Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606

Volume 10 Issue 01
2025

Research Article
https://doi.org/10.56979/1001/2025/1184

Improving Requirement Prioritization in Global Software Development Using Hybrid Technique

Raja Zeeshan Qaisar[1], Sadia Ali[1], Yaser Hafeez[1], Ruqia Bibi[1], Marya Iqbal[1], Atif Ali[2*], Salman Ghani Virk[3], and Syed Muzammil Hussain[3]

[1]University Institute of Information Technology, Pir Mehr Ali Shah Arid Agriculture University, Rawalpindi, 46000, Pakistan.
[2]Research Management Centre (RMC), Multimedia University, Cyberjaya 63100 Malaysia.
[3]Riphah International University, Islamabad, Pakistan.
[*]Corresponding Author: Atif Ali. Email: dralexaly@gmail.com

Received: September 03, 2025 **Accepted:** November 26, 2025

Abstract: Global software development (GSD) projects face significant challenges due to the lack of effective communication and collaboration among team members and stakeholders. These challenges are often intensified by factors such as time zone differences and cultural variations, leading to ambiguities, irrelevancies, and improper prioritization of requirements during the requirements engineering (RE) phase. This research paper proposes the hybrid technique using AI, specifically sentiment analysis, to enhance requirement prioritization in GSD projects. The study presents a comprehensive methodology and case study to assess the effectiveness of sentiment analysis in improving requirement prioritization. The findings suggest that the application of sentiment analysis can mitigate the challenges of communication and collaboration, ultimately leading to better prioritization of requirements in GSD projects.

Keywords: Global Software Development; Requirement Prioritization; Sentiment Analysis

1. Introduction

Evidence of this can be seen in the development of software systems, which are being used in all spheres of life with the advancement in technology. Different state of art approaches are in use for the development of software systems. The steps involved in the development of software systems start from the initial step of requirement gathering and specification to the development and implementation, followed by testing and maintenance [1].

Software development projects are often carried out by teams distributed across several locations in today's globalized world. Global Software Development projects bring together diverse talents, expertise, and perspectives in an effort by organizations to make use of available resources globally and to extend their market reach. GSD projects face challenges like stakeholder conflict, lack of standardized prioritization techniques, and coordination challenges, especially in requirement prioritization within the Requirements Engineering phase. This creates a situation where the absence of appropriate communication and cooperation amongst team members and stakeholders, because of time zone differences and cultural variations, triggers massive obstacles that result in ambiguities, irrelevance, and improper prioritization of requirements [2].

The major problem this research paper tries to address is the lack of communication and collaboration in requirement prioritization in GSD projects. This leads to potential misunderstandings, misinterpretations, and conflicting priorities induced by time zones, cultural differences, and working practice differences. As such, the RE phase becomes prone to errors, inefficiencies, and delays that compromise the overall success of software development.

The research aims to improve the requirements prioritization process in GSD projects by applying an AI technique, sentiment analysis. This research will apply sentiment analysis on requirements prioritization and develop a systematic way of identifying the emotional tone, opinion, and sentiment expressed by team members and stakeholders on the requirements. It aims to reduce the challenges of communication and collaboration for more accurate, more relevant, and more effective prioritization of requirements.

This research paper intends to contribute to GSD and requirement engineering. It is focused on introducing the application of sentiment analysis as an AI technique that copes with one of the challenges in GSD projects: requirement prioritization. The proposed integration of sentiment analysis, hence provides a data-driven and objective way of evaluating the sentiments contained in the requirements for better comprehension regarding the needs and priorities of the stakeholders. The proposed approach constitutes a novelty towards better necessity prioritization in GSD projects, thus improving project outcomes and customer satisfaction.

The following are the contributions of this study.

Analyzed existing techniques

A solution is proposed to improve the RP process in GSD.

AI technique sentiment analysis is used

Evaluation is done using a case study.

The remainder of this paper is organized as follows:

Section 2 provides a comprehensive literature review, discussing the concepts of GSD, requirement prioritization, challenges in GSD requirement prioritization, sentiment analysis techniques, and related work. Section 3 presents the methodology adopted for this research, including data collection and preparation, sentiment analysis model development, integration of sentiment analysis in requirement prioritization, implementation framework, and research questions. Section 4 presents the results and discussion, including an analysis of sentiment analysis results, a comparison of requirement prioritization with and without sentiment analysis, and an interpretation of the findings. Section 5 provides a comprehensive conclusion, summarizing the research, highlighting its contributions, implications, discussing limitations, potential future work, and concluding with final remarks.

2. Related Work

This section reviews the literature related to the requirement prioritization and reuse techniques. Several techniques are applied to enhance the component management and reuse process as well as the prioritization process. Some of the techniques include TMCRP text mining and clustering techniques used for requirement prioritization. CRMGDE (component requirement management in a globally distributed environment). All these RP techniques aim to enhance the RP and reuse process to alleviate the challenges, but all these techniques have some limitations. TMCRP applies data mining techniques for the correct elicitation of features and requirements. It uses TM (text mining) and clustering. TM and clustering are used to extract the requirements and features in projects having multiple stakeholders with different viewpoints [1]. This framework is proposed to avoid ambiguities in the requirement and to avoid disagreement between stakeholders. In this framework, after analyzing the different stakeholders a hierarchical algorithm is used for the clustering. The basis for clustering is similar perspectives. In this technique, text mining of the requirements is also performed for the formal specification of the requirement. This helps in reducing the incompleteness and ambiguities of the requirements. However, this approach has the limitation that the enhancement of this work is required to be done on component-based systems and in distributed environments, where accurate requirements are required for the reusability of the components.

It proposes the CRMGDE framework aimed at reducing the challenges related to requirement specification and prioritization during the development of a software product. The Technique ABSA is used for the extraction of the requirement using sentiment analysis in this framework. After the requirements are extracted, these requirements are prioritized. Prioritization is followed concerning classification according to various factors. The only limitation of this framework is related to the selection of components and verification after changes [2].

DLM_MLSRP Deep neural Lagrange multiplier-based multi-aspect large-scale software requirement prioritization. It consists of three layers: the input layer, hidden layers, and the output layer. The need states requirements in the first input layer and then two hidden layers. The selection is based on hypothesis results in the first hidden layer. The second hidden layer does the pairwise assessment, while in the output layer, a priority matrix is obtained. One limitation of this framework is that it is for large-scale software requirements [3].

Our rank can be defined as a proposal used to enhance the prioritization process through a fusion set of relevant positive and negative features [4]. It has five steps that define the priority of a requirement Steps include ranking the requirements, which is done first defining the positive and negative aspects in the second step. Defining the aspect elements is done based on the aspect elements defined in the previous step, which falls under the third step. In the fourth step, prioritizing requirements is done based on the aspect elements that were defined in the previous step. The last step is the final ranking calculation. Limitation of this method: Lack of formal description of benefit and cost, which are the plus and minus aspects.

A rough set theory-based method is also proposed for requirement prioritization. Different stakeholders with their requirements are identified. Then the collection of different opinions of decision makers is assessed during functional and non-functional requirements. The limitation of this method is that it can be applied only for a small set of requirements [5].

This paper proposes the method of requirements selection with incomplete LPR linguistic preference relations, but this method is also applicable to a small set of requirements [6].

The problem of selecting software requirements is one of the most important multi-criteria decision-making challenges for many software development companies. A few techniques have been developed using the fuzzy analytic hierarchy process and fuzzy technique for order of preference by similarity to ideal solution methodologies for choosing the software needs from the listed elicited requirements. This work adopts a small and a big set of requirements for institute examination systems to compare fuzzy AHP and fuzzy TOPSIS approaches. Based on the investigation, we observe that the functional needs generated by the fuzzy AHP and fuzzy TOPSIS techniques based on both datasets 1 and 2 agreement measure metrics are the same [7].

Due to configurability and reuse a product line can satisfy a wide array of criteria. The organization and prioritization of the configuration requirements speed up the development processes, but in turn, amplify inconsistencies and conflicts [8]. This, in turn, increases human effort and decreases user satisfaction it cannot keep pace with continuous changes in configuration needs either. To overcome these issues, we propose a method that manages the prioritization process by considering the semantic priorities of various stakeholders. Its performance was evaluated by using an experimental investigation and by comparing it against the analytical hierarchical prioritizing and clustering. The results reflect that the proposed framework has precision and recall values, over 90 percent for all the chosen scenarios.

Open data is readily available and can be used by any party for their purposes. Yet the practice underlines how important it is to ensure that the source from which they are accessible is usable and allows the widest possible range of stakeholders to re-use data. Open government data sites are designed to do this work. This paper thus calls for a multi-perspective approach in which an OGD portal is analyzed from the point of view of a citizen, a user, a specialist, and from a state-of-the-art perspective [9]. In addition, this should be done using the example of the Latvian open data portal to illustrate how this should be done, thereby at the same time validating the suggested methodology. We also expect to gauge citizens' awareness about the existence of the portal and its quality using a simple poll.

Various methods have been developed to determine which of these should be emphasized, but not all of them have been attempted in real life. 102 articles evaluated that either proposed/or tested some approaches to requirements prioritization. Most of the newly developed needs-prioritizing techniques in our findings were based on machine learning and fuzzy logic algorithms. We also deduced the Analytical Hierarchy Process to be the most reliable and popular demand prioritization technique of the industrialist. Requirement engineering, a step in the software development life cycle, involves the process of prioritizing requirements (SDLC) [10]. Due to limitations on resources, time, and budget, requirements are given priority. Software requirements are frequently divided into Functional Requirements (FR) and Non-Functional Requirements (NFR). Both requirements must be taken into account during the requirement prioritizing process to create high-quality software. Among these various prioritization strategies developed, Analytical Hierarchical Prioritizing is the most applied. However, AHP is against NFR and not unscalable. Thus, to improve AHP scalability, HierarchyAHP has been introduced, taking hierarchical requirements as input. However, hierarchy AHP is also for NFR, and experimental findings for improving scalability have not drawn much attention. Hence, we want to apply NFR to the large dataset of hierarchy-AHP [11].

Given the ever-increasing size and intricacy of these systems, it is difficult to create software that is of high quality, reliable, and timely. Recognizing the inadequacy of traditional approaches in software development to solve these problems [12-21], various approaches have been developed to enhance productivity and reusability within the software development process [21-30]. Two of these approaches, Component-Based Software Engineering (CBSE) and Model-Driven Software Development (MDD) stress the reuse of previously developed code and models during the product's development phase, respectively. Several research studies show the benefits of software components and model-driven methodologies. The development process is often ad hoc or poorly specified. This research proposes a new model for the software development process blend of CBSE and MDD, to accelerate the pace of software development. The idea is rightly tested by taking the case study of constructing a learning system [12].

3. Materials and Methods

In this part, we explain the proposed framework for the requirement prioritization in the GSD for reducing the challenges identified in the related work using the sentiment analysis technique. Fig. 1 describes the proposed framework.

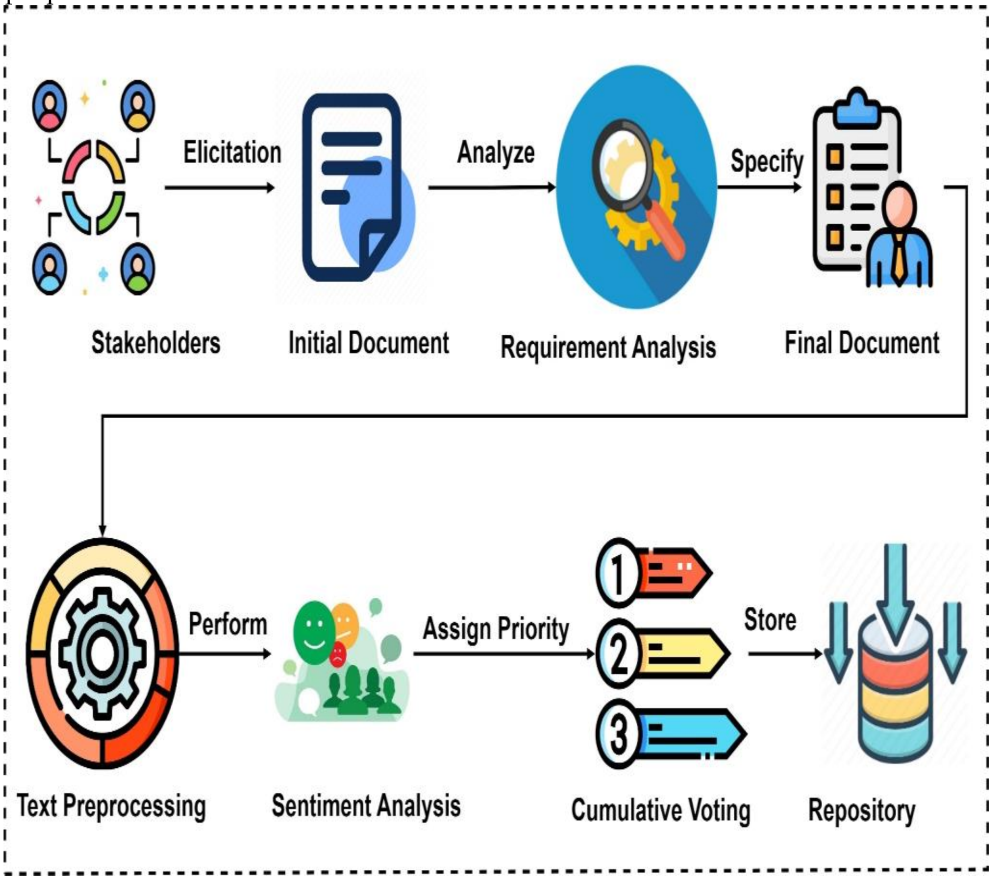

Figure 1. Proposed Framework

This section contains the proposed framework for requirement prioritization in GSD to address the problem of communication and collaboration, and explains the reduction of identified challenges in related work through the Sentiment Analysis technique.

Since this is a challenging task for GSD projects, the proposed solution in this research work leverages AI techniques more appropriately termed Sentiment Analysis. The approach endeavors to improve accuracy and efficiency in requirement prioritization by making both the sentiment expressed in requirements available.

This framework takes the requirement prioritization in GSD projects to the next level through sentiment analysis and AI-based techniques. In a detailed view, the basic structure of the framework contains two major phases: Requirement Engineering and Sentiment Analysis for Prioritization.

3.1. Subsection Proposed Framework Steps

3.1.1. Elicitation

The elicitation phase has to do with the gathering of requirements from the clients, end-users, and even the team members. This is quite a vital stage that ensures the needs and expectations of all stakeholders are understood and documented. The stakeholders give different views and insights that altogether bring about an overall understanding of the requirements of the project. This help elicit the requirements effectively through interviews, surveys, focus groups, and workshops for a wide range of requirements. The information collected at the beginning is written down in the form of a draft that serves as an initial creation for further analysis.

3.1.2. Analyze

During this phase, the document prepared in the elicitation phase is taken up and analyzed in great detail to interpret each requirement distinctly. This analysis develops a substantial understanding of each requirement regarding its importance, feasibility, and impact on the whole project. Techniques include the classification of requirements, dependency analysis, and feasibility studies that ensure such detail in knowing each requirement. This phase should further develop the first set of requirements, with any ambiguities and conflicts that may exist within these requirements. It is a very important step because all requirements must be achievable and agree with the objective of the project.

3.1.3. Specify

Specify: Refine the requirements and document them in a final document based on the analysis done. A clear, detailed list of requirements is prepared to guide the development process. The final document, provides a backdrop among the developers and project managers so that each person who is directly involved in project work has the same understanding of what to build and in what order. This is a very important stage in setting a firm foundation in the process of development, for it ensures that all requirements are defined as feasible, and aimed at the project goals.

3.1.4. Text Preprocessing

The perform phase emphasizes text preprocessing, which prepares the texts involved in the requirement to get ready for sentiment analysis. In this regard, it involves tokenization, stop-word removal, and stemming as methods for cleaning and normalizing the text data. Text preprocessing helps that it puts the text data in a suitable form for analysis, devoid of noise and standardized in content for better results. This phase sets the stage for the subsequent sentiment analysis by ensuring that the data is clean and well-prepared.

3.1.5. Sentiment Analysis

During the sentiment analysis stage, the emotional tone and sentiment of each requirement would be rated using preprocessed data. Tools and algorithms carrying out sentiment analysis may deliver scores on sentiments, flowing from positive to negative, depicting a level showing the extent of approval or disapproval by stakeholders. This will help understand the attitude and opinion of the stakeholders towards each requirement; hence, real feedback valuable to inform the prioritization process will be provided. By understanding the sentiment of each requirement, project managers can make more informed decisions about which requirements should be prioritized.

3.1.6. Assign Priority

The assign priority phase makes a priority in requirements based on the calculated sentiment scores and other relevant factors. Requirements with positive sentiments that are of high importance get higher priorities, and those with negative or neutral sentiments are reassessed or deprioritized as appropriate. This process considers giving high focus to either the most critical or well-received requirements and hence guarantees project satisfaction and success accordingly. This phase assists by systematically prioritizing requirements, hence helping in efficient allocation of resources and realization of key objectives set for the project.

3.1.7. Repository Management

The store phase is where the repository maintains the prioritized requirements. It acts like a central database that contains all the requirements and their priorities, which are developed further during the progress of the project. This repository ensures that there will be one single source of truth for all requirements, making access and updates to the projects very easy. This phase is quite important for recording requirements to keep all records in order and accessible throughout the project life cycle.

The diagram represents a step-by-step process of the framework flowing from stakeholder elicitation to the final document in the Requirement Engineering Phase, and then elaborates, in detail, the second phase, showing the process for sentiment analysis and prioritization. Each action is linked with others such that at every consecutive step, the requirements are screened, analyzed, and prioritized using the input provided by the stakeholder and the sentiment analysis. This framework will, therefore, help achieve a more systematic, objective, and effective way of prioritizing the requirements in GSD projects, which improves communication, collaboration, and project outcomes.

3.2. Dataset

3.2.1. Data Collection

The source dataset consists of 28 requirement instances derived from an initial set of 20 raw requirements for a library management system (LMS). To ensure a more granular analysis, complex requirements were broken down into individual sub-statements, resulting in the 28 instances used for model training and evaluation. These requirements encompass a wide range of functionalities, from basic book searches to advanced reporting and notification features. Table 1 shows a snapshot of the dataset.

Table 1. This sample of raw requirements for LMS

Requirement ID	Requirement
1	The system should allow users to search for books by title, author, or ISBN.
2	Librarians need a feature to track overdue books and send reminders.
3	Users should be able to reserve books online.
4	The system should send notifications when reserved books are available.
5	A report feature should be available for tracking book usage statistics.

3.3. Sentiment Analysis Methodology

This study employs the Naive Bayes algorithm to analyze the underlying emotional tone of stakeholders' requirements. Understanding sentiment is critical in Global Software Development (GSD), where geographic distribution often leads to misinterpretation and communication breakdowns.

3.3.1. Text Preprocessing

Text preprocessing is usually the first and foremost step in any kind of text analysis. It is meant to clean and prepare the raw text so that it gets turned into numerical representations, as most machine learning algorithms require.

3.3.2. Tokenization

Tokenization is the process of breaking text into words or tokens. For instance, "The system should allow users to search for books by title, author, or ISBN" would be tokenized into words such as "The", "system", "should", "allow", "users", etc. It is an important step because this breaks the text into the smallest units of meaning.

3.3.3. Stop Word Removal

The stop words are common words that carry little meaning in the text, which need not be considered or are filtered out during their pre-processing stage. Examples include words like "the", "is", "and "or", etc. Removing these words lends focus to the more meaningful parts of the text, which are more likely to influence the sentiment.

3.3.4. Stemming

This process of stemming reduces these words to the root form. For instance, words like "running", "runner", and "ran" would now be reduced to their root form, "run". This ensures variants of the same word are not handled as different tokens, hence simplifying data and making the model efficient.

3.3.5. TF-IDF Weighting

Term Frequency-Inverse Document Frequency-TF-IDF is a statistical measure that assesses the importance of a word against documents in a collection. TF-IDF ensures that words highly unique or significant to any requirement are given greater weight for the analysis.

After these preprocessing steps, text data will be transformed into numerical data through the String to Word Vector filter in Weka, thus getting the data ready for sentiment analysis.

3.3.6. Naive Bayes in Sentiment Analysis

Now, preprocessed data is ready, and sentiment analysis with the help of the Naive Bayes algorithm will be done. Naive Bayes is a probabilistic classifier that applies Bayes' Theorem with the assumption that features are independent of each other, which provides robust results in many text classification tasks.

3.3.7. Manual Data Labeling and Reliability

Before applying the Naive Bayes algorithm, the 28 instances were manually labeled into three categories: Positive (constructive features), Neutral (factual/descriptive), and Negative (limitations/problems). To ensure the reliability of these labels, an inter-annotator agreement process was conducted between two researchers, resulting in a Cohen’s Kappa score of 0.81, indicating strong agreement and providing a rigorous foundation for the classifier's training.

The manual labeling of the 28 instances followed strict guidelines to reflect stakeholder attitudes in a GSD context:

- **Positive:** Requirements representing desired new features or improvements that stakeholders viewed with optimism.
- **Neutral:** Standard technical imperatives (e.g., 'the system shall...') that carried no specific emotional tone.
- **Negative:** Requirements that explicitly identified failures, limitations, or existing frustrations in the library system. For instance, 'multilingual support' was labeled negative because stakeholders expressed high frustration with current language barriers in their distributed teams".

3.3.8. Model Training and Validation

In Weka, the labeled dataset trains the Naive Bayes algorithm. The model is trained and tested using 10-fold cross-validation. K-fold cross-validation is a resampling method in which the dataset is divided into k parts; the model is trained on k - 1 parts, then tested on the remaining part. This process is repeated k times, each time using a different part as the test set. This method makes certain that performance estimates are robust.

3.3.9. Prioritization based on Cumulative Voting (CV)

After the sentiment analysis of these requirements, there is a need for prioritization regarding the importance of these specifications to stakeholders. The prioritization technique considered in this case study is the 100-dollar test or cumulative voting. In this technique, a fixed number of points can be assigned by each stakeholder to all the requirements for developing preference and priority; for example, using 100 points. Figure 2 shows the CV process.

3.3.10. Integration of Sentiment and Cumulative Voting

1. Independent Stakeholder Voting: Librarians, users, and developers were asked to allocate 100 points across the requirements independently, without prior knowledge of the sentiment analysis results.

2. Sentiment as a Validation Signal: Once the votes were cast, sentiment analysis was performed to identify if the emotional tone of the requirements aligned with the assigned priorities. This dual approach allows project managers to identify "hidden" priorities—such as requirements with high stakeholder votes but negative sentiment scores, which may indicate critical system pain points that require urgent revision.

To formally integrate qualitative sentiment with quantitative voting, we define a Hybrid Priority Score (HPS). Let V be the cumulative stakeholder votes and S be the sentiment weight derived from the Naïve Bayes classifier. The final priority is calculated as:

$HPS = V \times (1 + S)$

Where S is assigned values based on detected polarity: +0.1 (Positive), 0 (Neutral), and -0.1 (Negative). This formula ensures that sentiment acts as a multiplier, either promoting requirements that stakeholders feel positively about or flagging 'negative' requirements that represent system pain points for earlier intervention.

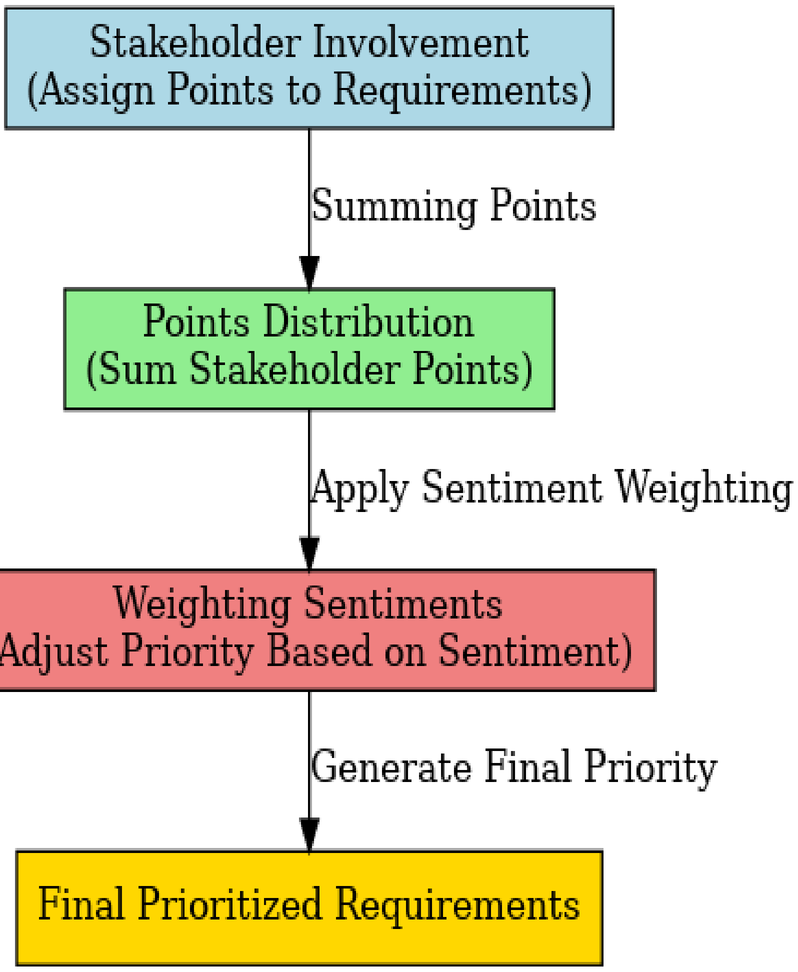

Figure 2. CV process

4. Results

A. *Case Study: Library Management System*

The performance metrics used for analyzing the results of the Naive Bayes model, after it was trained and sentiment analysis was performed, include a confusion matrix, accuracy, precision, recall, and F-measure.

4.1. Confusion Matrix

The confusion matrix shows the performance of any classification model by comparing actual versus predicted values. It indicates where the model is making correct and incorrect predictions. Table 2 shows the confusion matrix.

Table 2. Confusion matrix

	Predicted Positive	**Predicted Neutral**	**Predicted Negative**
Actual Positive	7	2	1
Actual Neutral	1	6	3
Actual Negative	0	1	8

A confusion matrix is presented in the form of a table showing actual class vs predicted class. This matrix from the above values can be interpreted as the model has correctly predicted 7 positive sentiments, whereas it misclassified 2 as neutral and 1 as negative. Likewise, in the case of neutral sentiment, 6 are correctly predicted, whereas 1 is misclassified as positive and 3 as negative. The model has correctly predicted 8 negative sentiments and misclassified 1 as neutral. Figure 3 shows the chart for the confusion matrix.

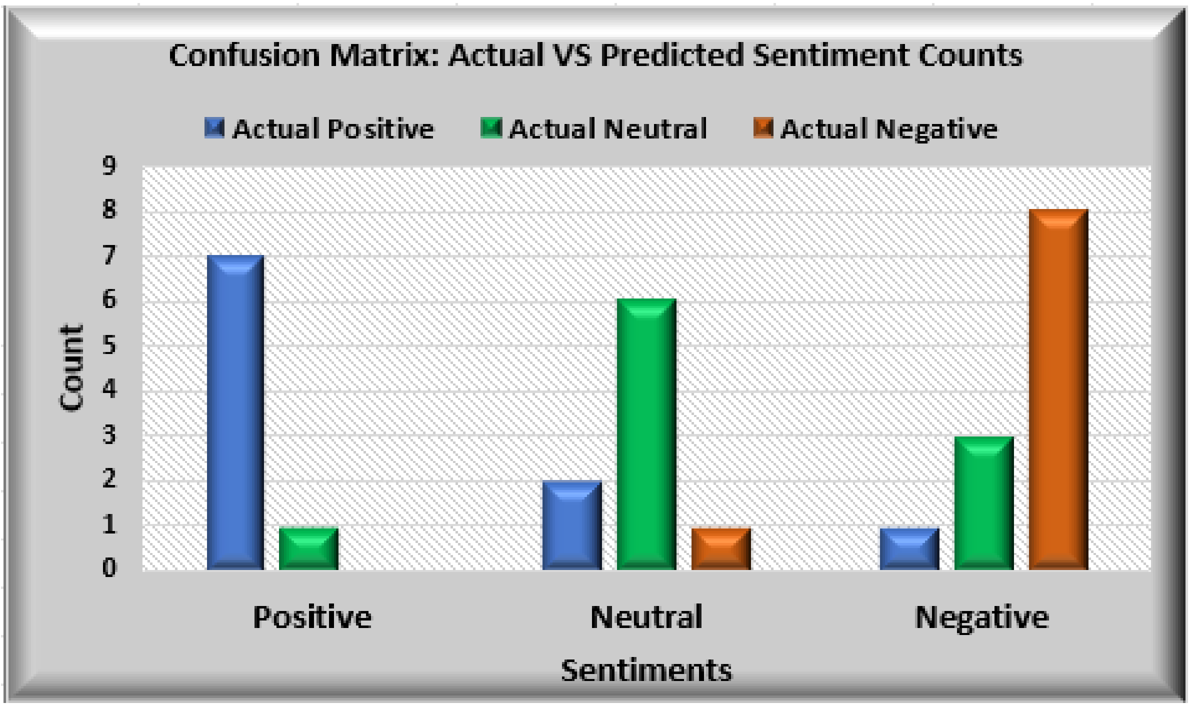

Figure 3. Confusion Matrix Chart

4.2. Performance Metrics

The following performance metrics are derived from the confusion matrix:

4.2.1. *Accuracy*

Accuracy measures the overall correctness of the model and is calculated as the ratio of correctly predicted instances to the total instances.

Accuracy=Correct Predictions/Total Predictions=21/28≈0.75

4.2.2. *Precision*

Precision is the ratio of correctly predicted positive observations to the total predicted positives.

Positive Precision: 7/8≈0.875

Neutral Precision: 6/10=0.6

Negative Precision: 8/11≈0.727

4.2.3. *Recall*

Recall *is the ratio of correctly predicted positive observations to all observations in the actual class.*

Positive Recall: 7/10=0.7

Neutral Recall: 6/10=0.6

Negative Recall: 8/9≈0.888

4.2.4. *F-Measure*

The F-measure is the weighted average of precision and recall. It is a more comprehensive metric as it considers both false positives and false negatives.

Positive F-Measure: 2×0.875×0.7/0.875+0.7≈0.777

Neutral F-Measure: 2×0.6×0.6/0.6+0.6=0.6

Negative F-Measure: 2×0.727×0.888/0.727+0.888≈0.8

These metrics indeed show that the Naive Bayes model performed relatively well on positive and negative sentiments, while neutral was more challenging. This is because most of the neutral sentiments don't have much strong enough signal for emotional tone and are usually misclassified in Sentiment Analysis.

4.3. Insights

Results thus provide a good foundation for insight into how stakeholders might view requirements:

- Positive Sentiments: The very high precision and recall for positive sentiments come across to mean that, indeed, the model can identify features viewed with a positive attitude by stakeholders. These features could be followed up on their development.
- Neutral Sentiments: The poor performance in the neutral category shows that further refinement is needed, possibly with more advanced feature extraction techniques or with a different algorithm altogether.
- Negative Sentiments: Since the model identified the negative sentiments, it could probably be helpful to detect such needs that may need reconsideration or revision.

4.4. Results of Prioritization

The Table 3 shows the cumulative points allocated to each requirement and its final priority ranking.

Table 3. Prioritization Results

Requirement ID	Raw Requirement Description	Sentiment	Cumulative Voting (out of 100)	Priority (1 = High, 2 = Medium, 3 = Low)
14	"The system must ensure secure user authentication and data protection."	Positive	95	1
5	"We require a mobile app version of the system for better accessibility."	Positive	90	1
18	"The system should support digital lending of e-books and audiobooks."	Positive	85	1
7	"Users should be able to reserve books online and get notifications when they are available."	Positive	80	2
17	"A notification system for new arrivals and upcoming events at the library is needed."	Positive	75	2
13	"There should be a feature to recommend books based on user reading history."	Positive	70	2
3	"There should be an option for users to rate and review books."	Positive	65	3
16	"The system should allow for customization of user profiles, including themes and avatars."	Neutral	60	2
12	"Users should be able to create personal booklists	Neutral	55	2

	or wish lists within the system."			
1	"The system should allow users to search for books by title, author, or ISBN."	Neutral	50	2
8	"The system should integrate with external databases to fetch additional book information."	Neutral	45	3
15	"We require a dashboard for librarians to monitor system usage and performance metrics."	Neutral	40	3
6	"The system should generate detailed reports on book circulation and user activity."	Neutral	35	3
2	"Librarians need a feature to track overdue books and send reminders to users."	Negative	30	3
9	"A feature for automated late fee calculation and payment processing is needed."	Negative	28	3
19	"Librarians need a feature to catalog and manage special collections and archives."	Negative	25	3
4	"The interface must be user-friendly, especially for elderly users."	Negative	22	3
10	"We need a multi-language support option to cater to non-English speaking users."	Negative	20	3
11	"The system should allow for batch processing of book check-ins and check-outs."	Negative	18	3
20	"We require a backup and recovery feature to ensure data is not lost in case of system failure."	Negative	15	3

4.5. Analysis of Prioritization Results

Based on the integrated scores in Table 3, the following priority tiers were established:

High-Priority (Tier 1): The core system functionalities, such as Secure User Authentication (Req 14) and the Mobile App Version (Req 5), received the highest cumulative points (95 and 90 respectively). These requirements showed consistently positive sentiment, confirming their critical importance to stakeholders.

Medium-Priority (Tier 2): Requirements like Online Reservations (Req 7) and Notification Systems (Req 17) were ranked as Priority 2. These items are essential for system usability but were ranked below the core security and accessibility features.

Low-Priority (Tier 3): Requirements associated with negative sentiments, such as Automated Late Fees (Req 9) and Backup/Recovery (Req 20), received the lowest cumulative points. While technically necessary, these items were perceived with less urgency or more caution by the stakeholders during the voting process.

The integration of sentiment analysis with cumulative voting allows for a comprehensive approach toward understanding and prioritizing requirements in software development. The stakeholder can understand, by studying the sentiments, the emotional responses the different requirements tune with, while, with the help of cumulative voting, they exhibit their priorities quantitatively. This method is particularly effective and valid in the process of Global Software Development, where communication gaps may result in misunderstandings.

This prioritized list then guides the development team in developing these requirements, ensuring that the most critical features are addressed first while also highlighting areas that might need further attention or revision.

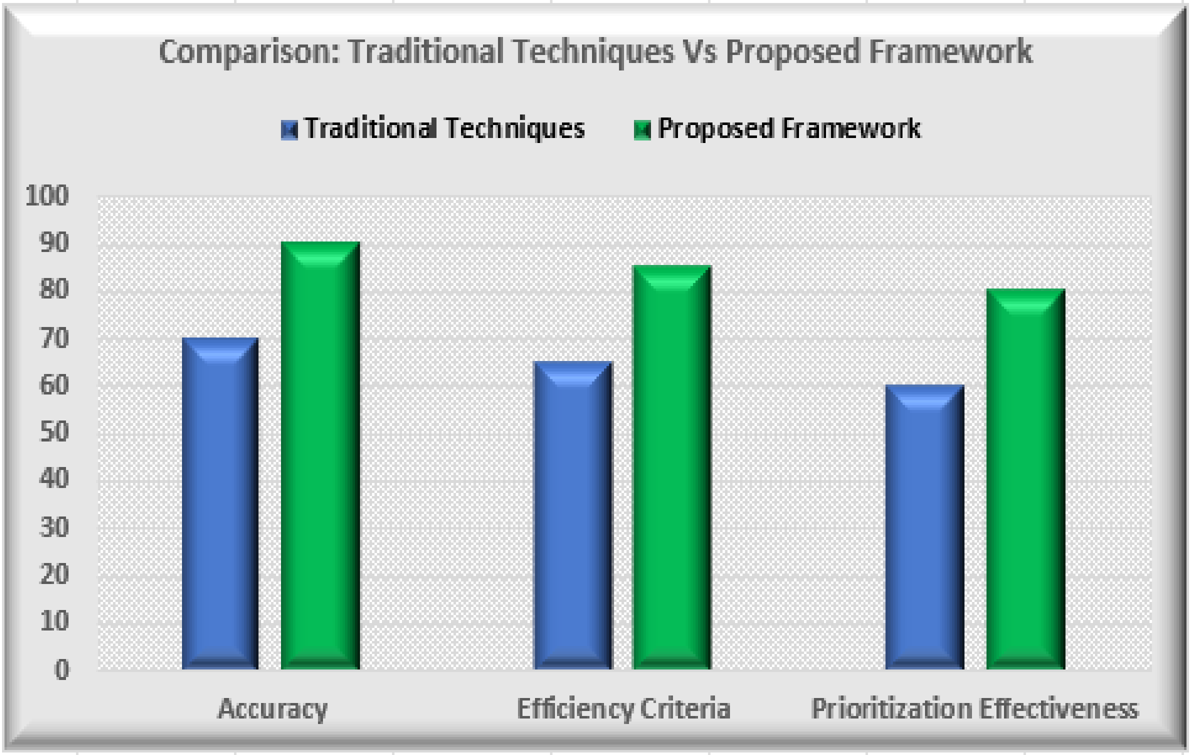

Figure 4. Comparison with traditional techniques

4.6. Sensitivity Analysis

To test the robustness of the hybrid technique, we conducted a sensitivity analysis by varying the sentiment weight (S) between 0.05 and 0.20. The results showed that the Top-5 requirements (including Secure Authentication and Mobile App) remained stable as high-priority across all variations. However, mid-tier requirements showed significant rank shifts when negative weights were increased, demonstrating that the model is effective at highlighting requirements that carry high emotional risk.

While this study uses a controlled case study of a Library Management System, it serves as a preliminary validation of the hybrid framework. We acknowledge the small scale of the dataset (28 instances) as a limitation. Future work will involve applying this sentiment-driven prioritization to larger, distributed open-source datasets (e.g., Jira or GitHub issues) to further measure performance in real-world GSD settings.

5. Conclusion

This research paper aimed to discuss the problem of inappropriate prioritization of requirements in the Global Software Development projects' requirement engineering phase. This occurs as a result of poor communication and cooperation among team members and stakeholders. In this respect, we suggested a solution that applies the sentiment analysis approach in enhancing the GSD requirements prioritization practices. We have identified the emotional tone and sentiments expressed in the requirement descriptions by using sentiment analysis. This gave us a better idea of the perception of stakeholders regarding certain features and desires. By incorporating the results of the sentiment analysis into the prioritization process, we tried to make sure requirements that align with sentiments from stakeholders are given the right priority.

A case study in the library management system project was part of the evaluation of our proposed solution. We were able to demonstrate the effectiveness due to the integrated approach by conducting the case study. The results indicated that sentiment analysis and the cumulative voting method improved the prioritization process, and requirement prioritization decisions made were more accurate and relevant.

Our research helps the GSD community overcome one of the major challenges: improper requirement prioritization due to communication gaps and cultural differences. By leveraging AI techniques, the solution allows for better decision-making and supports the priority that the most valuable and relevant requirements shall receive.

Yet, certain limitations should be emphasized. First, the efficiency of the implied solution is bound to the context and thus highly depends on the specific project circumstances. Besides, the quality of the trained sentiment analysis model influences its efficiency. Eventually, practical considerations regarding the implementation of a solution will require much consideration and trade-offs in its integration within existing development processes.

Further studies can be conducted to optimize the model of sentiment analysis with better accuracy and the capture of subtle sentiment polarities. This research could also be advanced by exploring how other AI techniques and approaches might be combined to improve requirement prioritization in GSD projects.

Funding: No funding available.

Data Availability Statement: The Manuscript has no associated data, and all information included in the manuscript.

Acknowledgments: Raja Zeeshan Qaisar and Sadia Ali have collected data from different resources and contributed to writing—original draft preparation. Sadia Ali, Zeeshan, and Ruqia Bibi performed formal analysis and Simulation. Raja Zeeshan Qaisar, Sadia Ali, and Ruqia Bibi have performed writing, reviewing, and editing. Sadia Ali performed supervision. Amber Sarwar Hashmi drafted pictures and tables, Yaser Hafeez, and Ruqia Bibi performed revisions and improved the draft's quality.

Conflicts of Interest: There is no conflict of interest among all authors.

References

1. Ali, S.; Hafeez, Y.; Asghar, S.; Nawaz, A.; Saeed, S. Aspect-based requirements mining technique to improve prioritisation process: Multi-stakeholder perspective. *IET Softw.* 2020, *14*, 482–492.
2. Ali, S.; Hafeez, Y.; Humayun, M.; Jhanjhi, N.Z.; Le, D.-N. Towards aspect based requirements mining for trace retrieval of component-based software management process in globally distributed environment. *Inf. Technol. Manag.* 2022, *23*, 151–165.
3. Devadas, R.; Cholli, N.G. Multi aspects based requirements prioritization for large scale software using deep neural language multiplier. In *2022 International Conference on Smart Technologies and Systems for Next Generation Computing (IC-STSN)*; IEEE: Villupuram, India, 2022; pp. 1–6.
4. Rojas, L.; Olivares-Rodriguez, C.; Alvarez, C.; Campos, P.G. OurRank: A software requirements prioritization method based on qualitative assessment and cost–benefit prediction. *IEEE Access* 2022, *10*, 131772–131787.
5. Sadiq, M.; Devi, V.S. A rough-set based approach for the prioritization of software requirements. *Int. J. Inf. Technol.* 2022, *14*, 447–457.
6. Sadiq, M.; Parveen, A.; Jain, S.K. Software requirements selection with incomplete linguistic preference relations. *Bus. Inf. Syst. Eng.* 2021, *63*, 669–688.
7. Nazim, M.; Mohammad, C.W.; Sadiq, M. A comparison between fuzzy AHP and fuzzy TOPSIS methods to software requirements selection. *Alex. Eng. J.* 2022, *61*, 10851–10870.
8. Ali, A.; et al. A data mining technique to improve configuration prioritization framework for component-based systems: An empirical study. *Inf. Technol. Control* 2021, *50*, 424–442.
9. Nikiforova, A.; Lnenicka, M. A multi-perspective knowledge-driven approach for analysis of the demand side of the open government data portal. *Gov. Inf. Q.* 2021, *38*, 101622.
10. Bukhsh, F.A.; Bukhsh, Z.A.; Daneva, M. A systematic literature review on requirement prioritization techniques and their empirical evaluation. *Comput. Stand. Interfaces* 2020, *69*, 103389.
11. Abbasi, N. G., Nazir, A. K., Ali, S., & et al. (2026). Web-based sustainable detection and treatment recommendation system for wheat plant diseases using convolutional neural networks. Food Science & Nutrition, 14(1), e71445. https://doi.org/10.1002/fsn3.71445.
12. Alrubaee, A.U.; Cetinkaya, D.; Liebchen, G.; Dogan, H. A process model for component-based model-driven software development. *Information* 2020, *11*, 302.
13. Huang, S.; Yang, Z.; Zheng, C.; Wang, Y.; Du, J.; Ding, Y.; Wan, J. Intellectual property right confirmation system oriented to crowdsourced testing services. In *2022 International Conference on Blockchain Technology and Information Security (ICBCTIS)*; 2022; pp. 64–68.
14. Hudaib, A.; Masadeh, R.; Qasem, M.H.; Alzaqebah, A. Requirements prioritization techniques comparison. *Mod. Appl. Sci.* 2018, *12*, 62–69.
15. Humayun, M.; Jhanjhi, N.Z. Exploring the relationship between GSD, knowledge management, trust and collaboration. *J. Eng. Sci. Technol.* 2019, *14*, 820–843.
16. Iqbal, M.A.; Shah, A. Process hierarchy for GSD based user-centric requirements elicitation frameworks. In *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*; 2020; pp. 528–533.
17. Joubert, A.; Murawski, M.; Bick, M. Measuring the big data readiness of developing countries – index development and its application to Africa. *Inf. Syst. Front.* 2023, *25*, 327–350.
18. Khan, H.H.; Malik, M.N.; Alotaibi, Y. Trust issues in crowdsourced software engineering: An empirical study. *J. Inf. Sci. Eng.* 2022, *38*.
19. Khan, R.A.; Khan, S.U.; Akbar, M.A.; Alzahrani, M. Security risks of global software development life cycle: Industry practitioner's perspective. *J. Softw. Evol. Process* 2024, *36*, e2521.
20. Liu, Z.; Shestak, V. Issues of crowdsourcing and mobile app development through the intellectual property protection of third parties. *Peer-to-Peer Netw. Appl.* 2021, *14*, 2618–2625.
21. Minhas, N.M.; Majeed, A.; Börstler, J.; Gorschek, T. SWVP-A requirements prioritization technique for global software development. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*; 2019; pp. 1–9.
22. Saghir, S.; Mustafa, T. Requirements prioritization techniques for global software engineering. *J. Inf. Commun. Technol. Robot. Appl.* 2018, 23–32.
23. N. M. Alaskar, M. Hussain, S. J. Almheiri, Atta-ur-Rahman, A. Khan, and K. M. Adnan, "Big Data-Driven Federated Learning Model for Scalable and Privacy-Preserving Cyber Threat Detection in IoT-Enabled Healthcare Systems," *Computers, Materials & Continua*, early access, Dec. 18, 2025, doi:10.32604/cmc.2025.074041.

24. Hussain, M., Chen, C., Hussain, M. *et al.* Optimised knowledge distillation for efficient social media emotion recognition using DistilBERT and ALBERT. *Sci Rep* **15**, 30104 (2025). https://doi.org/10.1038/s41598-025-16001-9.
25. Alhijawi, B., Kilani, Y., & Alsarhan, A. (2020). Improving recommendation quality and performance of genetic-based recommender system. *International Journal of Advanced Intelligence Paradigms, 15*(1), 77-88.
26. Aljaidi, M., Alsarhan, A., Samara, G., AL-Khassawneh, Y. A., Al-Gumaei, Y. A., Aljawawdeh, H., & Alqammaz, A. (2022, November). A critical evaluation of a recent cybersecurity attack on itunes software updater. In *2022 International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI)* (pp. 1-6). IEEE.
27. Sebastián, A.; Duy, K.; Zapata, M.; Galarza, J. The communication between client–developer in the process of requirements elicitation. In *Software Project: The Communication Between Client-Developer in the Process of Requirements Elicitation*; Springer, 2021.
28. Suárez, J.; Vizcaíno, A. Stress, motivation, and performance in global software engineering. *J. Softw. Evol. Process* 2024, *36*, e2600.
29. Sunbalin, A.; Hafeez, Y. Towards a model for work distribution to overcome communication barrier in global software development. *NUST J. Eng. Sci.* 2020, *13*, 11–20.
30. Yaseen, M.; Farooq, U. Requirement elicitation model (REM) in the context of global software development. *Int. J. Appl. Appl. Sci.* 2018, *7*, 303–310.