs on big data were conducted by Wang et al. [16] to ascertain the enhancement offered by graph analysis, as graphs enjoy widespread application for its flexibility and self-expressive outlook. Similarly, a document-level extraction model on rhetorical structure theory (RST)-graph was the focus of Geng et al. [17] introducing rhetorical structure RST to select suitable evidence to show cognitive process on new document graph, while Kaur and Jindal [18] propose a joint-entity and relation extraction framework on a robust semantics. Highlights of the approach include the end-to-end method while utilizing allied information between data points and relations. The introduction of external features was not necessary in the approach as the framework achieves improvement on public benchmarks.

Review

The main thrust of the proposed methodology stems from the need to review research articles through graph analytics of semantic relationship. In traditional literature review, possibilities of oversights are widespread, which could result in the misrepresentation of facts, posing a threat to internal validity. Hence, SRE is implemented in this study to aid review by mapping specific research question to primary study clusters through graph analytics. A network of sub-graph is thereby constructed with the computation of degree centrality of nodes from the tokenized research abstracts. This will aid the design of sub-graphs that show the association of semantic relations between research documents that make up the corpus. Extraction of insights from the resulting conceptual relations would shed lights on mutual techniques being addressed by each research article contained in the cluster. As presented in Figure *1*, the proposed framework involves four main components, including data acquisition, features extraction, text preprocessing, and the graph analytics through SRE. Therefore, the first component acquires literatures on predicting software defect using machine learning, while the second component extracts useful features from each research article. A natural language pre-processing task is conducted in the third component to yield a bag of tokenized text corpus for each research document. The extracted tokenized summary features help in the relation extraction component of the fourth phase, which is represented graphically and thereby completing the proposed fully automated literature annotation.

Experimental setup environment

The experimental setup for this study involves the use of the Orange Data Mining toolkit, a versatile, open-source data analysis platform. This toolkit (version 3.28.0) was used in this study due to its user-friendly interface. It is a robust NLP toolkit, implementing diverse Python libraries. Its support for visual programming is also key. Its interactive data visualization capabilities allow for seamless preprocessing, exploration, and analysis of textual data, making it particularly suitable for tasks involving semantic relationships and network graphs. The decision to use Orange over other tools like WEKA or RapidMiner was influenced by its strong integration of data visualization and analytical pipelines, offering both efficiency and clarity in presenting the results. The following steps outline the environment and procedures:

Corpus Preparation and Preprocessing

· A dataset comprising 31 studies (journals, conference proceedings, and book chapters) is prepared based on defined inclusion and exclusion criteria.

· Extracted metadata includes titles, keywords, abstracts, publication years, and study types.

· The corpus undergoes preprocessing steps, such as tokenization, stop-word removal, lemmatization, and standardization, to ensure uniformity and relevance.

· The preprocessed dataset is saved as a CSV file for subsequent analysis.

Software Environment

· Orange Data Mining Toolkit (version 3.28.0) is utilized as the primary analytical platform.

· The software is chosen for its user-friendly visual programming interface, making it suitable for complex data visualization and network analysis.

Network Construction

· The prepared CSV file is imported into Orange for further processing.

· The Network Add-on is activated to enable graph-based analytics.

· A similarity threshold of five shared semantic words is applied to determine edges between nodes, with each node representing a study and edges indicating semantic relationships.

System Specifications

· The analysis is conducted on a machine running Windows 10 operating system.

· Hardware includes an Intel Celeron processor, 4 GB RAM, and a 16 GB SSD to ensure efficient processing of the corpus and graph construction.

Experimental Workflow in Orange

· The CSV file is loaded using the File widget.

· Preprocessing tasks are reviewed using the Corpus Viewer or similar widgets.

· The Network widget is employed to construct the graph based on the semantic similarity threshold.

· Visualization widget is used to show resulting graph.

Source of the dataset

The dataset used in this study comprises 31 published works, including journals, conference proceedings, and a book chapter obtained through a systematic search adhering to inclusion and exclusion criteria. The sources of the dataset include repositories such as IEEE Xplore, Google Scholar, and Scopus. The characteristics of the dataset include metadata such as titles, keywords, abstracts, publication years, and study types, which were extracted to form a corpus. This corpus was utilized to construct a semantic network graph, where studies are represented as nodes, and edges connect nodes sharing at least five semantic words in their texts. The references for the sources of the dataset are included in the reference section. Table *1* shows list of consulted digital libraries, which are reputable for software engineering resource as well as the inclusion and exclusion criteria stipulated in Table *2*. The source of the dataset is presented in Table *3*.

The inclusion and exclusion criteria were designed to ensure relevance and focus on the study's objectives while maintaining methodological rigor. For example, including studies that deploy hybridized methodologies alongside machine learning acknowledges the growing trend of integrating multiple techniques to enhance predictive accuracy in SDP. Hybridized approaches often combine machine learning with diverse feature engineering techniques, which provide a richer perspective on methodologies advancing the field. Including such studies enhances comprehensiveness by capturing the diversity of effective practices and innovations, reducing the risk of overlooking impactful research. Excluding studies that do not adopt machine learning-based methodologies ensures focus, aligning with the study's scope to evaluate its application in SDP. This exclusion minimizes bias toward irrelevant methods that may dilute the analysis. However, it may introduce a limitation by excluding potentially valuable insights from alternative approaches, which will be the focus of future studies.

Feature extraction

The extraction of relevant text features from the acquired 31 primary studies is followed by the feature extraction phase. Five features are extracted from each study, including the research topic, research aim and methodology (from the abstract), keywords attached to each study, the type of article (journal, proceeding,

or book chapter), and the year of publication. The extracted features, excluding the year of publication, for each research article serve as the summary report for each research entry. The type of article (journal, proceeding, or book chapter) is established as the ground truth for each summary report contained in the dataset. These features are believed to be representative of major data points that could unravel germane research intents. Consequently, a total of 46,945 word tokens are contained in the entire summary report, yielding a 31:46,945 feature-attribute corpus. The textual summary reports are prepared as a CSV corpus and organized in a text editor for the next phase of the framework. Figure *2* shows a snapshot of the summary reports extracted from the primary studies.

Text preprocessing

In this phase, NLP techniques are applied to the unstructured summary report, specifically the 31:46,945 textual corpus. The goal of preprocessing is to convert the 31 summary reports into distinct words, a crucial step for text mining studies that rely on the lexical and semantic relationships between data points to construct the proposed network graph. The preprocessing techniques employed include lemmatization, stop-word removal, transformation, and tokenization. A detailed description of the preprocessing tasks for each technique is provided in Table *4*. Notably, terms such as "machine learning," "defect," and "bug" are deliberately removed, as they are common within the context of the targeted research articles and would appear in nearly all the summary reports. By focusing on words with higher information gain, the analysis avoids semantic redundancy and ensures that the extracted relationships reflect meaningful and distinctive research trends. This preprocessing step enhances the clarity and specificity of the resulting network graph, aligning it closely with the study's objectives.

SRE graph

This section describes how the graph network is constructed through SRE conducted on the tokenized text corpus. The semantic relationships between the 31 primary studies could be word-based or sentence-based and can exist between two or more report documents (referred to as nodes) to establish the graphical illustration. A word or sentence-based semantic relationship threshold parameter determines if a relationship will exist between two or more nodes (two or more research articles). For the purpose of this study, a semantic relation threshold of 5 is set, which will determine the structure of sub-graphs that make up the proposed graph network. The structural makeup of the proposed network is determined by the degree centrality of nodes. The degree centrality of a node (a tokenized summary report in this case) is defined by the number of links incident upon a node. Hence, the degree centrality of a node is the number of neighbors (links) that a node has. In the proposed network, an edge will exist, linking neighboring nodes, if a 5-threshold semantic relationship exists between them. The degree centrality of a node, therefore, determines the degree of semantic relation that the node has with other nodes within the network. A network of lexical mentions is thereby established; so far all 31 nodes do not have null degree centrality. Nodes with degree centrality greater than zero will, therefore, be clustered within the network, from where words that establish their semantic relationship will be reviewed to uncover topical issues.

Consider the structural representation of the proposed network of mentions in Figure *3*. The circular nodes (n) represent each research study, with a total of 31 nodes within the network. An edge (e) will exist between two nodes if there is a word-based or phrase-based semantic relation between neighboring nodes. The edges are abstractions of degree centrality relations within the network. The position of a node in the network and its number of inbound or outbound edges are functions of its degree centrality, which determines the number of mentions contained in its tokenized summary. The output degree centrality scores from the resulting network of mentions would be used to further evaluate the degree of associativity of words contained in each summary report and are calculated as follows:

$$d_c = \frac{e}{n-1} \qquad (1)$$

where d_c is the degree centrality of a node, e is the number of in-degree or out-degree edges of the node, and n is the number of nodes within the entire network. Therefore, the number of mutually shared semantic words contained in each nodes determines their degree centrality (likewise reveals the consensus methodology in the respective research articles). The SRE graph of this study is constructed by Python libraries on the Orange data mining toolkit.

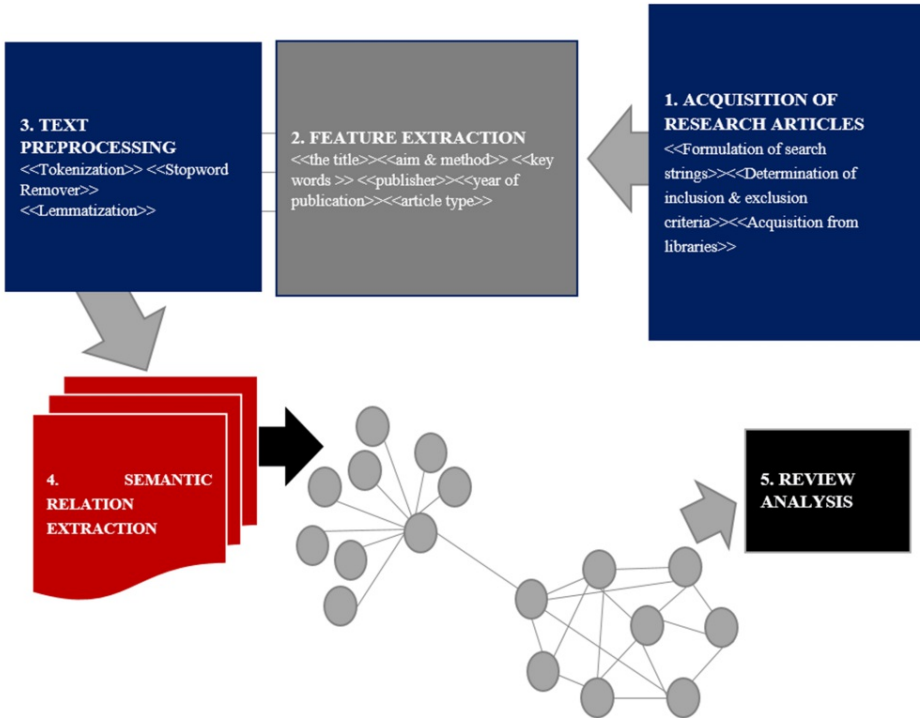

FIGURE 1: Overview of the study framework

Library	Source	Results found	Initial selection	Final selection after quality assurance
IEEE	@ieeexplore.ieee.org	536	23	9
ACM	@dl.acm.org	204	12	5
Springer Link	@link.springer.com	354	14	4
Google Scholar	@scholar.google.com	1,466	28	5
Elsevier	@elsevier.com	102	10	6
Total		2,662	86	29
Snowballing		11	10	2
Final total		2,673	96	31

TABLE 1: Data collection distribution across libraries

ACM: Association for Computing Machinery; IEEE: Institute of Electrical and Electronics Engineers

Inclusion Criteria	**Articles that use machine learning, NLP and written in English for software defect prediction**
	Articles deploying hybridized methodologies alongside machine learning
Exclusion Criteria	Studies outside English language, journal, conference, and book chapters
	Studies adopting non-machine learning methodologies
	Studies without keyword section

TABLE 2: Inclusion and exclusion criteria

NLP: natural language processing

Study type	Authors	Total
Journals	[19-39]	21
Proceedings	[40-48]	9
Book chapter	[48]	1

TABLE 3: Source of dataset per study category

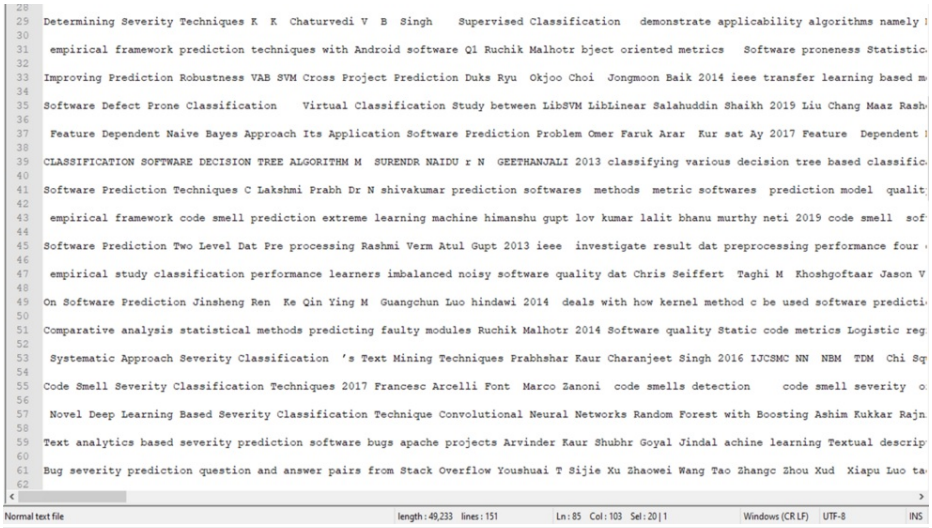

FIGURE 2: Summary reports of extracted features

S/N	Technique	Task(s)	Result
1.	Tokenization	Summary report of each research article (as described in ….) is split into a bag-of-words	A tokenized textual corpus for each 31 report corpus
2.	Stop-word and special character removal	Unnecessary words with no relevance in answering research questions are discarded from the tokenized textual corpus since they do not convey specific information.	A tokenized 31-no summary report devoid of ‘a’, ‘an’, ‘the’, ‘on’, ‘of’, ‘to’, ‘and’, ‘we’, ‘in’, ‘will’, ‘this’, ‘by’, ‘therefore’, ‘using’, ‘bug’, ‘defect’, ‘machine learning’, ‘,’, ‘.’, ‘:’, ‘;’, ‘-‘, etc.
3.	Lemmatization	The resulting tokenized corpus is reduced and transformed into their root words to avoid duplication of words with analogous meaning	A fully lemmatized 31-no tokenized summary report
4.	Transformation	Conversion to lower case letters	Bag-of-words in small letters

TABLE 4: Preprocessing techniques and tasks

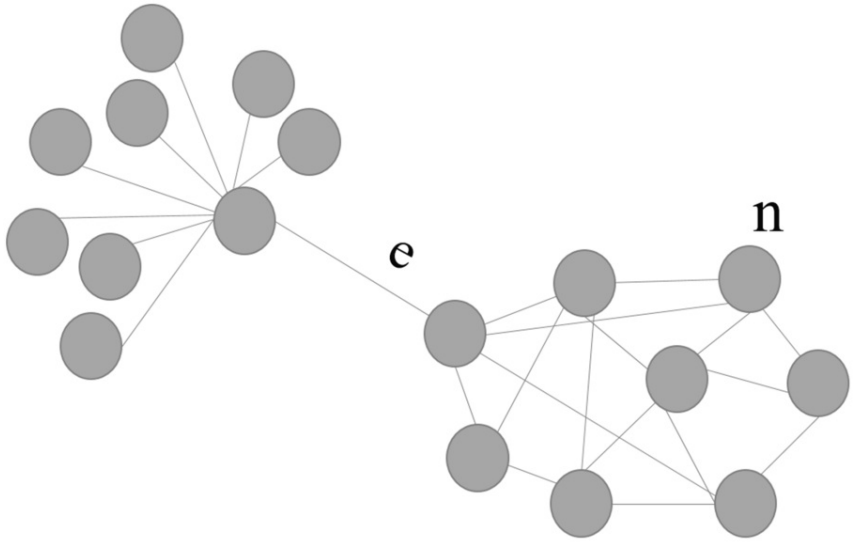

FIGURE 3: Structural graph illustrating the intended network of mention showing nodes (n) of 31 primary studies and edges (e) actualized by 5-no semantic relation threshold

The pairwise semantic relationship between two or more research articles, as graphically represented in the semantic relation network, is based on a five-relation threshold parameter. The experimental results produced a network of 31 nodes (0.55 per edge) and 130 edges (1.81 per node). The structure of the output graph network is presented in Figure *4*, showing the semantic relationships existing between all research articles under review, with their degree centrality as computed with respect to Equation (1).

In the network, 40.2% of the research nodes are isolated with zero degree centrality, while the remaining 59.7% have semantic relations with neighboring nodes. About 55.1% of isolated research nodes, with zero degree centrality, are journal articles, while 41.4% are of conference proceedings. Research articles clustered around the north-eastern part of the network are connected by edges, meaning that they are sufficiently semantically related and co-expressed, while each of the isolated articles around the south-western part of the network have no edge with neighboring articles, hence with a degree centrality of zero. The resulting network graph gives insights into germane review questions, including the direction of research methodology, popular machine learning techniques deployed in literature, and important topics semantically captured and co-expressed by research nodes on the network.

The large white space between isolated articles and communicative articles as observed in Figure *4* under

reference is indicative of a sharp contrast between research interests and the uniqueness of articles in the isolated cluster. With the aid of the degree centrality of each node, cohesiveness amongst research articles is easily determined to uncover area of consensus among communicative articles. In-degree and out-degree of each nodes likewise show the level of topical relativity and similarity in research direction. The isolated nodes, which comprise 40.2% of the research articles with zero degree centrality, represent unique research directions. These nodes do not share a sufficient semantic relationship with other studies in the corpus. These articles explore specialized or emerging areas of SDP, such as SA of defect reports, as indicated by prominent tokens like "report" and "sentiment" (Table *8*). Words in the cluster with high weights, including "prediction", "software", "classification", are basic data science-based terms that do not carry enough information gain to warrant being termed as methodological approach. Their isolation, therefore, reflect a lack of widespread alignment with mainstream methodologies.

As seen in Table *5*, the most communicative journal article records 22 inbound and outbound edges, thus having the highest degree centrality, while the conference proceeding article with the most semantic relations with other nodes records 19 in-degree and out-degree centralities (as shown in Table *6*). Communicative conference papers, however, are more related to neighboring nodes than their journal article counterparts, as indicated by the comparative analysis in Tables *5* and 6. Hence, conference proceedings in the primary studies have similar research directions compared to the journal articles.

The heat map in Figure *5* is instructive, visualizing degree centralities for the most communicative journal articles. The high centrality values for articles such as "software prediction supervised learning algorithm" (0.309859) and "ensemble software predictors" (0.197183) indicate that these works are key hubs in the research network, suggesting they are foundational or highly influential in the field of SDP. In contrast, articles with lower centrality values, like "investigating tree family techniques software" (0.042254), are less central, indicating they may be more specialized or less frequently replicated in studies. This distribution highlights the concentration of methodological focus in certain areas of SDP, providing insights into prevailing trends and potential gaps in the literature.

The degree centrality among the most communicative conference proceedings (presented in Figure *6*) highlights the relative influence and connectivity of various techniques within the conference domain. High centrality values for proceedings such as "comparative study ensemble feature selection" (0.267606) and "emotion based automated priority prediction" (0.239437) signify that these studies are pivotal in shaping the discourse on SDP, indicating a strong presence and influence in the proceedings network. Conversely, proceedings with lower centrality, like "empirical improving severity prediction reports" (0.028169), suggest a more niche or specialized focus, with less overall influence in the broader network of SDP research. This distribution underscores the concentration of significant methodologies in certain areas while revealing the presence of more focused, potentially emerging topics in the field.

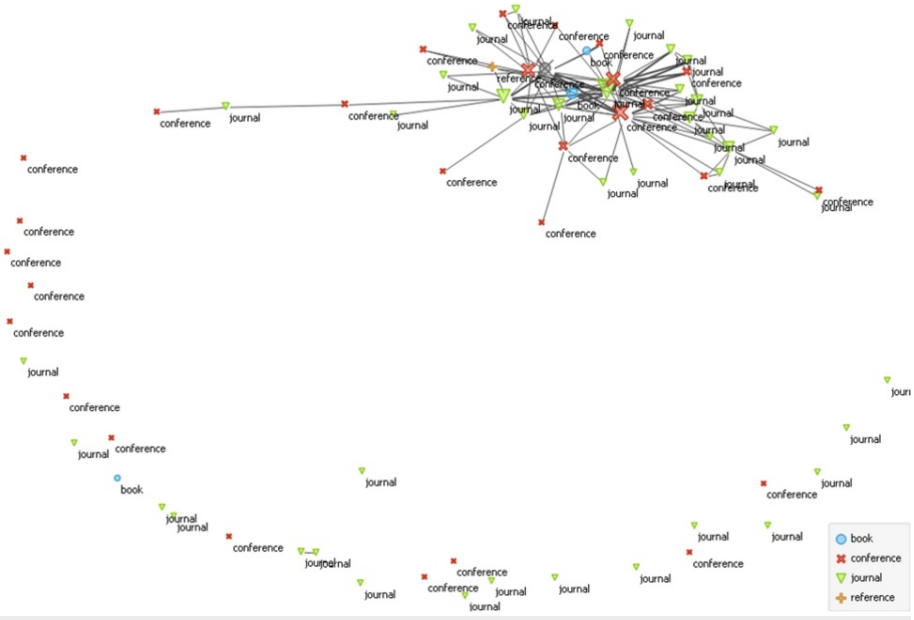

FIGURE 4: The network graph of acquired primary studies plotted using the degree centrality on 5-relation threshold

Tokenized corpus	In-degree	Out-degree	Degree centrality
Software prediction supervised learning algorithm …	22	22	0.309859
Ensemble software predictors …	14	14	0.197183
Prediction severity mining software project reports	9	9	0.126761
Classification framework software prediction…	9	9	0.126761
Text analytics based severity prediction software …	8	8	0.112676
Software prediction heterogeneous ensemble class…	7	7	0.098592
Predicting software techniques …	5	5	0.070423
Feature dependent naive Bayes approach…	4	4	0.056338
Empirical study classification performance…	4	4	0.056338
Novel deep learning severity classification…	4	4	0.056338
Hybrid smote ensemble approach software prediction	4	4	0.056338
Comparative study iterative feature selection tech…	3	3	0.042254
Software fault prediction tree based clustering…	3	3	0.042254
Investigating tree family techniques software …	3	3	0.042254
Prediction severity mining software reports …	3	3	0.042254

TABLE 5: Distribution of degree centrality among 15 most communicative journal articles

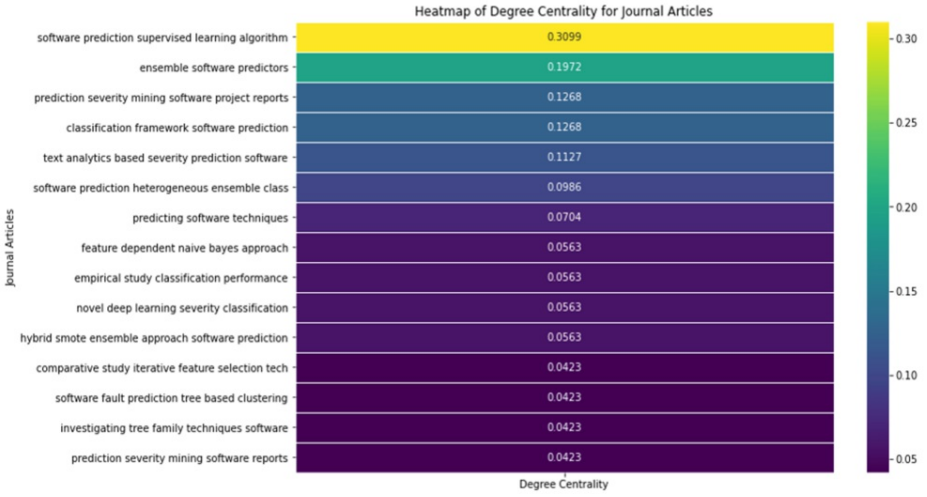

FIGURE 5: Heat map showing degree centrality of the most related journal articles

Tokenized corpus	In-degree	Out-degree	Degree centrality
Comparative study ensemble feature selection…	19	19	0.267606
Emotion based automated priority prediction…	17	17	0.239437
Deep neural network based severity prediction…	16	16	0.225352
Novel multiple ensemble learning models based…	16	16	0.225352
Learning predict severity software vulnerability…	10	10	0.140845
Software prediction neural networks …	9	9	0.126761
Building ensemble software prediction diversity…	6	6	0.084507
Using smote heterogeneous stacking ensemble…	5	5	0.070423
Severity prediction software vulnerabilities…	3	3	0.042254
Improve accuracy severity categorization semi…	3	3	0.042254
Ensemble oversampling model class imbalance…	3	3	0.042254
Applying dat analytic techniques fault detection…	3	3	0.042254
Empirical framework code smell prediction…	2	2	0.028169
Software prediction data preprocessing…	2	2	0.028169
Empirical improving severity prediction reports…	2	2	0.028169

TABLE 6: Distribution of degree centrality among 15 most communicative conference proceedings

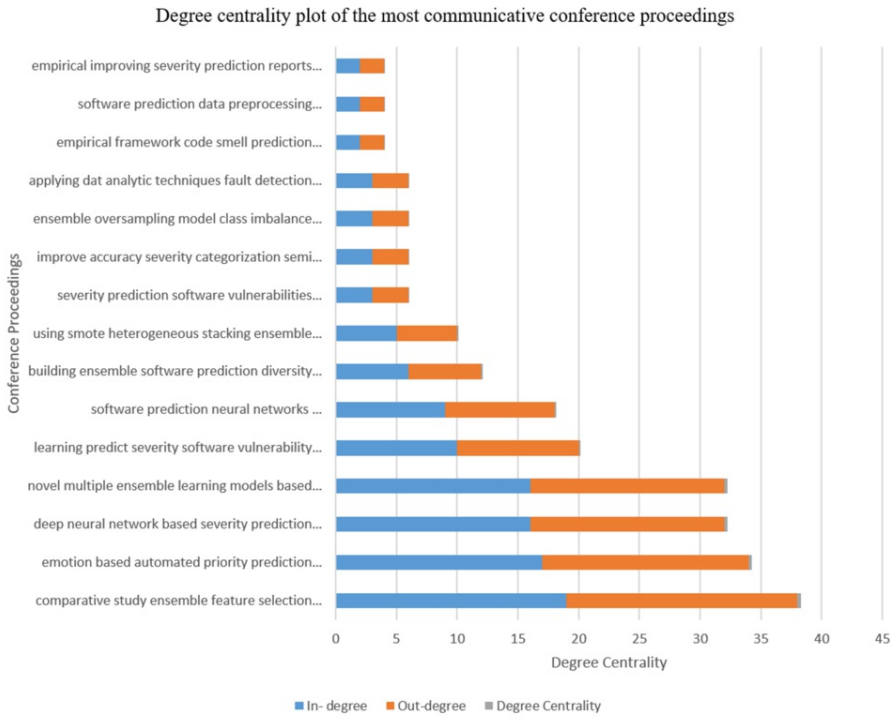

FIGURE 6: Clustered bar plot showing degree centrality of the most communicative conference proceedings

Words contained in studies with the highest degree centralities are identified and are as presented in Table 7, while those with null degree centrality (without an edge with any other study) are presented in Table 8. As could be observed from Table 7, prominent topic of discuss amongst the articles could be modeled as a research topic as follows:

"prediction" of "software" defects "severity" using "ensemble" machine learning "classification" and "neural" "techniques" on defect "feature" "reports."

The study with the largest degree centrality of 0.309859 clearly encapsulates the methodological approach of 59.7% of the entire 31 primary studies. Tokens from its input data are presented in Figure 7, with 88 tokenized words capturing its aim, data source, methodology, and evaluation criteria. Focusing on articles with null degree centrality reveals their unique research approach. With journal articles leading the pack, none of the studies have more than five mutual semantic relations with neighboring nodes. As observed in Table 8, four tokens with the most prominence share semantics with nodes with high degree centralities in Table 7. However, the fifth and sixth tokens, "report" and "sentiment," are indicative of the popular topic in the cluster. It could be inferred that SA of software defect reports is widespread in the cluster. Analysis of the sub-graphs reveals techniques employed by the acquired primary studies. Studies use cross-project metrics for their SDP predictive analytics, while others that deploy feature engineering techniques, such as feature selection, make up a separate sub-graph. Studies also investigate the implications of class imbalance for SDP, just as some nodes focus on detecting software defect severity levels.

Implications for empirical software engineering

Experimental results indicate a diverse deployment of data science-based feature engineering techniques, showing the depth of methodological approach in SDP studies. Primary studies have widely deployed ensemble and deep learning modeling for SDP just as defect severity detection has also been prominent. The use of NLP in Data Science is also prominent in literature, especially adopting defect reports in repositories. Studies have produced state-of-the-art results with verifiable conclusion validity tests. In a real-world scenario, trends in the 31 primary studies show a popular methodological approach as including acquisition of software metrics in repositories, preprocessing to remove noise or for tokenization (if defect report data are adopted), ensemble, base or deep learning of patterns, and defect prediction. This appears to be the basic description of SDP approach adopted by primary studies. It is, however, important to note that Graph Theory has not been fully employed in SDP studies, as none of the primary studies reflected this paradigm. Graph Neural Networks have been revolutionizing predictive analytics using graph data, and this offers SDP a robust predictive analytics, especially to advance from defect prediction to defect localization.

S/N	Word	Weight
1.	prediction	21
2.	software	19
3.	ensemble	13
4.	techniques	12
5.	classification	12
6.	reports	12
7.	severity	11
8.	feature	10
9.	neural	8

TABLE 7: Weights of words contained in nodes with largest degree centrality

S/N	Word	Weight
1.	prediction	50
2.	software	48
3.	severity	29
4.	classification	27
5.	report	20
6.	sentiment	10
7.	2019	9
8.	ensemble, Bayes	8
9.	supervised, lexicon	6
10.	Elsevier, SVM	5

TABLE 8: Weights of words contained in nodes with null degree centrality

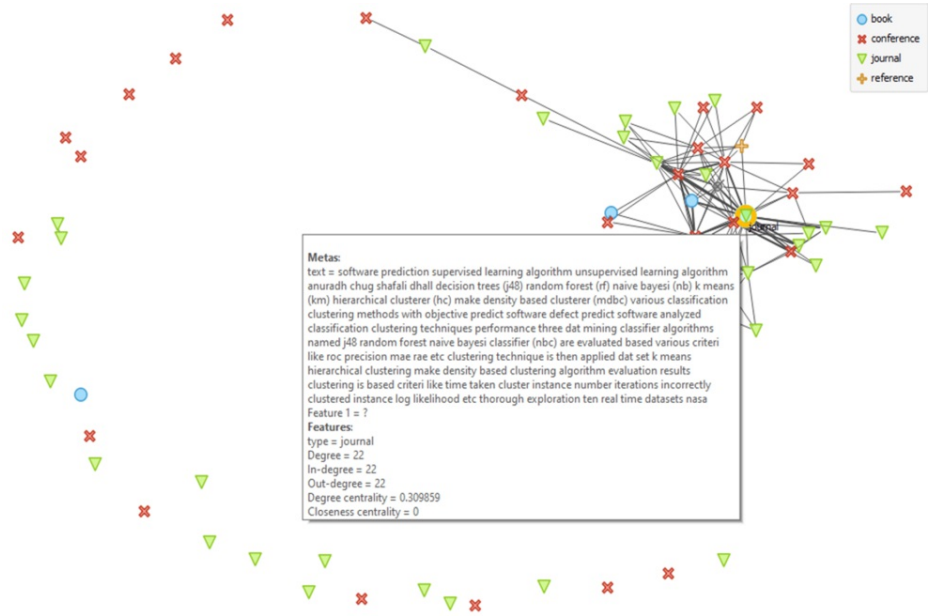

FIGURE 7: Study node with largest degree centrality

Conclusions

The need to review research articles to unravel similarities in methodologies, techniques, concepts, etc., through SRE is the main focus of this study. A graph analytics framework is proposed to deploy degree centralities for constructing a network of research semantics. The methodology is evaluated on 72-no textual abstracts extracted from the topic, keywords, aim, and methodology part of research articles that use machine learning to predict software defects. A five-relation threshold is set to build sub-graphs with semantic relationships, and the experimental results present a graph network of 72 nodes and 130 edges from an input of 72:46,945 tokenized textual corpus. The experimental results show that 59.7% of the entire articles making up the primary studies have a minimum of five lexical semantic relations with neighboring nodes, each with varying degree centrality figures, while 40.2% of the primary studies, with null degree centrality values, are largely either unique in their approach or fail to capture their topic, methodology, techniques, aims, etc., within the captured part of their abstracts.

The article with the largest degree centrality returned a 22-no edge with neighboring articles, indicative of the semantic relations existing between their studies. Prominent topics and techniques in their respective studies include ensemble machine learning, software defect severity, feature selection, cross-project

metrics, a supervised classification approach to machine learning, and neural network approaches, among others. Isolated articles primarily discussed SA of software defect reports to predict software defect severity, using ensemble methods, support vector machines, and naïve Bayes. As observed, an average aggregate of primary studies precisely captures major topics like methodology, techniques, aims, etc., within the title, keyword, and aim sections of their abstracts. Future work would expand the scope of the primary study as well as the extent of feature extraction.

Additional Information

Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Taiwo O. Olaleye, Oluwasefunmi Arogundade, Olusola J. Adeniran, Oluwatomisin Ajayi, Wilson Ahiara, Dada A. Aborishade

Acquisition, analysis, or interpretation of data: Taiwo O. Olaleye, Oluwasefunmi Arogundade, Olusola J. Adeniran, Oluwatomisin Ajayi, Wilson Ahiara, Dada A. Aborishade

Drafting of the manuscript: Taiwo O. Olaleye, Oluwasefunmi Arogundade, Olusola J. Adeniran, Oluwatomisin Ajayi, Wilson Ahiara, Dada A. Aborishade

Critical review of the manuscript for important intellectual content: Taiwo O. Olaleye, Oluwasefunmi Arogundade, Olusola J. Adeniran, Oluwatomisin Ajayi, Wilson Ahiara, Dada A. Aborishade

Supervision: Oluwasefunmi Arogundade, Olusola J. Adeniran, Dada A. Aborishade

Disclosures

Conflicts of interest: In compliance with the ICMJE uniform disclosure form, all authors declare the following: **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work. **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

References

1. Sneh Lata, Loar R: Text clustering and classification techniques using data mining . International Journal on Future Revolution in Computer Science & Communication Engineering. 2018, 4:859-864.
2. Jongeling R, Sarkar P, Datta S, Serebrenik A: On negative results when using sentiment analysis tools for software engineering research. Empirical Software Engineering. 2017, 22:2543-2584. 10.1007/s10664-016-9493-x
3. Aisha ID, Wulandhari LA: Topic prediction modelling on social media content using machine learning . Indonesian Journal of Electrical Engineering and Computer Science. 2024, 33:207-217. 10.11591/ijeecs.v33.i1.pp207-217
4. Olaleye T, Ajayi F, Aromolaran A, Solanke I, Akintunde S, JohnBosco A: Semantic relation evaluation of data science articles using network of mention. 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), Lagos, Nigeria. 2022, 1-9. 10.1109/NIGERCON54645.2022.9803141
5. Silva G, Ferreira R, Lins RD, Cabral L, Oliveira H, Simske SJ, Riss M: Automatic text document summarization based on machine learning. DocEng '15: Proceedings of the 2015 ACM Symposium on Document Engineering. 2015, 191-194. 10.1145/2682571.2797099
6. Kapanipathi P, Thost V, Patel S, et al.: Infusing knowledge into the textual entailment task using graph convolutional networks. Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34:8074-8081. 10.1609/aaai.v34i05.6318
7. Poliak A: A survey on recognizing textual entailment as an NLP evaluation [PREPRINT] . arXiv:2010.03061v1. 2020, 10.48550/arXiv.2010.03061
8. Olaleye TO, Arogundade OT, Misra S, Abayomi-Alli A, Kose U: Predictive analytics and software defect severity: A systematic review and future directions. Scientific Programming. 2023, 2023:6221388. 10.1155/2023/6221388
9. Geng ZQ, Chen GF, Han YM, Lu G, Li F: Semantic relation extraction using sequential and tree-structured LSTM with attention. Information Sciences. 2020, 509:183-192. 10.1016/j.ins.2019.09.006
10. Jin Z, Yang Y, Qiu X, Zhang Z: Relation of the relations: A new paradigm of the relation extraction problem [PREPRINT]. arXiv:2006.03719. 2020, 10.48550/arXiv.2006.03719
11. Rim K, Tu J, Lynch K, Pustejovsky J: Reproducing neural ensemble classifier for semantic relation extraction in scientific papers. Proceedings of the Twelfth Language Resources and Evaluation Conference. 2020, 5569-5578.
12. Zeng S, Xu R, Chang B, Li L: Double graph based reasoning for document-level relation extraction [PREPRINT]. arXiv:2009.13752v1. 2020, 10.48550/arXiv.2009.13752

13. Karaa WBA, Alkhammash EH, Bchir A: Drug disease relation extraction from biomedical literature using NLP and machine learning. Mobile Information Systems. 2021, 2021:9958410. 10.1155/2021/9958410
14. Kruiper R, Vincent JF, Chen-Burger J, Desmulliez MP, Konstas I: In layman's terms: Semi-open relation extraction from scientific texts [PREPRINT]. arXiv:2005.07751v2. 2020, 10.48550/arXiv.2005.07751
15. Nisar MU, Fard A, Miller JA: Techniques for graph analytics on big data . 2013 IEEE International Congress on Big Data, Santa Clara, CA, USA. 2013, 2379-7703. 10.1109/BigData.Congress.2013.78
16. Wang H, Qin K, Lu G, Yin J, Zakari RY, Owusu JW: Document-level relation extraction using evidence reasoning on RST-GRAPH. Knowledge-Based Systems. 2021, 228:107274. 10.1016/j.knosys.2021.107274
17. Geng Z, Zhang Y, Han Y: Joint entity and relation extraction model based on rich semantics . Neurocomputing. 2021, 429:132-140. 10.1016/j.neucom.2020.12.037
18. Kaur A, Jindal SG: Text analytics based severity prediction of software bugs for apache projects . International Journal of System Assurance Engineering and Management. 2019, 10:765-782. 10.1007/s13198-019-00807-8
19. Tan Y, Xu S, Wang Z, Zhang T, Xu Z, Luo X: Bug severity prediction using question-and-answer pairs from stack overflow. Journal of Systems and Software. 2020, 165:110567. 10.1016/j.jss.2020.110567
20. Aquil MAI, Ishak WHW: Predicting software defects using machine learning techniques . International Journal of Advanced Trends in Computer Science and Engineering. 2020, 9:6609-6616. 10.30534/ijatcse/2020/352942020
21. Gai J, Zheng S, Yu H, Yang H: Software defect prediction based on weighted extreme learning machine . Multiagent and Grid Systems. 2020, 16:67-82. 10.3233/mgs-200321
22. Alsawalqah H, Hijazi N, Eshtay M, Faris H, Radaideh AA, Aljarah I, Alshamaileh Y: Software defect prediction using heterogeneous ensemble classification based on segmented patterns. Applied Sciences. 2020, 10:1745. 10.3390/app10051745
23. Kukkar A, Mohana R, Nayyar A, Kim J, Kang BG, Chilamkurti N: A novel deep-learning-based bug severity classification technique using convolutional neural networks and random forest with boosting. Sensors. 2019, 19:2964. 10.3390/s19132964
24. Balogun AO, Basri S, Abdulkadir SJ, Hashim AS: Performance analysis of feature selection methods in software defect prediction: A search method approach. Applied Sciences. 2019, 9:2764. 10.3390/app9132764
25. Baarah A, Aloqaily A, Salah Z, Zamzeer M, Sallam M: Machine learning approaches for predicting the severity level of software bug reports in closed source projects. International Journal of Advanced Computer Science and Applications. 2019, 10:1-11. 10.14569/ijacsa.2019.0100836
26. Huda S, Abdelrazek LM, Ibrahim A, Alyahya S, Al-Dossari H, Ahmad S: An ensemble oversampling model for class imbalance problem in software defect prediction. IEEE Access. 2018, 6:24184-24195. 10.1109/access.2018.2817572
27. Arar OF, Ayan K: A feature dependent naive Bayes approach and its application to the software defect prediction problem. Applied Soft Computing. 2017, 59:197-209. 10.1016/j.asoc.2017.05.043
28. Arcelli FF, Marco Z: Code smell severity classification using machine learning techniques . Knowledge-Based Systems. 2017, 128:43-58. 10.1016/j.knosys.2017.04.014
29. Singh VB, Misra S, Sharma M: Bug severity assessment in cross project context and identifying training candidates. Journal of Information & Knowledge Management. 2017, 16:1750005. 10.1142/S0219649217500058
30. Malhotra R: An empirical framework for defect prediction using machine learning techniques with Android software. Applied Soft Computing. 2016, 49:1034-1050. 10.1016/j.asoc.2016.04.032
31. Kaur P, Singh C: A systematic approach for bug severity classification using machine learning's text mining techniques. International Journal of Computer Science and Mobile Computing. 2016, 5:523-528.
32. Seiffert C, Khoshgoftaar TM, Van Hulse JF: An empirical study of the classification performance of learners on imbalanced and noisy software quality data. Information Sciences. 2014, 259:571-595. 10.1016/j.ins.2010.12.016
33. Ren J, Qin K, Ma Y, Luo G: On software defect prediction using machine learning . Journal of Applied Mathematics. 2014, 2014:1-8. 10.1155/2014/785435
34. Malhotra R: Comparative analysis of statistical and machine learning methods for predicting faulty modules. Applied Soft Computing. 2014, 21:286-297. 10.1016/j.asoc.2014.03.032
35. Gayathri M, Sudha A: Software defect prediction system using multilayer perceptron neural network with data mining. International Journal of Recent Technology and Engineering. 2014, 3:54-59.
36. Khoshgoftaar TM, Gao K, Napolitano A, Wald R: A comparative study of iterative and non-iterative feature selection techniques for software defect prediction. Information Systems Frontiers. 2013, 16:801-822. 10.1007/s10796-013-9430-0
37. Naidu MS, Geethanjali D: Classification of defects in software decision tree algorithm . International Journal of Engineering Science and Technology. 2013, 5:1332-1340.
38. Bowes D, Hall T, Petrić J: Software defect prediction: Do different classifiers find the same defects? . Software Quality Journal. 2017, 26:525-552. 10.1007/s11219-016-9353-3
39. Sun Z, Song Q, Zhu X: Using coding-based ensemble learning to improve software defect prediction . IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2012, 42:1806-1817. 10.1109/tsmcc.2012.2226152
40. Malhotra R, Vidushi: Severity prediction of software vulnerabilities using textual data . Proceedings of International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications. Advances in Intelligent Systems and Computing. Gunjan, VK, Zurada JM (ed): Springer, Singapore; 2021. 1245:453-464. 10.1007/978-981-15-7234-0_41
41. Prabha C, Shivakumar N: Software defect prediction using machine learning techniques . 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India. 2020, 728-733. 10.1109/ICOEI48184.2020.9142909
42. Shaikh S, Changan L, Malik MR, Khan MA: Software defect-prone classification using machine learning: A virtual classification study between LibSVM & LibLinear. 2019 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS), Karachi, Pakistan. 2019, 1-6.

10.1109/MACS48846.2019.9024799

43. Gupta H, Kumar L, Neti LBM: An empirical framework for code smell prediction using extreme learning machine. 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India. 2019, 189-19. 10.1109/IEMECONX.2019.8877082
44. Wang F, Ai J, Zou Z: A cluster-based hybrid feature selection method for defect prediction . 2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS), Sofia, Bulgaria. 2019, 1-9. 10.1109/QRS.2019.00014
45. Ha DA, Chen TH, Yuan SM: Unsupervised methods for software defect prediction. SoICT '19: Proceedings of the 10th International Symposium on Information and Communication Technology. 2019, 49-55. 10.1145/3368926.3369711
46. Kukkar A, Mohana R: A supervised bug report classification with incorporate and textual field knowledge . Procedia Computer Science. 2018, 132:352-361. 10.1016/j.procs.2018.05.194
47. Fei W, Xiao-Yuan J, Ying S, Jing S, Lin H, Fangyi C, Yanfei S: Cross-project and within-project semisupervised software defect prediction: A unified approach. IEEE Transactions on Reliability. 2018, 67:581-597. 10.1109/tr.2018.2804922
48. El-Shorbagy SA, El-Gammal WM, Abdelmoez WM: Using SMOTE and heterogeneous stacking in ensemble learning for software defect prediction. ICSIE '18: Proceedings of the 7th International Conference on Software and Information Engineering. 2018, 44-47. 10.1145/3220267.3220286
49. Mihaylov M: Predicting the resolution time and priority of bug reports: A deep learning approach . Department of Computer and Information Sciences University of Strathclyde. 2019, 1-58.