electronics

MDPI

Article

An Automated Framework for Prioritizing Software Requirements

Behnaz Jamasb [1], Seyed Raouf Khayami [1], Reza Akbari [1] and Rahim Taheri [2,*]

[1] Department of Computer Engineering and IT, Shiraz University of Technology, Shiraz 71557-13876, Iran; b.jamasb@sutech.ac.ir (B.J.); khayami@sutech.ac.ir (S.R.K.); akbari@sutech.ac.ir (R.A.)
[2] PAIDS Research Centre, School of Computing, University of Portsmouth, Portsmouth PO1 3HE, UK
* Correspondence: rahim.taheri@port.ac.uk

Abstract: Requirement Engineering (RE) is a critical phase in software development, integral to the successful execution of projects. The initial stage of RE involves requirement elicitation and analysis, where the prioritization of requirements is critical. Traditional methods of requirement prioritization (RP) are diverse, each presenting unique challenges. In response to the challenges of traditional methods, this paper proposes an entirely automated framework designed to eliminate the disadvantages associated with excessive stakeholder involvement. This innovative framework processes raw natural language inputs directly, applying a three-phase approach to systematically assign priority numbers to each requirement. The first phase preprocesses the input to standardize and prepare the data, the second phase employs advanced machine learning algorithms to analyze and rank the requirements, and the third phase consolidates the results to produce a final prioritized list. The effectiveness of this method was tested using the RALIC (Replacement Access, Library, and ID Card) dataset, a well-known benchmark in the field of requirement engineering. The results confirm that our automated approach not only enhances the efficiency and objectivity of the prioritization process but also scales effectively across diverse and extensive sets of requirements. This framework represents a significant advancement in the field of software development, offering a robust alternative to traditional, subjective methods of requirement prioritization.

Keywords: software requirement; RALIC; machine learning; requirement engineering

Academic Editors: Paolo Ciancarini, Ireneusz Miciuła, Łukasz Radliński, Aleksander Jarzębowicz and Włodzimierz Wysocki

Received: 20 January 2025
Revised: 8 March 2025
Accepted: 17 March 2025
Published: 20 March 2025

Citation: Jamasb, B.; Khayami, S.R.; Akbari, R.; Taheri, R. An Automated Framework for Prioritizing Software Requirements. *Electronics* **2025**, *14*, 1220. https://doi.org/10.3390/electronics14061220

1. Introduction

Requirement engineering (RE) is a critical phase in the Software Development Life Cycle (SDLC), involving the gathering, analysis, and documentation of software requirements from users. This phase is important as it defines the framework within which software functionalities are designed and implemented. An integral component of RE is requirement prioritization, a decision-making process in which software engineers assess stakeholder demands to establish the sequence in which software requirements should be addressed [1].

Advancements in machine learning have significantly improved the capability of prioritizing software requirements, facilitating data-driven decisions that consider factors such as stakeholder value, technical feasibility, and cost. Modern agile methodologies, in conjunction with tools that support distributed teams, have streamlined the collaborative aspects of prioritization. In addition, the integration with DevOps practices promotes real-time alignment with ongoing development objectives [2].

Ethical considerations are now more prevalent in the prioritization process, with an increasing emphasis on sustainability and inclusivity. This shift is particularly notable

in organizations committed to environmental and social responsibility. Additionally, the advent of low-code platforms and collaborative tools has democratized the prioritization process, enabling even non-technical stakeholders to participate effectively [3].

However, a common challenge in RE is the discrepancy between the number of requirements proposed by stakeholders and the realistic capacity for feature implementation within time and resource constraints. Consequently, not all proposed features can be developed in the immediate term; some are deferred to future releases [4].

Current RP techniques face several significant challenges. These include a lack of scalability, which hampers the ability to manage a large volume of requirements; a deficiency in time efficiency, particularly evident when prioritizing extensive sets of requirements; and insufficient processes for quantifying and prioritizing stakeholder inputs, as well as a general absence of automation [5].

Traditionally, the identification of key requirements is performed manually, introducing the potential for human errors to affect the outcomes. Moreover, the role of stakeholders varies within a project, with each one interpreting requirements based on their unique perspective, thereby influencing the system requirements differently [6]. The effective selection of stakeholders involved in the prioritization process is crucial for accurately determining the sequence of requirements. However, existing techniques often lack a clear methodology for distinguishing and selecting stakeholders appropriately, which leads to inconsistencies in requirement prioritization (RP) [5].

While traditional RP techniques, such as the Analytic Hierarchy Process (AHP), perform adequately with a limited number of requirements, they struggle to cope with the complexity and volume that are characteristic of modern software products [5]. These techniques typically require stakeholders to engage in extensive comparisons among requirements, a process that becomes increasingly time-consuming as the number of requirements escalates [5,7]. This paper introduces a machine learning framework designed to alleviate the workload on stakeholders and enhance the efficiency of prioritizing large and complex sets of requirements. The framework aims to streamline the prioritization process by automating the comparison tasks, thus addressing the critical scalability issues identified in existing RP methods.

This paper addresses the limitations of existing requirement prioritization methods by proposing an automated machine learning approach. Traditional manual techniques often fail to effectively manage large volumes of requirements and heavily rely on stakeholder involvement, which can introduce bias and inefficiency. In contrast, the method developed in this study is engineered to handle extensive sets of requirements efficiently while minimizing the necessity for direct stakeholder input in the prioritization process. Additionally, the proposed approach recognizes and accommodates the diverse roles and influences of stakeholders, ensuring that their contributions are aligned with their specific responsibilities and expertise, rather than being treated uniformly. This alignment enhances the accuracy and relevance of the prioritization outcomes.

In this paper, we present a framework that inputs software requirements expressed in natural language directly into the system, without any prior modification or processing by stakeholders. After an initial preprocessing phase, these requirements are clustered and prioritized. Traditional methods typically require manual intervention from stakeholders to analyze and prioritize requirements, often failing to leverage the capabilities of natural language inputs in their raw form. The proposed approach reduces the reliance on stakeholder involvement, which significantly enhances the speed of the prioritization process, streamlining operations and improving efficiency in handling large datasets.

The proposed framework primarily focuses on Requirement Prioritization (RP) but originates from Requirement Engineering (RE). The framework is designed to

automate requirement prioritization by clustering and ranking requirements without stakeholder intervention.

- Requirements Engineering (RE)—Covers the entire process of gathering, analyzing, specifying, and validating requirements.
- Requirement Prioritization (RP) [Main Focus]—A critical subtask of RE, determining which requirements should be addressed first.

The paper starts from RE because prioritization is an essential step after requirement elicitation. It shifts focus to RP by proposing a fully automated prioritization method, eliminating stakeholder bias and manual ranking. Unlike traditional methods where stakeholders assign numerical weights, this method clusters and prioritizes requirements from raw natural language inputs.

In the subsequent sections, the structure of the paper is outlined as follows: Section 2 reviews related work, providing a concise overview of several key studies in the field. Section 3 provides the study's background, including a concise explanation of the RALIC dataset, which forms the basis of the analysis, highlighting the project's structure and its relevance. Additionally, it offers an overview of the algorithms and methods employed in the research. Section 4 outlines the proposed framework, detailing its architecture and phases. Section 5 showcases the results of implementing the framework on the RALIC dataset, along with a comparative analysis against the baseline approach [7]. Section 6 highlights the benefits of adopting the proposed framework, and finally, Section 7 presents the conclusion of the study.

2. Related Work

Numerous studies have proposed varied methods for requirement prioritization, each addressing specific challenges associated with the task. Here, we briefly explore a selection of these methodologies, primarily using the RALIC dataset.

In [5], the semi-automated Requirements Prioritization (RP) technique, SRPTackle, was introduced. This technique targets the scalability issues, excessive dependency on expert intervention, and the extensive time requirements prevalent in existing methodologies. It integrates an RPV formulation function with clustering algorithms (K-means and K-means++), and a Binary Search Tree (BST) to provide a holistic solution. SRPTackle was evaluated through seven experiments on the extensive RALIC benchmark dataset, affirming its effectiveness. However, its assumption of independent requirements limits its applicability in scenarios with interdependent requirements.

Reyad et al. [8] tackled the clustering of stakeholders for system requirement selection and prioritization within requirements engineering. Their Genetic K-Means Adaption Algorithm for Requirements Engineering (GKA-RE) automatically determines the optimal number of stakeholder clusters, enhancing clustering by dynamically adjusting initial seeds. The effectiveness of GKA-RE was validated on two datasets from the RALIC project, showing improvements over traditional K-means approaches in stakeholder prioritization.

The authors in [9] developed a Recommender System (RS) for Software Requirement Prioritization using Intuitionistic Fuzzy Sets (IFSs). This system addresses the challenges of information overload and stakeholders' uncertainty by employing collaborative filtering with IFSs. The system was tested using a large dataset, including RP data from the RALIC project, demonstrating enhanced prioritization effectiveness. This work prioritizes requirements based solely on a single criterion—requirement importance—when assigning ratings in the dataset. While this approach simplifies the prioritization process, it may fail to account for the complexities of real-world scenarios.

Soo Ling Lim et al. [6] introduced StakeRare, a method leveraging social networks and collaborative filtering to identify and prioritize stakeholders and their requirements in

large software projects. An extensive empirical evaluation of StakeRare was conducted, comparing it against traditional methods. Despite its efficiency in reducing effort during the rating process, its effectiveness might be compromised by the quality of initial requirements, especially in projects where stakeholders are less aware of their requirements. StakeRare assumes that stakeholders provide recommendations and ratings honestly. However, malicious stakeholders may manipulate responses for personal gain. Changes in the stakeholders' recommendations or changes in the requirements' ratings mean that the requirements have to be reprioritized.

The study in [7] presented a systematic literature review (SLR) on RP within software development, addressing research questions about the application contexts, frequency of use, and publication trends. It also discussed future directions and validity threats, pointing out the need for increased automation, intelligent features, and the more effective quantification of stakeholder contributions in RP techniques.

The study in [10] proposed a semi-automated, data-driven RP approach for Software Product Line (SPL) development, aiming to minimize stakeholder effort by using AI techniques such as Natural Language Processing (NLP) and sentiment analysis. This approach was designed to tackle the complexities of managing common requirements and dependencies in SPLs, with a focus on automating the prioritization process. Lastly, the paper [11] explored a clustering-based technique using the k-means algorithm to prioritize large-scale software requirements efficiently. This method addressed scalability and rank reversals common in traditional techniques and was validated on the RALIC dataset. Future research was suggested to explore hybridization with other algorithms for further optimization.

1. StakeRare: This approach is a stakeholder-based RP technique that utilizes a weighting mechanism based on stakeholders' influence and expertise. While StakeRare effectively incorporates stakeholder perspectives, it suffers from biases due to subjective prioritization and the varying expertise levels of stakeholders. In contrast, our proposed method eliminates stakeholder dependency, reducing subjective bias and ensuring consistency through automated processing of raw requirements.
2. SRPTackle: SRPTackle integrates a hybrid approach that combines stakeholder preferences with prioritization models. Similarly to StakeRare, it involves active participation from domain experts to rank requirements. In the SRPTackle method, each stakeholder is assigned a Stakeholder Participation Value (SPV) based on their level of involvement. Using these SPVs, a Requirement Priority Value (RPV) is calculated for each requirement. The requirements are then classified into three predefined priority levels: high, medium, and low, according to their RPVs. Finally, the Binary Search Tree (BST) algorithm is used to sort the requirements.
   The foundation of this approach relies heavily on stakeholder-defined values, as RPV is determined based on the stakeholders' participation and their assigned initial weights for each requirement.
   In contrast, our proposed method eliminates the dependency on stakeholder input, ensuring a fully automated approach. Instead of relying on manually assigned weights, our framework leverages AI-driven clustering and prioritization, using BERT for contextual representation, K-Means for clustering, and centroid similarity for ranking. This automation not only reduces subjectivity but also enhances scalability, making it more suitable for large and complex requirement datasets.
3. Genetic K-Means: This method applies evolutionary optimization techniques to cluster stakeholders. The primary objective is to enhance the K-Means algorithm for more effective stakeholder clustering. While requirement prioritization is mentioned as a challenge in this approach, it is not actually performed; instead, the focus is solely on

clustering stakeholders. Our method, on the other hand, clusters the requirements, determines the optimal number of clusters using Elbow and Silhouette Score methods, ensuring an efficient and scalable clustering process without the complexity of genetic algorithms, and then prioritizing them.

All the mentioned methods initially assign numerical values to the requirements in various ways before ultimately prioritizing them.

However, in our proposed method, no numerical values are assigned to the requirements. Instead, the raw natural language requirements themselves serve as the real input to the framework. Both clustering and prioritization are performed directly on the requirements without any predefined numerical weighting, ensuring a more data-driven and automated approach.

In summary, while existing methods in the literature provide various frameworks for prioritizing software requirements, they often rely heavily on manual stakeholder input, which can be both time-consuming and susceptible to bias. Furthermore, these methods often struggle to handle natural language inputs effectively, requiring extensive preprocessing that may strip away the essential nuances needed for accurate prioritization. In contrast, the proposed method in this study introduces an automated approach that leverages machine learning to process requirements in their raw natural language form. This not only reduces the dependency on stakeholder interpretation and involvement but also significantly accelerates the prioritization process. By automating the initial stages of input handling and prioritization, this method enhances the scalability and efficiency of requirement engineering practices, offering a novel contribution to the field that addresses the critical gaps identified in the current literature. Requirement Prioritization (RP) involves ranking software requirements based on their importance, urgency, or other criteria to optimize resource allocation during development. Common Requirement Prioritization (RP) techniques are given as follows:

- Analytical Hierarchy Process (AHP): Uses pairwise comparisons to assign weights to requirements.
- MoSCoW Method: Categorizes requirements into Must have, Should have, Could have, and Do not have.
- The 100-Dollar Method: Stakeholders distribute 100 points (or dollars) among requirements based on importance.
- Cumulative Voting: Similar to the 100-dollar method but allows more flexibility in voting.
- Value-Based Prioritization: Prioritizes requirements based on business value and cost–benefit analysis.
- Hybrid Methods: Combine multiple techniques (e.g., AHP + MoSCoW) for more accurate prioritization.

We compare traditional prioritization methods in Table 1.

Table 1. Comparison of different traditional methods with our proposed method.

Aspect	**AHP (Analytic Hierarchy Process)**	**MoSCoW Method**	**MCDM (Multi-Criteria Decision Making)**	**SRP Tackle**	**Proposed Method**
Type	Traditional	Traditional	hybrid (multi-criteria)	Algorithmic	Automated clustering prioritization
Stakeholder Involvement	High (pairwise comparisons)	High (manual categorization)	High (weight-based)	High (SPV and RPV-based)	Low (automatic prioritization)
Prioritization Basis	Pairwise comparisons	Expert judgment	Weight-based decision criteria	Stakeholder rank and weights	Requirement clustering & Centroid similarity
Automation Level	Low (manual pairwise)	Low (manual)	Medium (criteria-based automation)	Medium (algorithm-driven sorting)	High (automated requirement categorization and prioritization)
Scalability	Low (complex for large datasets)	Medium (simple for small datasets)	Medium (depends on criteria)	High (scalable to many stakeholders)	High (scalable to large requirement sets)
Computational Complexity	High ($O(n^2)$ for comparisons)	Low (categorization-based)	Medium (depends on MCDM model)	Medium (BST sorting applied)	High (clustering and cosine similarity computation)
Ranking Granularity	Precise (numerical weights)	Subjective (categories: must, should, etc.)	Precise (weight-based)	Three levels (high, medium, low)	Cluster-based with fine-grained granularity
Main Limitation	Subjectivity and lack of precise comparisons	requires expert-dependency	stakeholder dependency	Computational overhead for clustering	Dependency on clustering

3. Background

This section provides an overview of the RALIC Project, detailing its objectives, stakeholder involvement, and data structuring. We also explore preprocessing techniques and various analytical methods relevant to our study, including BERT, K-Means clustering, and other evaluative metrics.

3.1. RALIC Project

The RALIC (Replacement Access, Library, and ID Card) project at University College London (UCL) was initiated to modernize access control systems by integrating them with library access and borrowing capabilities [6]. This large-scale project underwent two and a half years of development and has been in operation for over two years. The RALIC dataset categorizes software requirements at three abstraction levels: Project Objectives, Requirements, and Specific Requirements. It includes a 'rated profile' where stakeholders assess requirements on a scale from 0 (not important) to 5 (very important), with -1 denoting non-development requirements. This dataset encompasses ratings from 76 stakeholders across 439 ratings for 10 project objectives, 1514 ratings for 48 requirements, and 3113 ratings for 104 specific requirements [9]. Table 2 presents a summary of stakeholder ratings for the RALIC project.

Table 2. Summary of Stakeholder Ratings.

Project Objectives	**Rate**
Number of stakeholders provide more than one rating	76
Number of items	10
Number of ratings	439
Requirements	
Number of stakeholders provide more than one rating	76
Number of items	48
Number of ratings	1514
Specific Requirements	
Number of stakeholders provide more than one rating	76
Number of items	104
Number of ratings	3113

Key Aspects of the RALIC Project

- **Requirements:** The project aimed to enhance security and access control, design a multifunctional access card, and reduce operational costs.
- **Stakeholders:** More than 60 stakeholder groups from various UCL faculties and departments participated, highlighting a diversity of perspectives and, at times, conflicting requirements.
- **Ranking and Weighting:** Methods such as pairwise comparison and hierarchical organization were used to prioritize requirements at different levels.
- **Ground Truth:** Derived from extensive project documentation and validated through interviews with management-level stakeholders to ensure accuracy and completeness [5].

Lim's dissertation provides comprehensive documentation of the RALIC dataset, with appendices detailing its structure, stakeholder roles, requirement rankings, and prioritization methodologies. The RALIC project questionnaires define three distinct requirement ranking methods, each aligned with specific project goals: [12].

Based on the questionnaires provided in the RALIC project and their respective goals, there are three types of requirement ranking methods:

- Rate—A scoring system where requirements are assigned a value between -1 and 5 based on their importance (method used in this paper)
- Rank—A normalized ranking where values range between 0 and 1, indicating relative importance.
- Point—A distribution-based approach where stakeholders allocate a total of 100 points among different requirements, reflecting their perceived significance.

Each method captures different perspectives on requirement importance, allowing for a more comprehensive prioritization process.

Different stakeholders, based on their roles, expertise, and other parameters, assigned rankings to certain requirements. For example, Mike Dawson, who holds the role of Access and Security Services, provided ratings for 31 requirements, whereas Jason Ortiz, in the same role, rated only seven requirements. For example, Table 3 presents the list of stakeholders and their ranks within the Security and Access Systems role, along with the ranked requirements in this category. This table provides insight into how different stakeholders contribute to the prioritization process based on their assigned.

Table 3. Requirements specification.

ID	Short Description of Requirement	Rate (−1.5)
h.2	Include payment mechanism	1
h.3	Used for computer logon	1
g.2	Export data to other systems	5
g.3	Import data from other systems	5
g.1.2	Data should not be duplicated	4

Table 4 presents an example of five requirements that Mike Dawson has ranked. This example illustrates how an individual stakeholder evaluates and assigns rankings to requirements.

Table 4. Stakeholder roles and rankings.

Role Rank		Stakeholder Rank	
1	**Security and Access Systems**	**1**	**Mike Dawson**
		2	Jason Ortiz
		3	Nick Kyle
		4	Paul Haywood

3.2. Preprocessing

Preprocessing involves preparing raw data for further analysis, which includes tokenization, stemming, and stop word removal, enhancing the accuracy and efficiency of the prioritization process.

- **Stop word removal:** Common words such as “the”, “is”, and “and” are removed to reduce data dimensionality, although their removal can be context-dependent.
- **Lemmatization and stemming:** These processes reduce words to their base or dictionary form and root form, respectively, aiding in the consistency and performance of text analysis tasks.

3.3. BERT Model

BERT (Bidirectional Encoder Representations from Transformers) is a cutting-edge language model using the Transformer architecture, pre-trained through tasks like masked language modeling and next sentence prediction. This bidirectionality allows BERT to capture nuanced word relationships within sentences, making it highly effective for tasks like text classification and named entity recognition [10,13].

BERT is a Transformer-based [14] NLP model for natural language processing (NLP) tasks. Unlike traditional models that process text in a left-to-right or right-to-left manner, BERT is bidirectional, meaning it considers both previous and next words in a sentence for better context understanding.

3.3.1. Key Features of BERT

1. **Bidirectional context understanding:** Unlike traditional models (e.g., LSTMs or unidirectional transformers), BERT processes text in both directions simultaneously, capturing richer context.
2. **Masked Language Model (MLM):** Instead of predicting the next word (like GPT), BERT randomly masks some words in a sentence and tries to predict them using the surrounding context.

3. **Next-Sentence Prediction (NSP): black**BERT is pre-trained to determine whether two sentences logically follow each other, improving their ability to understand relationships between sentences.

3.3.2. Mathematical Formulation

Token Embeddings in BERT

Each input token x_i is converted into a vector representation:

$$E_i = T_i + P_i + S_i$$

where T_i is the token embedding (word representation), P_i is the positional embedding (captures word position in the sentence), and S_i is the segment embedding (distinguishes different sentences in NLP tasks).

Self-Attention Mechanism (Scaled Dot-Product Attention)

BERT relies on self-attention to weigh words differently based on their relevance. The attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, V are Query, Key, and Value matrices (learned representations of input words) and d_k is the dimension of the key vectors (scaling factor).

This allows BERT to dynamically focus on important words in a sentence.

Masked Language Model (MLM)

During training, 15% of input tokens are randomly masked, and BERT is trained to predict the missing word:

$$P(w_i | w_1, \ldots, w_{i-1}, \ldots, w_{i+1}, \ldots, w_n)$$

This forces BERT to understand the full context of a sentence.

Next Sentence Prediction (NSP)

BERT is trained on pairs of sentences to learn relationships. The loss function is binary cross-entropy:

$$L = -\sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

where y_i is 1 if the second sentence follows the first, otherwise 0.

3.4. K-Means Clustering Algorithm

K-Means is a robust method for data clustering based on similarity measures. It involves selecting initial centroids, assigning data points based on similarity, and iteratively refining centroids until convergence. Despite its simplicity, the method's effectiveness depends on the correct specification of cluster numbers and initial conditions. Advantages of K-Means include simplicity and computational efficiency, particularly for datasets with a high number of variables. K-Means finds applications in various domains, including market segmentation, document grouping, image compression, customer segmentation, and dynamic data trend analysis. Overall, K-Means clustering offers a straightforward yet powerful technique for data partitioning and analysis in diverse fields of applications [15]. Letter "k" represents the number of desired clusters. The algorithm iteratively assigns data points to clusters and calculates cluster centroids, which serve as the centers of each group.

K-means partitions a dataset into K distinct, non-overlapping clusters by minimizing the variance within each cluster. The goal is to group similar data points together while ensuring that the clusters are as compact as possible.

3.4.1. K-Means Algorithm Steps

1. **Select K:** Choose the number of clusters, K.
    - The Elbow Method or Silhouette Score is often used to find the optimal K.
2. **Initialize Centroids:** Select K points as initial cluster centroids.
3. **Assign Points to Clusters:** Each data point is assigned to the nearest centroid using the Euclidean distance.
4. **Update Centroids:** Compute the new centroid for each cluster by taking the mean of all points assigned to that cluster.
5. **Repeat Steps 3–4** until the centroids no longer change or a predefined number of iterations reached.

3.4.2. Mathematical Formula: Objective Function (Minimizing Inertia)

K-Means minimizes the sum of squared distances between each data point and its assigned cluster centroid:

$$J = \sum_{i=1}^{K} \sum_{x_j \in C_i} \|x_j - \mu_i\|^2$$

where K is the number of clusters, C_i is the set of points in cluster i, x_j is a data point, and μ_i is the centroid of cluster i. $\|x_j - \mu_i\|^2$ is the squared Euclidean distance between a data point and its cluster centroid.

The centroid acts as a "summary" or representative of the cluster, and the sentences are assigned based on their similarity to this representative. As the algorithm progresses, the centroids are refined to better represent the central theme or topic of the sentences in the cluster. The Cluster Centroid Similarity drives the entire process of assigning sentences to clusters and refining cluster boundaries. In K-means clustering, Cluster Centroid Similarity refers to the idea that each cluster is represented by a "centroid," which is the average of the data points assigned to that cluster. The algorithm uses this centroid to measure the similarity between each data point and the cluster. Here is how Cluster Centroid Similarity works in K-means, step by step:

- Initialization: Initially, the algorithm selects k data points as the centroids of the clusters. These centroids serve as the reference points for measuring similarity during the clustering process.
- Document Representation: For text data, each or sentence is converted into a vector representation, typically using techniques like word embeddings (e.g., BERT).
- Assignment Step: Each document is assigned to the cluster whose centroid is the most similar to it. The similarity between a document's vector and the centroid is often measured using cosine similarity or Euclidean distance. (Cosine similarity measures the angle between two vectors, and a smaller angle means the vectors are closer to each other.)
- The sentence is assigned to the cluster with the centroid that is closest in terms of the chosen similarity measure.
- Update Step: Once all sentences have been assigned to clusters, the centroid of each cluster is updated. The new centroid is the mean of the document vectors within that cluster. This new centroid is then used to measure similarity in the next iteration.

- Iteration: The algorithm repeats the assignment and update steps until convergence, meaning the centroids no longer change significantly or a maximum number of iterations is reached.

3.5. Elbow Method

This method assists in determining the optimal number of clusters by identifying the point where the Sum of Square Errors (SSE) decreases sharply, known as the 'elbow point'. It balances the model's complexity against performance, optimizing the number of clusters for effective analysis [16].

3.6. Silhouette Score

The Silhouette Score evaluates the quality of clustering by measuring both the cohesion within clusters and the separation between them. It provides a quantitative measure to assess the effectiveness of the clustering approach, ranging from -1 (poor clustering) to +1 (excellent clustering) [17].

3.7. Weighting Methods in Multi-Criteria Decision Making (MCDM)

Weighting methods in MCDM are crucial for reflecting the relative importance of criteria in decision-making processes. These methods integrate both subjective judgments and objective data, enhancing the decision-making process's accuracy and reliability [13].

3.8. Mean Absolute Error (MAE)

MAE is a straightforward metric in regression analysis that calculates the average magnitude of errors between predicted and actual values, offering a clear measure of prediction accuracy without the influence of error direction [18].

3.9. Precision, Recall, and F1-Score

Precision, Recall, and F1-Score are key evaluation metrics commonly used in ranking and classification tasks to measure the effectiveness of a system in retrieving relevant items.

- Precision represents the proportion of correctly identified high-priority requirements among the top-ranked results, indicating how accurate the framework is in selecting relevant requirements.
- Recall measures the proportion of actual high-priority requirements that were successfully retrieved by the framework, showing how well it captures all relevant requirements. There is often a trade-off between precision and recall, as increasing the number of retrieved items may improve recall but reduce precision.
- To balance these two metrics, F1-Score is used, which is the harmonic mean of precision and recall, providing a single measure of effectiveness. A high F1-score indicates that the framework performs well in both selecting relevant requirements and ensuring comprehensive retrieval, making it a crucial metric for evaluating automated requirement prioritization [19].

In this sections, the key concepts used in the proposed framework are briefly explained. BERT was utilized in Phase 1 of the framework for processing raw requirements. K-Means clustering was applied in Phase 2 to categorize the requirements into meaningful clusters. The Silhouette Score and Elbow Method were used to determine the optimal number of clusters. MCDM (Multi-Criteria Decision Making) was employed to implement the baseline method for comparison. MAE (Mean Absolute Error) was used to evaluate and compare the performance of the proposed method against the baseline study.

4. Proposed Method

The proposed method seeks to address the limitations of previous approaches by automating the entire process and eliminating the reliance on stakeholder opinions, which can be biased or inaccurate. This section describes the three-phase framework used in our approach, with each phase's output serving as the input for the subsequent phase.

4.1. Phase 1: Preprocessing and BERT Embedding

Phase 1 utilizes the BERT model, renowned for its efficacy with large text datasets. Prior to applying BERT, raw natural language requirements undergo several preprocessing steps, which are essential for cleaning and normalizing text data. The input for this phase consists of approximately 400 raw natural language requirements that have not been manually processed or categorized. These requirements vary in phrasing and sentence structure. The preprocessing steps include:

- Stopword removal: Removes commonly used but uninformative words.
- Lemmatization: Reduces words to their base or dictionary forms.
- Stemming: Removes suffixes from words to reduce them to their root or stem form.

After preprocessing, the BERT algorithm processes the requirements into contextual embeddings, capturing the semantic meanings and relationships between words and sentences. These embeddings are then utilized to cluster the requirements in the next phase.

4.2. Phase 2: Clustering with K-Means

This phase eliminates human intervention by employing the K-Means clustering algorithm to organize the processed requirements. To determine the optimal number of clusters, methods such as the Elbow Method and Silhouette Score are applied. Both methods suggested that 10 clusters provide a meaningful distribution for analysis and prioritization. The K-Means algorithm is chosen for its simplicity, efficiency, and clarity in handling large datasets. This clustering facilitates the analytical and prioritization tasks in the subsequent phase.

K-Means offers notable advantages, including simplicity (easy to implement for practical applications), speed (computationally efficient), and scalability (effective for large datasets with numerical values). Compared to other clustering techniques such as K-Medoids, Partitioning Around Medoids, CLARA (Clustering Large Applications), and Fuzzy Clustering, K-Means demonstrates superior efficiency [5]. Therefore, this study employs K-Means to cluster the requirements.

The K-Means algorithm is known to be sensitive to the selection of initial centroids. However, this issue can be mitigated by employing an appropriate initialization rather than relying on a random selection. In this study, the optimal number of centroids was determined using the Elbow Method and Silhouette Score. The selection of K-Means over alternative clustering techniques was based on multiple factors:

1. Efficiency and Scalability: K-Means is computationally efficient and scalable for large datasets, making it suitable for handling a high number of requirements. In contrast, Agglomerative Clustering has higher time complexity ($O(n^2)$ or $O(n^3)$ depending on the implementation), making it less efficient for large-scale requirement sets.
2. Interpretability: K-Means provides clear, well-defined clusters with centroids, which help in structuring the prioritization process. Hierarchical clustering methods like Agglomerative Clustering generates dendrograms, which are not as straightforward as defining ranked groups.

3. Requirement Nature: Since the requirements were transformed into numerical vector representations using BERT embeddings, K-Means was well suited for handling these representations. Density-based clustering techniques such as DBSCAN, which rely on proximity-based density estimation, were found to be less effective in this context.
4. Empirical Results: To ensure the suitability of K-Means, a comparative experiment was conducted with Agglomerative Clustering. The results showed that while Agglomerative Clustering could form meaningful hierarchical structures, it struggled with the high-dimensional embeddings generated by BERT and had higher computational complexity. Additionally, K-Means provided a more balanced distribution of requirements across clusters, making it preferable for subsequent prioritization.
5. Necessity of a Centroid-Based Clustering Method: A key requirement of the proposed framework was to prioritize requirements within each cluster using centroid similarity (Phase 3). Since K-Means inherently assigns a centroid to each cluster, it allowed for a straightforward, objective prioritization process using cosine similarity between each requirement and the cluster centroid. Hierarchical clustering methods, including Agglomerative Clustering, do not generate explicit centroids, making them unsuitable for this purpose.

4.3. Phase 3: Prioritization Within Clusters

In the final phase, prioritization within each cluster is achieved by evaluating the centroid similarity using the K-Means algorithm. This method leverages cosine similarity to rank the requirements based on their proximity to the cluster centroid, ensuring that the most representative requirements are prioritized at the top. If multiple requirements have the same similarity score, they are assigned the same priority level. This objective method significantly reduces human error and bias, aligning the prioritization with the inherent structure of the dataset.

After clustering the requirements, each cluster contains a group of related requirements. For instance, one cluster may focus on security-related requirements, another on design-related requirements, and another on process-related requirements. Within each cluster, the associated requirements are then prioritized based on their similarity to the cluster centroid. Requirements that are closer to the centroid are considered more central and, therefore, have a higher priority for implementation. Conversely, as requirements move farther from the cluster center, their priority decreases accordingly. This systematic approach ensures that the most representative and essential requirements within each category are addressed first, enhancing the overall efficiency and coherence of the prioritization process.

The prioritization is performed within each cluster. After clustering requirements based on their content, the requirements within each cluster are prioritized according to their similarity to the cluster centroid. The priority number assigned to each requirement depends on the number of requirements in the cluster and its similarity score to the centroid. The prioritization starts from 1 within each cluster and increases as the similarity to the centroid decreases. Additionally, the similarity score is rounded to three decimal places to ensure that requirements with the same rounded score share the same priority level. As mentioned earlier, the baseline paper presents requirements hierarchically in three levels: Project Objective, Requirements, and Specific Requirements. Each objective contains multiple requirements, and each requirement has several specific requirements. And Prioritization Process in the Baseline Paper is as follows:

1. Objectives are prioritized first.
2. Requirements within each objective are then ranked separately.
3. Finally, specific requirements within each requirement (if there are any) are prioritized independently.

An example of an objective, its associated requirements, and specific requirements, along with their priorities, is shown in the Table 5 below.

Table 5. An example of an objective, its associated requirements, and specific requirements

Rank	Project Objective	Rank	Requirement	Rank	Specific Requirement
1	Better user Experience	1	All in 1 card	1.5	Combine ID card and session card
				1.5	Combine library card
				3	Combine Bloomsbury fitness card
				4	The combined card should not have many features
		2	Easier to use	1	More accurate scanning
		3	Use the same access control for library entrance		

The framework is presented in Figure 1. The output from this framework consists of 10 well-defined clusters with prioritized requirements within each. These results will be compared against the baseline [7] in the subsequent analysis to validate the efficacy and relevance of the proposed method.

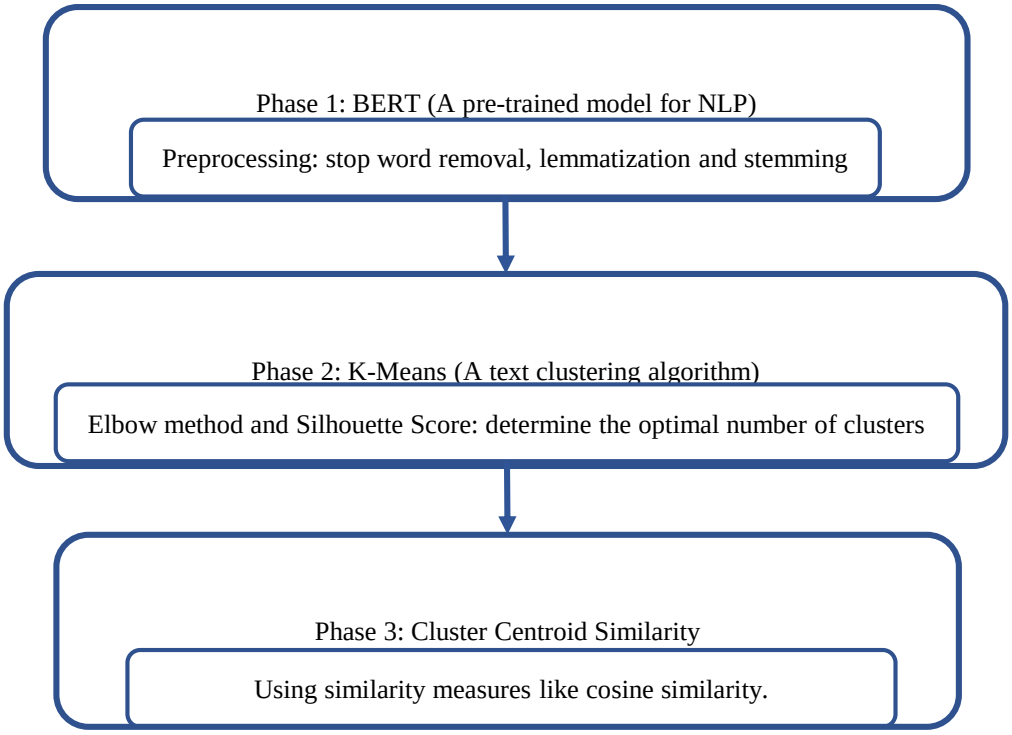

Figure 1. Research methodology.

5. Results

The RALIC dataset is a real-world dataset implemented as part of the RALIC software project in University College London. It is one of the few publicly available datasets containing requirements in natural language along with actual prioritization labels. This feature allows a direct comparison between the prioritization results of the proposed method and the existing real-world prioritization, ensuring the validity of the evaluation. Furthermore, the RALIC dataset is comprehensively documented in Lim's dissertation, where the ground truth prioritization results are provided [12]. This allows benchmarking the proposed framework against established prioritization outcomes.

In contrast, other available datasets mainly contain class labels rather than detailed prioritization information or categorize requirements into broad priority levels such as High (H), Medium (M), and Low (L), limiting the granularity of comparison. Additionally, if a new dataset were to be generated, it would lack predefined prioritization, making validation and comparison infeasible.

RALIC was a large-scale software project implemented at UCL to enhance and consolidate the existing access control system. The system replaced outdated access mechanisms such as swipe cards, contactless cards, photo ID cards, and digital security codes with a single smart card-based solution. RALIC integrated multiple functionalities into one system, allowing students, staff, and visitors to use an unified card for building access, library services, fitness center entry, and IT access. Benefits of RALIC for the Library:

- Improved efficiency: The system automated library access and borrowing processes, reducing manual data entry and processing time.
- Centralized management: RALIC streamlined user authentication for library access, ensuring that only authorized members could enter or borrow materials.
- Enhanced security: The system allowed the real-time monitoring of library access logs, reducing unauthorized use and security breaches.
- Convenience for users: Instead of managing multiple cards, students and staff had a single card for access to library resources, eliminating the need for barcode scanning at specific locations.

The implementation of the proposed framework was initiated with preprocessing approximately 400 raw natural language requirements. These requirements were subjected to a series of preprocessing steps aimed at refining the data for more advanced analysis. The preprocessing involved stop word removal, lemmatization and stemming, which are essential to reduce noise and standardize the language for processing.

Subsequently, these preprocessed inputs were fed into the BERT model. The BERT model's capability to understand the context and the intricate relationships between words in sentences transformed the requirements into semantically rich and contextually informed embeddings. This phase was critical as it significantly improved the accuracy and the integrity of the data for the clustering process that followed.

In the clustering phase, the K-Means algorithm was applied to these enriched embeddings to group similar requirements. The optimal number of clusters was determined through a meticulous application of the Elbow Method and the Silhouette Score, with both methods converging on 10 as the ideal number of clusters. This decision was substantiated by iteratively testing various initial cluster counts, with 40 being the upper limit tested. The iterative approach ensured that the chosen number of clusters maximized the meaningfulness and manageability of the data sets.

Initially, the Elbow method was used to determine the optimal number of clusters. Various values were tested as input parameters. However, the Elbow Method did not provide a precise cluster count, and multiple possible values could be interpreted as the optimal number based on the Figure 2. For this reason, the Silhouette Score Method was also applied to select the best cluster count from the range suggested by the Elbow Method. The Silhouette Score helped refine the selection process, ultimately indicating that 10 clusters was the most suitable choice.

Following clustering, the prioritization within each cluster was based on the proximity of the requirements to the cluster centroid, which effectively reflected the core themes of the grouped data. The centroid-based prioritization ensures that the most representative requirements of each cluster were identified and ranked accordingly. This step was particularly significant as it minimized human bias, enhanced objectivity, and streamlined the decision-making process.

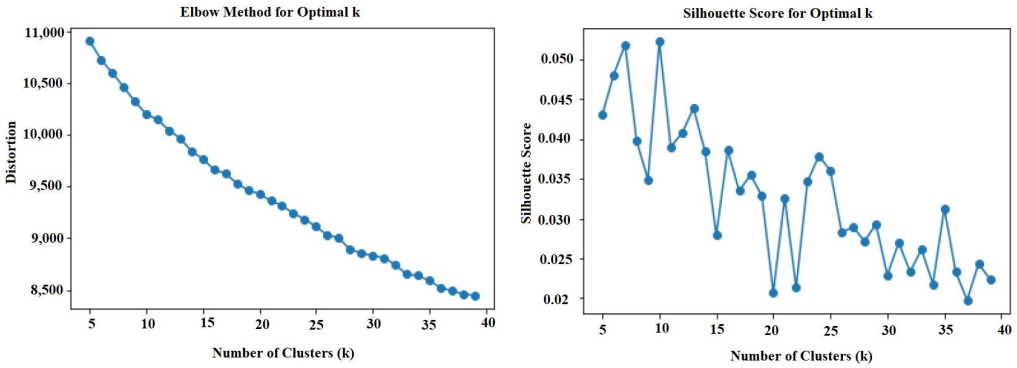

Figure 2. Algorithm for determining the optimal number of clusters using the elbow and silhouette score methods.

To validate the effectiveness of the proposed framework, a rigorous comparison was conducted with the baseline paper's prioritization. This involved a detailed manual matching process, where each requirement processed by the framework was mapped to the corresponding entries in the baseline paper [7]. This task was both time-consuming and prone to human error but was essential for ensuring accurate comparison.

The requirements in the dataset are raw requirements, with each requirement appearing multiple times in different phrasing and writing styles. However, the requirements actually used in the project and prioritized are structured differently. They are categorized and prioritized into three levels, with each requirement being summarized in a specific format and assigned a unique ID. In this study, instead of using the structured and categorized requirements, we worked with the raw, repetitive, and imbalanced requirements from the dataset. To compare the prioritization results with the baseline paper, it was necessary to map each raw requirement in the dataset to a structured requirement in the project. This mapping process was performed based on the semantic meaning of the requirements, rather than using any specific software or automated tool, as it required a conceptual understanding of each requirement.

The dataset's stakeholders, each with distinct roles, had provided varying ratings for the requirements. Where requirements received multiple ratings, these were averaged to compute a weight for each, which facilitated an organized prioritization within each cluster. Sorting these weighted requirements from highest to lowest provided a clear view of priority levels as perceived by the stakeholders.

The comparison of our automated prioritization results with those obtained using a traditional Multi-Criteria Decision Making (MCDM) method utilized the Mean Absolute Error (MAE) as the metric. This comparison highlighted the precision and alignment of our automated method with established prioritization methods.

For comparison with the base paper, the requirements within each cluster were prioritized using a method derived from the base approach. In the RALIC dataset, each stakeholder role is assigned a rank. For example, the role of Security and Access Systems has a rank of 1. Additionally, within each role, stakeholders are ranked. For instance, Mike Dawson holds rank 1, while Jason Ortiz holds rank 2 within this role.

Using these ranks, the total number of roles, the total number of stakeholders, and the stakeholders within each role, a formula was applied to compute the rank of each stakeholder. For example, the computed rank for Mike Dawson is 0.0238. Furthermore, in the RALIC dataset, stakeholders have assigned ratings to various requirements. For instance, Mike Dawson assigned a score of 4 (on a scale of -1 to 5) to requirement 30. The weight of this requirement is then calculated as 0.0952. If the same requirement was rated by another stakeholder, the same process was applied, and the resulting weight was

summed with the previously obtained value. This cumulative weight represents the final weight of the requirement.

Finally, the requirements were ranked based on their weights (Table 6), with higher-weighted requirements receiving a higher priority, while lower-weighted ones were assigned lower priority. Once the prioritization was performed using the approach adapted from the base paper, the resulting priorities were compared with those obtained through the proposed method.

Table 6. Mean absolute error values across clusters.

Cluster	MAE Value	Comments
1	1.00	A moderate level of accuracy
2	1.02	A moderate level of accuracy
3	1.33	Acceptable deviation
4	1.04	A moderate level of accuracy
5	1.01	A moderate level of accuracy
6	1.05	A moderate level of accuracy
7	0.81	Better-than-average
8	0.95	Fairly alignment
9	0.65	Excellent alignment
10	0.98	Fairly alignment

The MAE of approximately 1 across all clusters suggests that, on average, the predicted priority differs from the baseline paper by one rank position. While this may not indicate perfect accuracy, it does demonstrate that the framework consistently produces rankings close to stakeholder-based prioritization, especially considering the inherent subjectivity in manual prioritization. In comparison with existing automated methods, achieving an MAE of around 1 suggests that the proposed method performs within an acceptable error margin for prioritization tasks. These results validate the capability of the proposed framework to not only automate the prioritization process but also to achieve a high level of accuracy and reliability in outcomes.

To assess the effectiveness of the proposed automated requirement prioritization framework, we also adopt commonly used evaluation metrics, including Precision, Recall, and F1-score. These measures provide a comprehensive evaluation of how well the framework prioritizes requirements compared to the computed base paper prioritization.

The F1-score, which provides a balance between precision and recall, was around 0.6 for different categories. Although this value does not indicate an excellent comparison, it suggests that the results of the framework are acceptable.

To enhance comprehension of the proposed framework, 20 requirements were selected from one of the existing clusters. The similarity scores obtained from the clustering process were determined for these requirements. Additionally, the weights of these requirements, calculated using the baseline paper's proposed method, were included in the table under the title "Base Weight." Using both the baseline paper's weight and the scoring from our proposed method, these 20 requirements were prioritized. For this sample of the overall proposed framework, the MAE is 0.9, which is an acceptable value.

Table 7 presents the results of comparing the prioritization of 20 requirements, while Table 8 shows the 5 requirements after completing the preprocessing phase.

Table 7. Prioritization comparison of 20 requirements.

No.	Raw Requirement	Base Weight	Similarity Score	Base Priority	Proposed Priority
1	While providing security enable staff and students to access building and facilities easily	0.0677	0.886	1	1
2	The cards and readers should permit reuse for tracking presence	0.0655	0.869	2	2
3	Allow guards to see pictures of people passing the gate	0.0654	0.866	3	3
4	To enable smooth/efficient building access control whether via visual inspection	0.0654	0.845	3	5
5	Access to buildings and other services should be capable of being extended	0.0641	0.857	4	4
6	Allow access to buildings for different staff	0.0641	0.857	4	4
7	To allow checking of the roles a person has in their department	0.0641	0.838	4	6
8	Monitor alumni use, eg library use, shop, fitness etc.	0.0638	0.857	5	4
9	Enable additional access control to buildings	0.0625	0.838	6	6
10	To place card access on all buildings	0.0625	0.838	6	6
11	Readers can be easily installed in new locations	0.0625	0.780	6	8
12	Access for users of Bloomsbury Fitness	0.0619	0.838	7	6
13	To access university buildings & borrow library books	0.0616	0.831	8	7
14	To be able to gain access to different buildings, so correct level access given	0.0616	0.761	8	9
15	To be able to control the access to all buildings at UCL	0.0616	0.745	8	10
16	Enter libraries with appropriate access	0.0615	0.831	9	7
17	To allow entry to the Kathleen Lonsdale building where the user services office resides	0.0615	0.780	9	8
18	To control access to high hazard areas both in the main campus and in the satellite and "shared" facilities	0.0615	0.761	9	9
19	To be able to find out who has gained entry to cluster rooms / IT rooms	0.0615	0.745	9	10
20	To allow/deny access to rooms	0.0615	0.625	9	11

Table 8. Preprocessed requirements.

Raw Requirements	Pre-Processed Requirements
While providing security, enable staff and students to access building and facilities easily	provide secure enable staff student access build facil easili
The cards and readers should permit reuse for tracking presence	card reader permit reus track presens
Allow guards to see pictures of people passing the gate	allow guard see pictur peopl pass gate
To enable smooth/efficient building access control whether via visual inspection	enable build access control whether via visual inspect
Access to buildings and other services should be capable of being extended	access build servic capable extend

6. Discussion

The implementation of a fully automated framework for prioritizing raw natural language requirements represents a significant advancement in the field of requirements engineering. This section discusses the various facets of the framework, including its rationale, benefits, and potential challenges.

The core novelty of our approach is how it processes requirements compared to existing methods. Unlike previous studies, which relied on numerical scores assigned by stakeholders, our method directly works with raw natural language requirements—clustering and then prioritizing them without requiring predefined numerical inputs. This is a fundamental shift from traditional approaches, where prioritization is performed on numerical values given by stakeholders rather than on the actual textual requirements.

This distinction makes our approach more autonomous, scalable, and adaptable to real-world scenarios where stakeholders might not always provide structured numerical inputs. Our method ensures the following:

- No manual numerical weighting is needed before prioritization—the framework understands and processes textual requirements directly.
- Uses contextual embeddings from BERT, making it more effective than TF-IDF or rule-based NLP approaches.
- Prioritizes requirements within each cluster based on centroid similarity, ensuring that the most representative requirements are ranked higher.
- Provides a scalable and domain-independent solution that can generalize to different projects beyond RALIC.

This novel integration of unsupervised learning (K-Means clustering) and ranking via centroid similarity represents a major advancement over semi-automated approaches like StakeRare and SRPTackle, where stakeholder input plays a dominant role. By shifting from stakeholder-driven numerical scoring to fully automated natural language processing, our framework eliminates bias, inconsistency, and dependency on subjective human input.

6.1. Rationale for Automation

The primary motivation behind developing a fully automated framework stems from the need to overcome the inherent biases and inefficiencies associated with traditional stakeholder-based prioritization methods. In conventional settings, stakeholders' conflicting opinions and personal or departmental interests can significantly skew the prioritization process. Furthermore, the reliance on stakeholders' input is fraught with challenges such as

uneven distribution of attention across all requirements, with some being overly prioritized and others neglected. This often leads to an imbalance in requirement handling and can adversely affect the project outcome.

6.2. Advantages of the Automated Framework

The proposed framework addresses these issues by eliminating human intervention in the prioritization process, thus enhancing objectivity and fairness. Using advanced natural language processing techniques and machine learning algorithms such as BERT and K-Means clustering, the framework ensures that requirements are analyzed and prioritized based on their semantic content rather than subjective interpretations. This method not only streamlines the process, making it more efficient, but also reduces the potential for human error.

- **Consistency:** Automation ensures that the same criteria are applied uniformly to all requirements, promoting consistency in the prioritization outcomes.
- **Scalability:** The ability to process large volumes of requirements without incremental costs or delays is another significant advantage, making the framework highly scalable and adaptable to various project sizes.
- **Speed:** Automated systems can process requirements much faster than human teams, enabling quicker transitions from the planning phase to implementation.

6.3. Challenges and Limitations

Despite its numerous benefits, the automated framework is not without challenges. One of the main limitations is the initial setup and tuning of the algorithms, which requires significant expertise and understanding of both the tools and the project context. Additionally, while the framework reduces the influence of human biases, it also removes the nuanced understanding that experienced stakeholders bring to the table, particularly in complex projects where contextual insights are crucial.

- **Complexity in Setup:** Configuring the BERT and K-Means algorithms to accurately interpret and cluster complex requirement statements can be challenging and time-intensive.
- **Loss of Human Insight:** The absence of stakeholder input might lead to the oversight of context-specific priorities that are not apparent through textual analysis alone.
- **Adaptability Issues:** The framework's dependence on predefined models and algorithms might limit its adaptability to highly dynamic or unconventional project requirements.

The primary objective of this study is to automate the prioritization process, reducing reliance on subjective stakeholder input. This approach does not claim to be the most comprehensive solution but rather a step towards minimizing human biases while ensuring scalability and efficiency.

Since the prioritization is based on the inherent structure of the data, rather than individual preferences, it reflects an unbiased yet informed ranking of requirements. This method can be further adapted by incorporating expert-driven validation at the final stage to refine the prioritization results where necessary, allowing a balance between automation and expert judgment.

Future work will focus on incorporating stakeholder perspectives, which is itself a complex challenge. Integrating domain expertise with automation in a structured and reliable manner remains an open research problem, and a dedicated study is planned to explore effective ways of combining stakeholder opinions with this automated approach.

- Handling Evolving Requirements
  Since requirements can change over time, our framework's automated and iterative nature allows it to be reapplied periodically to accommodate new or modified requirements. The BERT-based embeddings ensure that semantic similarities remain valid, even if new requirements are introduced. Potential Enhancement: A mechanism for incremental updates could be explored in future work, where new requirements are dynamically incorporated without rerunning the entire pipeline.
- Managing Conflicting Priorities
  Our clustering approach groups similar requirements together, reducing the risk of conflicting requirements being prioritized at opposite ends of the ranking. However, if conflicts arise, a hybrid approach could be integrated in future work, combining automated prioritization with stakeholder opinions for the final stages of the proposed approach.
- Adapting to Dynamic Project Constraints
  Project constraints, such as budget, time, and resource limitations, can impact requirement prioritization. Our method can be extended by introducing constraint-aware ranking: For example, using a cost–benefit analysis to ensure that high-priority requirements are also feasible within the given constraints. If a requirement is ranked high but exceeds constraints, a constraint-adjusted prioritization step could be introduced.

6.4. Ethical and Practical Challenges in Requirement Prioritization Automation

Automating requirement prioritization presents several ethical and practical challenges that must be carefully addressed to ensure fairness, transparency, and usability in real-world applications.

1. While automation improves objectivity and efficiency, it removes valuable domain knowledge and contextual insights from stakeholders. This can lead to misinterpretation of nuanced requirements, especially in complex projects where expert judgment is essential.
2. Automated methods, including clustering and similarity-based ranking, rely on the data they are trained on. If the dataset is imbalanced or contains hidden biases, the prioritization results may reflect and even reinforce those biases, potentially leading to unfair or suboptimal decisions.
3. Many automated approaches, particularly those based on machine learning models like BERT, operate as "black boxes". Ensuring that prioritization decisions are interpretable and justifiable to stakeholders is crucial for trust and adoption.
4. Removing human decision-makers from the prioritization process may raise ethical concerns, especially in domains like healthcare, public policy, or critical infrastructure, where stakeholder values and ethical considerations play a significant role.
5. Automated approaches need to be adaptable to evolving requirements, changing business needs, and dynamic stakeholder expectations. A rigid automation framework might not be suitable for projects where flexibility is essential.

6.5. Proposed Framework's Practical Applicability

Our automated requirement prioritization framework can be applied to large-scale software projects where manual prioritization is inefficient or stakeholder involvement is complex. Here are two real-world examples where our framework could be beneficial:

1. **Smart City Infrastructure Management Systems**
   - Scenario:

  - A city government is developing an integrated smart city management system, which includes traffic monitoring, public transportation, security surveillance, and waste management.
- Challenges:
  - Hundreds of requirements from multiple departments (e.g., transport, security, public works, IT).
  - Conflicting priorities between stakeholders (e.g., security teams prioritize surveillance, but transport teams prioritize real-time traffic data).
  - Manual prioritization is slow and prone to biases.
- How our framework helps:
  - Clusters and prioritizes requirements automatically, reducing dependence on stakeholder input.
  - Ensures objective ranking based on inherent requirement characteristics rather than subjective opinions.
  - Enhances scalability, making it easier to manage large, evolving requirement sets.

2. **Large-Scale Healthcare Management Systems**
   - Scenario
     - A national healthcare system is upgrading its electronic medical records (EMR) system to integrate patient data, telemedicine services, and AI-based diagnostics.
   - Challenges:
     - Thousands of requirements across multiple domains (doctors, hospitals, insurance providers, IT teams, regulatory agencies).
     - High complexity due to legal and security constraints (e.g., patient data privacy).
     - Traditional prioritization methods struggle to account for dynamic changes in medical technology and regulations.
   - How our framework helps:
     - Uses BERT-based clustering to group requirements by theme (e.g., security, usability, compliance).
     - Ensures high-priority healthcare requirements (e.g., patient data protection, emergency response) are not overshadowed by less critical ones.
     - Reduces human effort in requirement analysis and decision making, enabling faster project execution.

These examples demonstrate its real-world applicability in government, enterprise, and healthcare sectors, ensuring efficient and objective requirement prioritization at scale.

6.6. Future Directions

To address these limitations, future work could explore hybrid models that integrate limited stakeholder input at critical junctures of the prioritization process. This could help balance the objectivity of the automated system with the contextual acuity of human oversight. Further research could also focus on developing more sophisticated NLP tools that can better capture and interpret the subtleties and complexities of natural language used in requirement documents.

7. Conclusions

This paper presented an automated framework designed to prioritize requirements with minimal stakeholder input, addressing the challenges commonly associated with subjective prioritization methods. Initially, a dataset comprising natural language text was meticulously preprocessed and then analyzed using the BERT model to ensure a deep semantic understanding of the requirements. Following this, the preprocessed data were organized into clusters via the k-means algorithm to group similar requirements effectively. Subsequently, the requirements within each cluster were prioritized using centroid similarity.

To validate the effectiveness of the proposed framework, the priority of each requirement was rigorously compared to its established ground truth using the mean absolute error (MAE).Each requirement was assigned a weight based on a Weighted Average method, which considered the rank and input from stakeholders, albeit minimally. These weighted requirements were then methodically sorted and prioritized, ensuring that the most critical requirements were identified and highlighted. The results from this comparison underscored the framework's capability to prioritize requirements accurately and objectively, demonstrating substantial alignment with baseline paper.

In summary, the proposed automated framework significantly enhances the efficiency and objectivity of the requirement prioritization process. However, it is crucial to acknowledge its limitations, particularly concerning the exclusion of nuanced stakeholder insights in complex projects. Ongoing research and development are essential to refine and adapt the framework, ensuring it can effectively respond to the diverse and evolving needs of different projects. With continued advancements, this framework holds considerable promise for widespread adoption in the field of requirements engineering, potentially setting a new standard for how requirements are prioritized in large-scale projects.

Author Contributions: Conceptualization, B.J. and S.R.K.; methodology, B.J.; software, B.J.; validation, S.R.K., R.A., and R.T.; writing—original draft preparation, B.J.; writing—review and editing, R.A.; supervision, S.R.K., R.A., and R.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: All authors declare that they have no conflict of interest.

References

1. Umar, M.A.; Lano, K. Advances in automated support for requirements engineering: A systematic literature review. *Requir. Eng.* **2024**, *29*, 177–207.
2. Khan, M.A.; Azim, A.; Liscano, R.; Smith, K.; Chang, Y.K.; Tauseef, Q.; Seferi, G. Machine Learning-based Test Case Prioritization using Hyperparameter Optimization. In Proceedings of the 5th ACM/IEEE International Conference on Automation of Software Test (AST 2024), Lisbon, Portugal, 15–16 April 2024; pp. 125–135.
3. Ajiga, D.; Okeleke, P.A.; Folorunsho, S.O.; Ezeigweneme, C. Navigating ethical considerations in software development and deployment in technological giants. *Int. J. Eng. Res. Update* **2024**, *7*, 50–63.
4. Vestola, M. A comparison of nine basic techniques for requirements prioritization. *Hels. Univ. Technol.* **2010**, 1–8.
5. Hujainah, F.; Bakar, R.B.A.; Nasser, A.B.; Al-haimi, B.; Zamli, K.Z. SRPTackle: A semi-automated requirements prioritisation technique for scalable requirements of software system projects. *Inf. Softw. Technol.* **2021**, *131*, 106501. [CrossRef]
6. Hujainah, F.; Bakar, R.B.A.; Abdulgabber, M.A.; Zamli, K.Z. Software requirements prioritisation: A systematic literature review on significance, stakeholders, techniques and challenges. *IEEE Access* **2018**, *6*, 71497–71523. [CrossRef]
7. Lim, S.L.; Finkelstein, A. StakeRare: Using social networks and collaborative filtering for large-scale requirements elicitation. *IEEE Trans. Softw. Eng.* **2011**, *38*, 707–735.

8. Reyad, O.; Dukhan, W.H.; Marghny, M.; Zanaty, E.A. Genetic k-means adaption algorithm for clustering stakeholders in system requirements. In *Advanced Machine Learning Technologies and Applications, Proceedings of the AMLTA 2021, Cairo, Egypt, 22–24 March 2021*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 195–204.
9. Tzimos, D.; C. Gerogiannis, V.; Son, L.H.; Karageorgos, A. A Recommender System based on Intuitionistic Fuzzy Sets for Software Requirements Prioritization. In Proceedings of the 25th Pan-Hellenic Conference on Informatics, Volos, Greece, 26–28 November 2021; pp. 466–471.
10. Limaylla, M.I.; Condori-Fernandez, N.; Luaces, M.R. Towards a semi-automated data-driven requirements prioritization approach for reducing stakeholder participation in SPL development. *Eng. Proc.* **2021**, *7*, 27. [CrossRef]
11. Achimugu, P.; Selamat, A.; Ibrahim, R. A clustering based technique for large scale prioritization during requirements elicitation. In *Recent Advances on Soft Computing and Data Mining, Proceedings of the First International Conference on Soft Computing and Data Mining (SCDM-2014) Universiti Tun Hussein Onn Malaysia, Johor, Malaysia, 16–18 June 2014*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 623–632.
12. Lim, S.L. Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation. Ph.D. Thesis, UNSW Sydney, Kensington, Australia, 2010.
13. Hu, W.; Xu, D.; Niu, Z. Improved k-means text clustering algorithm based on BERT and density peak. In Proceedings of the 2021 2nd Information Communication Technologies Conference (ICTC), Nanjing, China, 7–9 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 260–264.
14. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
15. Bird, S.; Klein, E.; Loper, E. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2009.
16. Kenton, J.D.M.W.C.; Toutanova, L.K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, p. 2.
17. Ilyas, F.M.; Priscila, S.S. An Optimized Clustering Quality Analysis in K-Means Cluster Using Silhouette Scores. In *Explainable AI Applications for Human Behavior Analysis*; IGI Global: Hershey, PA, USA, 2024; pp. 49–63.
18. Darji, K.; Patel, D.; Vakharia, V.; Panchal, J.; Dubey, A.K.; Gupta, P.; Singh, R.P. Watershed prioritization and decision-making based on weighted sum analysis, feature ranking, and machine learning techniques. *Arab. J. Geosci.* **2023**, *16*, 71. [CrossRef]
19. Shambour, Q.Y.; Abu-Alhaj, M.M.; Al-Tahrawi, M.M. A hybrid collaborative filtering recommendation algorithm for requirements elicitation. *Int. J. Comput. Appl. Technol.* **2020**, *63*, 135–146. [CrossRef]