Received: 20 March 2022 | Revised: 29 July 2022 | Accepted: 19 August 2022

DOI: 10.1049/cmu2.12555

ORIGINAL RESEARCH

A multi-agent K-means with case-based reasoning for an automated quality assessment of software requirement specification

Mohammed Ahmed Jubair[1] | **Salama A. Mostafa**[2] | **Aida Mustapha**[3] | **Mohamad Aizi Salamat**[2] | **Mustafa Hamid Hassan**[1] | **Mazin Abed Mohammed**[4] | **Fahad Taha AL-Dhief**[5]

[1]Department of Computer Technical engineering, College of Information Technology, Imam Ja'afar Al-Sadiq University, Al-Muthanna, Iraq

[2]Faculty of Computer Science and Information Technology, Universiti Tun Hussin Onn Malaysia, Johor, Malaysia

[3]Faculty of Applied Sciences and Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

[4]College of Computer Science and Information Technology, University of Anbar, Ramadi, Iraq

[5]Faculty of Engineering, School of Electrical Engineering, Universiti Teknologi Malaysia (UTM), Johor Bahru, Malaysia

Correspondence
Mohammed Ahmed Jubair, Department of Computer Technical engineering, College of Information Technology, Imam Ja'afar Al-Sadiq University, Al-Muthanna 66002, Iraq
Email: mohammed.a@sadiq.edu.iq

Funding information
the Fundamental Research Grant Scheme, Grant/Award Number: FRGS/1/2019/ICT04/UTHM/03/1

Abstract
Automating the quality assessment of Software Requirement Specification poses major challenges related to the need for advanced algorithms to extract the SRS quality features, interpret the context of the features, formulate accurate assessment metrics, and document the shortcomings as well as possible improvements. In the existing methods, such as Reconstructed Automated Requirement Measurement, and Rendex, some major processes are still handled offline by humans (semi-automated) or encompass automating the measurement of a few quality attributes due to the mentioned challenges. This paper addressed this gap and proposed an Automated Quality Assessment of SRS (AQA-SRS) framework to assess the SRS documents by automatically extracting features related to 11 quality attributes through a deep analysis of the SRS textual content. Also, it constructs a flexible platform that is able to minimize the human expert's role in the SRS assessment. The AQA-SRS framework integrates Natural Language Processing, K-means, Multi-agent, and Case-Based Reasoning. The AQA-SRS framework is evaluated by processing two standard SRS datasets and comparing the results with state-of-the-art methods and analysis by software engineering experts. The results show that the AQA-SRS framework effectively assesses the tested SRS documents and achieves a 78% total agreement with the tested methods and software engineering experts.

1 | INTRODUCTION

Requirement Engineering (RE) is one of the early processes in software development, where its role is to compile functional and non-functional requirements in the Software Requirement Specification (SRS) document. The SRS is a written document by software analysts in a natural language for the stakeholders and software development team [1]. The SRS defines the software requirements, spells out the limits of the software system, and highlights the surrounding system [2]. The requirement describes the software product's expected functionalities, capabilities, features, and services. It can be represented as an objective that a system must meet, and the specification is a description of how the objective must be met [3–5]. Requirements are written in natural language so as to be understood by all stakeholders without supplementary effort and specific requirements engineering background. One main challenge in preparing SRS documents is the complexity of writing requirements and structuring the document to describe best the software product [3].

The SRS is a step-by-step guideline for producing a software product, and poorly written requirements of the SRS can cause defects in the future software product [6]. Incomplete or ambiguous requirements generate additional effort due to unnecessary feedback loops to interpret the exact meaning of this requirement [7]. Software Quality Assurance (SQA) is a set of processes that utilize to ensure the quality of any delivered software by monitoring the activities of the software engineering life cycle. SQA aims to ultimately lead to or at least give confidence in producing high-quality software products [8]. The main focus of the SQA in the requirement stage is to ensure the quality of the SRS in the early starting steps of the software development process [9]. It sets explicit rules associated with the production of high-quality SRS documents. The Institute of Electrical and Electronics Engineers (IEEE) has recommended good practice of writing SRS documents.

Wilson et al. [1] propose a set of quality attributes (Q^a) including complete, consistent, correct, modifiable, ranked, testable, traceable, unambiguous, understandable, validatable, and verifiable and quality indicators (Q^i) including Imperative, continuances, directives, options, weak phrases, size, text structure, specification depth, and readability that used to assess the quality of the SRS document. These $Q^a s$ and $Q^i s$ are used to ensure that the SRS has high-quality requirements. Subsequently, the researchers in the field have proposed several methods to detect the defect in the SRS document based on the Q^a and Q^i, such as the work of Jani and Mostafa [3], Jani and Islam [4], Alshazly et al. [10],Haque et al. [11], and Femmer et al. [12].

The SRS document has an expensive impact on the software project, in which the success or failure of any software product highly depends on the quality of its SRS document. SRS assessment or inspection methods can be categorized into three main categories: automated, semiautomated, and non-automated or manual inspection. Manual inspection is considered a popular technique used to assess the quality of the SRS document. However, it is a time-consuming method that entails different reviewers inspecting the SRS and manually integrating the review results in one report [13]. Since the completion of a review cycle is achieved within several days or even weeks, the author of the requirements must wait a long time before receiving feedback [14]. Additionally, the reviews of different individuals' consequent inconsistency [15].

The implication of these problems affects the assessment quality of the SRS and increases the cost. Subsequently, several advancement attempts in the RE field have started to explore the automated extraction of quality attributes (complete, correct, consistent, etc.) from SRS documents to assess the quality of the SRS documents. Different Artificial Intelligence (AI) and statistical methods have been used to address solutions to these issues. Examples are case-based reasoning (CBR) as in [3], Fuzzy-CBR as in [16], natural language process (NLP) as in [17, 18], correlation coefficient as in [14], k-means as in, and multi-agent system (MAS) as in [19].

For instance, Jani and Islam [4] propose a semi-automated method based on CBR that combines manual inspection and automated assessment methods to reduce human assessment time, cost, and workload. However, there is still a need for improvement because humans (experts) still interact with the method to assess the quality of the document. Additionally, there is no standard Q^a and Q^i that can assess the quality of the SRS in an automated manner. The previous work attempted to assess the quality of the SRS based on a limited number of Q^a and Q^i. For instance, the work of Siahaan and Umami [20] introduces a method to detect the forward reference in the SRS document by using NLP. Carlson and Laplante [21] reconstructed the Automated Requirement Measurement (reconstructed ARM) method to assess the quality of the SRS based on three Q^as, which are ambiguity, complete, and understandability. Antinyan and Staron [14] introduced the Rendex framework to assess the understandability of the document. This leads to the need for a comprehensive framework that deals with different methods or techniques like; pre-processing, features extraction, analysis, and assessment to handle a wider range of Q^a 22]–[24.

Here, an Automated Quality Assessment of SRS (AQA-SRS) framework has been proposed that includes several statistical and AI methods to produce a highly flexible platform that is able to minimize the human expert's role in the SRS assessment. For this reason, this paper focuses on four main points. Firstly, define a new group of Q^a and Q^i that can be assessed in an automated way. Secondly, construct a framework that is able to assess the quality of the SRS based on the Q^a and Q^i and in an automated manner. Thirdly, integrates K-means clustering algorithms in collaboration with a MAS to cluster the extracted feature to their corresponding Q^a and Q^i. Finally, employing a CBR to manage the entire assessment process of SRS assessment based on the experience of previous SRS assessments and generate the assessment report. The contributions of this paper can be classified into five main points that are presented as follows:

- We propose an Automated Quality Assessment of SRS (AQA-SRS) framework with diverse methods and algorithms to assess the quality of the SRS documents. The framework encompasses CBR as a core component of the AQA-SRS framework for managing the entire assessment process, NLP for feature extraction, K-means for features clustering, MAS for interactive assessment, and feature selection decisions. This combination of methods and algorithms and their deployment in one framework is new.
- We define a new group of Q^a and Q^i that can be assessed in an automated way. To the best of our knowledge, this study is the first attempt to assess eleven Q^a at the same time. As mentioned earlier, the automated methods, such as the work of Génova et al. [25], Thitisathienkul and Prompoon [9], and Bakar et al. [24], only consider a limited number of Q^a such as ambiguity, completeness, and correctness.
- We propose a new MAS architecture that consists of K-means for features clustering, agents for interactive assessment, and feature selection decision. The main role of the

agents is to improve the clustering process by monitoring the boundaries of each cluster based on the $Q^a$and Q^i.

- We constructed a case base to retain benchmark SRSs that human experts assess. The CBR uses the case base to calculate the similarity percentages between the benchmark cases and new cases to verify the new cases' quality.
- We implement, test, and evaluate the AQA-SRS method using two standard datasets: PURE and reconstructed ARM and a self-collected dataset. The test results show that the AQA-SRS method is able to assess the quality of the SRS documents at low cost, time-efficient, low workload, and high consistency.

This section introduces the research scope and problem and presents the achieved contributions. The rest of the paper is organized into four sections as follows: Section 2 presents the related work. Section 3 describes the methods and materials that are required for conducting this work. Section 4 illustrates the AQA-SRS framework. Section 5 represents the implementation and results. Finally, Section 6 presents the conclusions and future work.

2 | RELATED WORK

Automated requirements analysis converges various research fields, including text processing, quality analysis, and d mining. This section represents the efforts that have been targeted toward improving the quality of the SRS document. Since most software requirements today are still written in natural language, these approaches focus on measurable automated indicators that can be derived from the text. The review covers a period between 2012–2019 based on two perspectives; SRS quality assessment and features classification process, as discussed in the following sections.

Text processing techniques were used in Carlson and Laplante [21]. Their work proposed an automated method known as the reconstruction Automated Requirement Measurement (reconstruction ARM) tool to assess the SRS document's quality. The traditional ARM tool proposed by NASA [1] did not consider the requirements embedded in tables. The text processing techniques in Reconstruction ARM assessed the quality of the SRS document based on several steps. First, the input document is read and flattened, the input text is reduced to a sequence of lines of text, whitespaces are eliminated, and lines are consolidated where possible. Second, each line is parsed for paragraph numbers to be used as both position indicators within the SRS and to determine the document depth.

Nordin et al. [2] focused on text processing techniques at the syntactic level in order to identify automated indicators that measure the quality of SRS documents. Their work developed a tool named SRS Quality Checker as a proof-of-concept of the rules. The SRS Quality Checker measures the quality of the SRS document based on Requirements Sentence Quality (RSQ) and Requirements Document Quality (RDQ). The RSQ is a syntactical quality of single sentences considered separately, whereas RDQ is the quality of the sentences considered in the context of the whole requirements documents. Seven quality indicators were used: implicit, optional, vague, weak, multiple, directive, and readability. These indicators were employed to identify each quality attribute determined beforehand: consistency, completeness, and understandability. Similar to previous works, a Part-of-Speech (PoS) tagger was used to assign each word in the requirement sentence and find the multiple verbs in the requirements.

Husain and Khanum [27] proposed a framework named Word Sense Disambiguation (WSD) for tackling and removing the ambiguity problem in the SRS document. WSD is part of the Natural Language Processing (NLP) technique that deals with words that have several interpretations in context. In their work, WordNet was used to address the issue of multiple interpretations, where the English Wordnet is used to discover the polysemy words in the document. The WSD framework consists of four main steps were used to produce a clear SRS document; (1) reading the input SRS document, (2) performing PoS tagging to tag each word in the document to shorten the list of possible synonym of the ambiguous word, (3) matching the ambiguous words against WordNet, and (4) employing n-gram and association rule mining to find the frequent itemsets and the relationships between apparently unrelated data in a relational database or another information repository.

Ali et al. [6] proposed an approach to improve the quality of SRS documents focusing on ambiguity and correctness. In the presented approach, utilize four steps. In the first step, one of the NLP techniques is used to parse the document to confirm the completeness of the requirement. While in the processing step, the mapping requirements are used to assess the overall perspectives of the different stakeholders. After certifying the final condition of requirements mapping, the external inspection is conducted by an expert to evaluate the overall quality attributes of SRS. Finally, the final report contains all details about the SRS document. The average Total Quality Score (TQS) after applying 3rd party inspection is 71.6%, which shows that the proposed approach can improve the quality better than the simple inspection. This process improves the SRS document's overall quality, which positively influences product development.

Antinyan and Staron [14] presented Rendex, an automated method to evaluate the understandability of the SRS document. Rendex assessed the understandability by introducing a new group of quality attributes and indicators. The quality attributes are; complexity, coupling, and size. Each quality attribute is relative to a specific quality indicator and group of features. The complexity depends on the number of conjunctions and a number of vague words. Meanwhile, the coupling attribute relies on the number of references and the number of references document. The size is affected by the number of words in the document. Rendex implemented four main steps to assess the quality of the SRS document. Primarily, extract the requirement from the document and memorize it in a structure file, calculate the quality of the requirements, sort the requirement in descending order, and finally, generate the assessment report.

Mezghani et al. [28] proposed an automated approach to detect the inconsistency and redundancy in requirements engineering through an unsupervised clustering algorithm, K-means. Text processing techniques were also employed to implement PoS tagging and noun chunking. The data mining algorithm used, K-means, was to partition the requirement based on the number of K, which was calculated based on the statistical gap method. The proposed approach is evaluated according to the optimal value of k generated via implementing the statistic gap method. Because data mining approaches are data-driven, three corpus were used to validate their proposed approach. Corpus 1 comprises 38 fully redundant requirements, Corpus 2 contains 42 fully inconsistent requirements, and Corpus 3 includes 337 redundancy and inconsistency requirements randomly selected from the datasets. The K-means provided truly relevant outcomes by providing only clusters with related information. The results of Corpus 3 illustrated the importance of the pre-processing step to enhance the clustering results in terms of accuracy and the number of detected clusters.

Manek et al. [29] also proposed a clustering approach to cluster the given requirements by using the Spectral Clustering (SC) algorithm. SC classified the given requirement into a group of requirements that describe a related function in the same cluster. K-means clustering algorithm was again used to calculate the Euclidian distance between clusters. Finally, the noisy statements were selected based on the largest distance between clusters. The performance of the proposed method is evaluated by computing the kappa coefficient value, which assesses the agreement level between noise prediction via the proposed method and noise assessment via three human experts. The result of the kappa coefficient is 0.4426, which is still very low.

The review of the related work shows that efforts have been directed toward proposing several frameworks to evaluate the SRS document in an automated way. For this purpose, different $Q^a$are were used, especially the ambiguity, completeness, and understandability. The existing SRS quality assessment metrics combine automated, semi, and non-automated attributes. However, there is no metric of fully automated Q^a and Q^i. Further, the literature lacks a fully automated framework that can assess all Q^as for the SRS document. Due to overlapping between Q^a and each attribute has a specific feature that must be extracted from the SRS document. The automated extraction of each Q^a and Qi requires several NLP techniques. The features extracted from the SRS document require a good method to distribute them within the Q^a and Q^i. The assessment process requires a highly flexible platform that manages high-quality assessment processes. Table 1 shows the summary of related work.

From the literature, four main steps are used to complete the assessment process: pre-processing, processing, post-processing, and reporting. The most popular technique used in pre-processing is NLP, POS, tokenization, and parsing. The K-means have been applied to classify the extracted features based on relative attributes and indicators of SRS. There is more attention on assessing the understandability and ambiguity attributes since these attributes directly influence the quality of the SRS document. This is because the literature has shown that the pre-processing steps increase the accuracy of the assessment process and, in turn, increase the quality of the SRS. It is also noted that the literature has revealed that most works are compared with the performance of human experts due to the absence of public SRS documents that can be used as a benchmark dataset.

All existing methods attempt to assess the quality of the SRS based on a limited number of quality attributes and indicators. This is due to several factors, including assessing each quality attribute requires extracting specific indicators or features and overlapping features between the quality attributes. For instance, the work of Siahaan and Umami [20] introduced an approach to detect the forward reference in the SRS document by using NLP. Carlson and Laplante [21] reconstructed the Automated Requirement Measurement (ARM) method [1] to increase the quality of the SRS document by assessing three QVs which are ambiguity, complete, and understandability. Antinyan and Staron [14] introduced the Rendex method to assess the understandability of the document. Nonetheless, these methods are still lacking in terms of contextual features, which affect the process of constructing relationships between different features, the depth of the analysis, and the assessment process.

Existing works neglect contents such as tables, text analysis, and, more importantly, relationships between the quality features. Subsequently, this research has addressed three main problems related to fully automated SRS quality assessment as follows: (i) The existing framework, such as Reconstructed ARM of [21], only focuses on a specific and limited number of quality attributes and indicators. (ii) There is no standard algorithm to quantify the data that is extracted by the NLP algorithms [28]. (iii) The assessment process of multiple quality attributes for various SRS documents requires a flexible platform that manages automated assessment processes.

3 | METHODS AND MATERIALS

This research proposes an Automated Quality Assessment of Software Requirements Specification (AQA-SRS) framework that can perform SRS quality assessment automatically and simultaneously tackle the problem of multiple or redundant quality attributes (Q^a) at the same time. To achieve this issue, Figure 1 illustrates the research framework's input, process, and output phases along with the research activities.

Based on the figure, the research framework consists of input, process, and output phases. The input phase includes the tasks to review and understand SRS written in natural language along with the related NLP algorithms, quality attributes (Q^a), quality indicators (Q^i), as well as quality assessment approaches and methods sourced from the literature and related work. The investigation also includes the review of a suitable dataset along with the evaluation metrics. The process phase relates to the achievement of Objective 1 and Objective 2, which are to propose the Automated Quality Assessment of SRS (AQA-SRS) framework and the Multi-Agent-based K-means clustering architecture in order to perform quality assessment automatically. The related activities involve analyzing the available methods that assess the quality of the SRS document and proposing a framework that has the potential to improve the

TABLE 1 Summary of related work

Ref.	Methods/techniques	Description	Remarks
[2]	• PoS • Flesch ease readability index	Introduces an automated method to measure the quality of SRS documents based on requirements sentence quality and requirements document quality	• Depth analysis for the sets of requirements is still required. • Need to clarify and explanation on how the process of the expert validation is done. • They used a private dataset.
[6]	• Parsing	They proposed an approach for improving the quality of the SRS document	• Their approach still requires human interaction to evaluate the quality of the document. • The examination of the human expert is unclear. • They used a private dataset.
[14]	• Parsing	They presented a method (called Rendex) for proactive SRS document reviews. Rendex was utilized to evaluate the understandability of the documents based on four internal quality measures of textual requirements.	• Still lacking in terms of depth analysis and the assessment process. • Assessing the quality of the SRS based on understandability. However, they neglected the other QA like semantic, and pragmatic. • They used a private dataset.
[21]	• Parsing • Eliminating whitespace • Flesch reading ease index • Flesch-Kincaid grade level index • Coleman-Liau grade level index • Bormuth grade level index	Proposed an automated method to assess the quality of the SRS document based on four quality attributes via using public datasets.	• Still lacking in terms of analysis depth to the assessment process.
[27]	• PoS • Wordnet • N-gram	Proposed a framework named Word Sense Disambiguation (WSD) for tackling and removing the ambiguity problem in the SRS document.	• Only focus on assessing the quality of the ambiguity attribute with do not care to other issues like complete and correct. • They used a private dataset.
[28]	• PoS • Noun chunking • K-means	Proposed an automated approach for detecting the defect in the SRS document	• The optimal value of k was found based on inertia and statistical gap. • They used a private dataset.
[29]	• Tokenization, • Lowercase converting • Stop word removal • lemmatizing • K-means	Proposed automated method to detect the noise defect in the SRS document	• The K-means algorithm is still suffering from the local optimum problem. • They used a private dataset.

automated quality assessment of the SRS document. The results of this phase are two-fold, (i) an Automated Quality Assessment of SRS (AQA-SRS) framework and (ii) a set of new Q^a and Q^i that can be extracted automatically without the intervention of a human expert. The last phase is the research output, which includes the performance evaluation of the AQA-SRS framework. The PURE and Reconstruction ARM datasets are used to assess the performance of the AQA-SRS framework. This phase has specified the outlines of the research contributions and future work with respect to specific evaluation metrics.

3.1 | Dataset description

This section describes the datasets used here. Two datasets are used to assess the performance of the proposed AQA-SRS framework.

3.1.1 | Natural language SRS dataset

PURE (PUblic REquirements dataset) contains a collection of 79 SRSs documents [30]. These documents are written in natural language, and these SRSs are open-access documents collected from different websites. The dataset contains 34,268 sentences and can be used for NLP tasks that are distinctive in requirements engineering, such as method abstraction identification, and document structure assessment. It can be further annotated as a benchmark for other tasks, such as ambiguity detection, requirements categorization, and identification of equivalent requirements. PURE dataset considered as a sample of the requirements that can be retrieved from the Web made an informally inspected for each document and labelled it according to the following categories as shown in Table 2. The dataset also supplied some first-stage qualitative information.

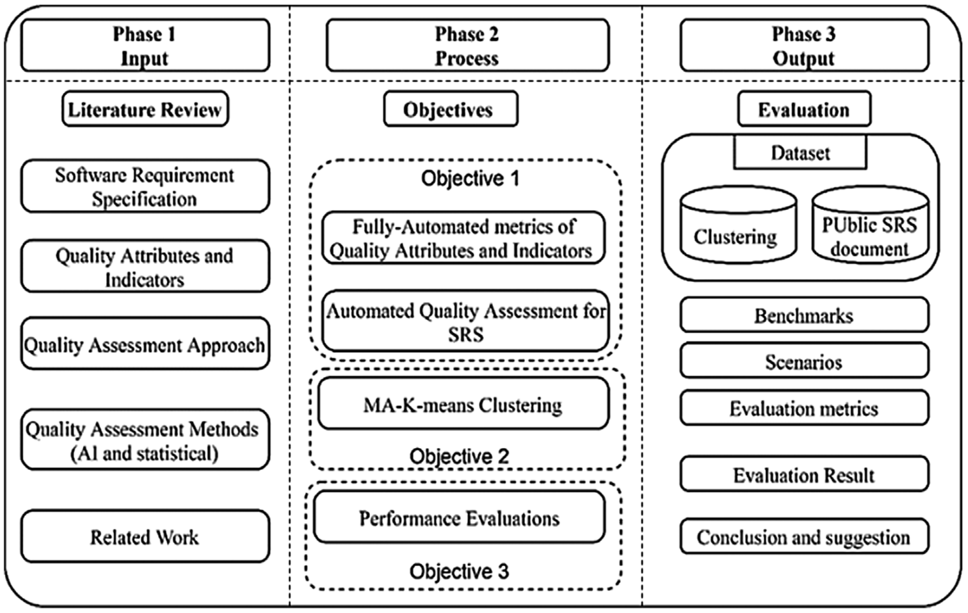

Phase 1
Input
Phase 2
Process
Phase 3
Output
Literature Review
Software Requirement Specification
Quality Attributes and Indicators
Quality Assessment Approach
Quality Assessment Methods (AI and statistical)
Related Work
Objectives
Objective 1
Fully-Automated metrics of Quality Attributes and Indicators
Automated Quality Assessment for SRS
MA-K-means Clustering
Objective 2
Performance Evaluations
Objective 3
Evaluation
Dataset
Clustering
PUblic SRS document
Benchmarks
Scenarios
Evaluation metrics
Evaluation Result
Conclusion and suggestion

FIGURE 1 Research framework

3.1.2 | Natural language SRS dataset

The Reconstruction ARM dataset is a collection of four SRSs documents that are obtained by [21] from different websites. These documents are written in the English language and are suitable for NLP. Experts evaluate them to form a benchmark for testing SRS assessment tools. The dataset belongs to open-source software projects as follows:

1. Warc-tools: A legacy software project consists of tools to facilitate the manipulation and management of web archives, or WARC files (http://code.google.com/p/warc-tools).
2. Vyasa: A digital asset and document management system (http://vyasa.sourceforge.net).
3. PeaZip: An open-source archive utility (http://peazip.sourceforge.net).
4. JHotDraw: A Java GUI framework for technical and structured graphics (http://www.jhotdraw.org).

3.2 | Expert validation

The literature has shown that gold standard evaluation is imperative in the study of SRS quality assessment [6]. Because there are no benchmark SRS datasets and there are several styles of writing SRSs and these styles might be changed based on many factors, including the involved expertise, type of projects, and project domain. For this purpose, human assessment is imperative. The agreement percentage (reliability) between experts and a method is calculated by using Krippendorff's Alpha (α), where α is a statistical measure used to assess the reliability coefficient between two or more observations. Formula 1 shows the probability of the observed agreement between the experts and the proposed method:

$$\alpha = 1 - \frac{D_o}{D_e} \tag{1}$$

where D_o is the observed disagreement among values assigned to units of analysis, the mathematical Equation (2) is presented to calculate the D_o.

$$D_o = \frac{1}{n} \sum_c \sum_k o_{ck\ metric} \delta^2_{ck} \tag{2}$$

D_e is the disagreement one would expect when the coding of units is attributable to chance rather than to the properties of these units. Equation (3) is used to calculate the D_e.

$$D_e = \frac{1}{n\,(n-1)} \sum_c \sum_k n_c n_{k\ metric} \delta^2_{ck} \tag{3}$$

The previous equations are used to calculate the reliability based on two observations only. However, this method is not suitable for our work. For this purpose, this research utilizes Nominal Krippendorff's Alpha ($\alpha_{nominal}$), able to assess the reliability based on several parameters. The following Equation (4) shows the $\alpha_{nominal}$ with respect to different functions.

$$\alpha_{nominal} = 1 - (n_{..} - 1) \frac{\sum_u \frac{1}{n_{u.}-1} \sum_c \sum_{K>c} n_{uc} n_{uk} \delta^2_{ck}}{\sum_c \sum_{c>k} n_{\cdot c} n_{\cdot k\ metric} \delta^2_{ck}} \tag{4}$$

Note that experts assessed each SRS through given a value for each quality attribute (like; complete, correct, and so on).

TABLE 2 Description of the PURE dataset

Type	Letter	Meaning	Description
Doc name	–	–	An alphanumerical (ID) that recognizes the documents
Pages	–	–	The number of pages of the document
Structure	S	Structured	The requirements are stated in a structured format.
	U	Unstructured	Requirements are labelled as unstructured NL descriptions.
	O	One statement	Each requirement is stated in a single NL statement.
	S + U	Structured and unstructured	The document includes structured and unstructured requirements.
Level	H	High-level	Further improvement of the document requires before implementation.
	L	Low-level	The content of the document was ready for implementation.
	U	University	Documents labelled with U normally comprise case studies or assignments performed by university students.
	I	Public or private organization	Documents labelled with I contain an industrial-strength requirement.

This study applies the same rules to the expert and the proposed framework to compare the results.

3.3 | Evaluation metrics

This section provides the evaluation metric used for quantifying the proposed framework's performance, Krippendorff's Alpha (α). α is a statistical method used to assess the reliability coefficient between two or more observations. Equation (4) shows the probability of the observed agreement between the experts and the proposed method, where D_o is the observed disagreement among values assigned to units of analysis, D_e is the disagreement one would expect when the coding of units is attributable to chance rather than to the properties of these units.

4 | AN AUTOMATED QUALITY ASSESSMENT OF SRS FRAMEWORK

This paper attempts to mimic the reviewers' steps when assessing the SRS document's quality. First, the reviewers read the document, and they take a general idea about it. Then, they focus on finding the defect in the SRS document. Next, generate

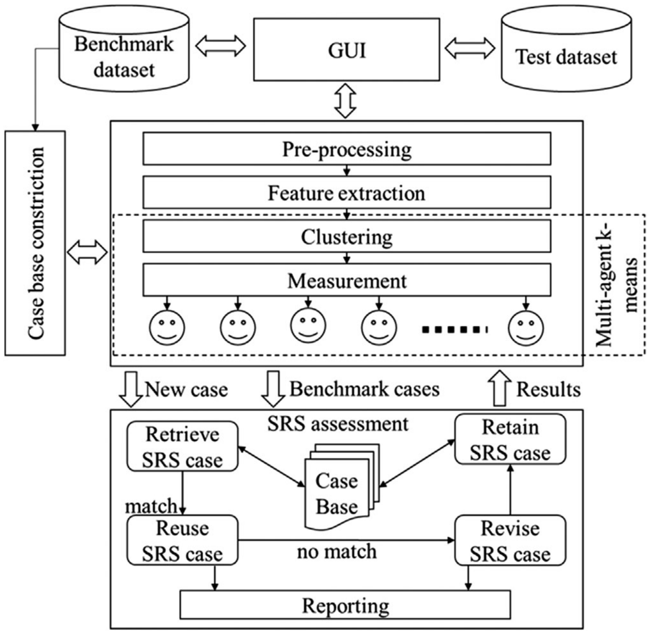


FIGURE 2 Framework of AQA-SRS

a report that shows the overall quality of the SRS. The proposed work also involves NLP algorithms to extract the text from the documents. The AI and statistical methods are used to assess the SRS quality and generate an assessment report. This technique saves time, reduces cost and workload, and produces consistent assessment. The proposed framework of the Automated Quality Assessment for Software Requirements Specification (AQA-SRS) framework is presented in Figure 2. The framework is combined with various elements that interact with each other. Each of the elements has a role in accomplishing the final objective.

The essential part of the method is the Graphical User Interface (GUI), which is responsible for providing interaction to the user, such as asking for parameters, case of preference, benchmarking and testing datasets, the value of K as well as the cluster radius that the CBR requires for the matching process. The benchmark datasets are exploited to create the needed functionalities for identifying the new cases used for case base construction.

The next part is the pre-processing, which arranges the SRS documents in a unique format and standard. The tasks in pre-processing include tokenization and part-of-speech tagging for the entire SRS documents. Additionally, pre-processing is needed when building a normalized bag of words histogram to transform the extracted features from the plain natural text into numeric style. Pre-processing also excludes low-frequency terms due to insignificance. Once the SRS documents have been converted to unstructured text, a feature extraction technique will be applied to extract a significant group of features that would improve the quality of the assessment process in the subsequent phases.

Upon features have been extracted, a clustering process will take place by using the proposed clustering called the

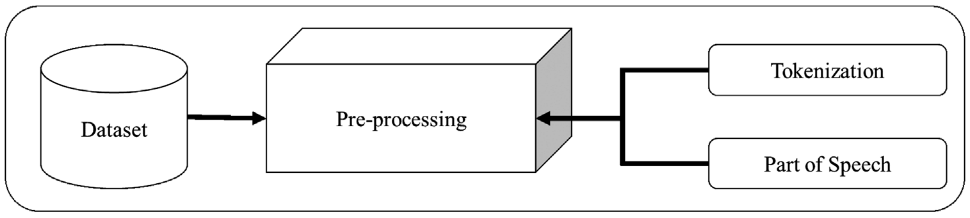


FIGURE 3 Pre-processing steps

Multi-agent K-Means architecture. The multi-agent capability is needed, so the agents play the role of coordinators to synchronize the clustering process based on several Q^a and Q^i. Once clustering has been completed, CBR techniques will begin. The CBR module receives two types of information, the first one is the benchmark dataset, and the second is the processed test case (new SRS). CBR is needed to label the data based on their clustering results automatically. SRS assessments are carried out during this phase as well. The final stage in the method is reporting phase. All the proposed framework and method details are explained with the corresponding equations in the following sections.

4.1 | Pre-processing

Assume that the dataset is combined with a set of SRS documents such that $SRS = \{srs_1, srs_2, ... srs_n\}$ where srs_i Represents one SRS document, and i represents an index. The size of the data is n. The quality of the documents with respect to eleven aspects of Q^i $\{y = (y_1, y_2, ..y_k)\}$ where $y_1 =$ imperative, $y_2 =$ continuances, $y_3 =$ directives, $y_4 =$ options, $y_5 =$ sizes, $y_6 =$ weak-phrases, $y_7 =$ vague words, $y_8 =$ implicit words, $y_9 =$ number of conjunctions, $y_{10} =$ security, and $y_{11} =$ readability statistics. Each one of these Q^i contribute to more than one Q^a.

The Q^a are divided into eleven categories, namely, completeness, correctness, modifiability, rank, testability, traceability, ambiguities, understandability, validatable, verifiability, and complexity. This means that any SRS document from the dataset can be assigned to a number of classes that indicates any possible subset of the Q^a that is found. The following Equation calculates the total possible number of classes,

$$NC = \sum_{i=1}^{11} m\, C_i^{11} \tag{5}$$

where NC denotes the maximum possible number of clusters, m refers to the level of each indicator, and C denotes the combinatory operator. Afterward, two NLP or text processing algorithms are performed, which are tokenization and Part-of-Speech (POS) tagging. Figure 3 shows the pre-processing steps.

The task of tokenization is to split the SRS text into smaller units or tokens. Two types of tokenization appear at the word level or sentence level. Word tokenization divides the sentences into words. Meanwhile, sentence tokenization separates the paragraph into sentences. Once tokenization has been completed, POS tagging is implemented to mark up the words in a text as their corresponding part of speech by relying on definition and context, such as nouns, adjectives, verbs, and adverbs.

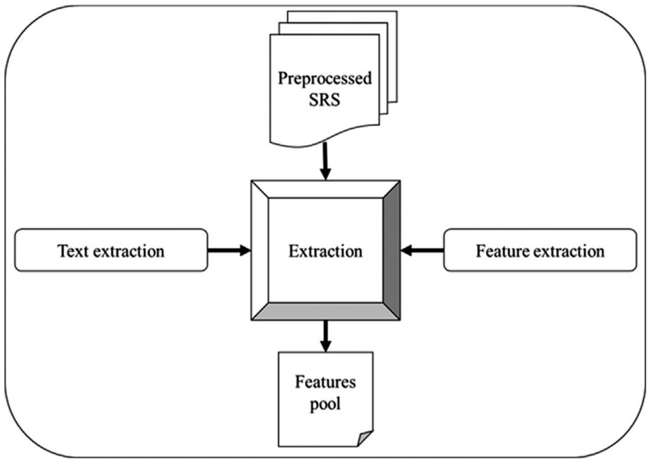


FIGURE 4 Feature extraction process

4.2 | Features extraction

This stage is conducted to extract specific features from the SRS document. These features directly impact the quality assessment process of the SRS document. Figure 4 shows the process for feature extraction. A textual feature extraction operator is utilized to extract the most important features from the SRS document. This process is employed to transform a list of essential phrases/words into a feature (phrases/words) set that is usable by the assessment process.

4.3 | Features extraction

To design an automated measurement metric, one needs to understand what quality attributes and indicators are suitable to make the requirement able to measure automatically. To determine the desired metric, the quality attributes and indicators mentioned in the previous research [1, 2, 14, 31–33] are analyzed and compared. Three main rules are implemented to select the desired metric, which is automated, suitable for NLP operators, and most frequent. Automated refers to the indicators that have

TABLE 3 Automated assessment metric

Indicators of quality attributes

Quality indicator (Q^i)	**Quality attributes (Q^a)**										
	Complete	**Correct**	**Modifiable**	**Ranked**	**Testable**	**Traceable**	**Unambiguity**	**Understandable**	**Validatable**	**Verifiable**	**Complexity**
Imperative	X		X			X	X	X	X	X	
Continues	X		X	X	X	X	X	X	X	X	
Directives	X	X			X		X	X	X	X	
Options	X				X		X	X	X		
Size	X				X		X	X	X	X	X
Weak phrases	X	X			X		X	X	X	X	X
Vague words								X			X
Implicit words	X		X			X	X	X	X	X	
No. of conjunctions							X	X			X
Security			X		X	X					
Readability			X		X	X	X	X	X	X	

a specific list of words or phrases used to assess the document's quality. On the other hand, suitable for NLP refers to the indicators that are able to extract by the NLP operators. The most frequent refers to the attributes and indicators that numerous researchers compute.

Table 3 shows the automated metric that comprises several automated indicators/features. These features have a direct impact on the quality of the requirement document. The indicators Q^i are imperatives, continuances, directives, options, size, weak phrases, vague words, implicit words, number of conjunctions, security, and readability statistics. The Q^i are well defined in the literature, such as in (references).

The quality indicators Q^is are used to assess the quality attributes Q^a, which are complete, correct, modifiable, ranked, testable, traceable, unambiguous, understandable, validatable, verifiable, and complexity in the form of quality metrics, as shown in Table 3. This quality metric is called an automated metric designed to measure the quality of the SRS documents in an automated manner. The automated metric is produced by updating the metrics as proposed in previous works [1, 2, 14, 31, 34, 35] by adding four Q^i which are vague words, implicit words, number of conjunctions, security, and adding one Q^a which is complexity as well as adding a new list of words/phrase to Q^is. The following Table 3 shows the relationship between Q^a and Q^i.

4.4 | Multi-agent K-means clustering

Clustering is a method that uses to divide a group of data into several subgroups. Most of the clustering methods need a priori specification (i.e. number of groups). Another recurrent feature among many popular solutions is the initialization and maintenance of structures such as centroids, which represent their corresponding clusters. To avoid those types of global information about data, this paper proposed an improved K-means clustering algorithm based on Multi-Agent Systems (MAS). The proposed Multi-Agent K-Means (MA-K-means) algorithm aims to build interactive cluster structures from the original data objects.

The conventional K-means algorithm is insufficient because it only assigns the data to a specific cluster. However, in the case of the SRS, one data might be associated with more than one cluster. For instance, the quality attribute Q^a is categorized for both imperative and vague clusters, Q^i. Moreover, the measurement of one feature or Q^i might be affected by other features, as proven by [15]. This entails the flexibility of agent interaction and measurement association in order to come up with accurate calculations. To achieve this goal, some key properties of artificial agents are adopted, such as the ability to sense the surrounding environment, perform autonomous actions, and communicate with each other.

The MA-K-means algorithm works on classifying data by using several agent for performing the clustering considering a different subset of features for each agent. Assuming the total number of features is m, then the total number of a subset of features is $2^m - 1$. The first one will include the first feature, and the second one will include the second feature until all single feature sets are considered, then other sets containing two pairs of features. This process is repeated until the last set that contains all features is considered. The feature sets will be as an input of an agent, which is responsible for generating the clustering results of the data based on its input set. The clustering results will be used for predicting non-labelled data using a voting process that is guided by labelled data. Equation (6) represents the voting process for cluster prediction (CP).

$$CP = argmax_{class}\left(No.\ of\ SRS\ of\ a\ class\right) \quad (6)$$

ALGORITHM 1 Agent-based K-means model

```
Input
$X = \{x1, x2 \dots xN\}$//SRS
$xi = \{f1\ f2 \dots fm\}$ //features
$Y = \{y1, y2 ..yn\}$ //labels n < < N
//n is the number of SRS assessed by an expert
Output
$Yp = \{yk\ y(k+1) \dots y(N-n)\}$ // the predicted classes
Features j
Start
1-Initiate $2 \wedge m - 1$ agents
2-Initiate main agent
3-In parallel for each agent $i$ from $2 \wedge m - 1$ agents
    3.1 C$lusters(i)$ = call K-means with respect to selected features $i$
    3.2 Labelling of clusters based in Y using a voting process
    3.3 Send output $Yp, i$ for agent $i$ and the error $ei$ to the main agent
End
4-Main agent will select the decision of $agent(j)$ with mini $ej$
5-Winning agent $j$ provided the output of non-labeled data $Yp = Ypj$
6-Selected features are features $j$
End
```

The MA-K-means numerates the agents with an index matching the order of the candidate set. The agents, after generating their parameters based on the benchmark datasets, connect with the main agent to provide their scoring of performance to it. Finally, one of the agents will be selected as the winning agent, which is the one that accomplishes the highest score. As it is depicted in the pseudocode, the inputs are X which represents the SRS data, xi, which represents the features codes, Y which represents the benchmark part of the data. The output is the prediction vector Yp, for the non-labelled data. The initialization of the agents is performed in Step 1 until step 3, the operations of the agents for predicting the labelled part of the data are performed in Step 3.1 until Step 3.5. Finally, the agents communicate with the main agent in Step 4, which will select the winning agent from them. The winning agent is the one that accomplishes the least prediction error (maximum prediction accuracy) with respect to the ground truth data. The selection is performed in Step 4, while its results generation for the non-labelled data is performed in Step 5. The proposed agent-based K-Means architecture is shown in Algorithm 1.

In multi-agent clustering, MAS collaborates with the K-means algorithm by providing roles to the agents' to observe the boundaries of each group/cluster and coordinate the related groups of a Q^i for a particular Q^a, and performs the required measurement (i.e. the distance between clusters, voting process). Figure 5 depicts the main steps used in the K-means clustering model. In the first step, initiate $2^m - 1$ agents. Then, implement the K-means to cluster the data based on the number k that started from 1. Next, the main agent observes all the clustering results. Finally, predict the best number of clusters.

Recall that there are four steps before assessment in the proposed AQA-SRS framework shown in Figure 6; pre-processing, processing, clustering, and measurement. The pre-processing stage is responsible for preparing the SRS document. This stage's advantage is removing unnecessary information from the document, reducing the storage space, decreasing the processing time, diminishing the document's overall size, and increasing the accuracy of the extracted features. Next, the feature extraction is conducted in the processing stage to extract all potentially relevant features from the SRS document. The main goal of this stage is to decrease the length of the text. This, in turn, enhances the efficiency of the assessment and classification. The third step uses the multi-agent and K-means capabilities to cluster the extracted features from the SRS document. Bag-of-Words (BoW) is implemented to convert the unstructured vector of features into a numeric vector. Finally, the output of this phase is a group of clusters that comprise a group of features; each cluster belongs to a specific Q^i.

Similarly, clustering of quality attributes Q_n^a must break this assumption and propose a clustering model that is applicable for the non-standard or spherical distribution of clusters. This work presents a new model that calculates the inertia of Mahalanobis distance instead of classical Euclidian distance. Mahalanobis distance is given by Equation (7), based on SRS document x, cluster with center u, and covariance matrix m.

$$D_{(x,u,m)} = \sqrt{(x-u)^T Q^{-1} (x-u)} \tag{7}$$

Note that in special cases when the cluster is spherical, m will become an identity matrix which will turn the measurement into a classical Euclidian distance formula. In order to visualize the way of handling the non-spherical shape of the cluster by using the Mahalanobis distance.

The differential of inertia can detect the optimal number of clusters when the differential of inertia becomes steady (approximately at $K = 3, 4$). Next, in order to automatically detect the optimal value of K, a condition, as shown in Equation (8).

$$k^* = \textit{the lower } k \textit{ that gives } |I_k - I_{k-1}| < \in \tag{8}$$

From the Equation, the value of k is incremented by one from a starting value $k_0 = 1$ in order to calculate the value of inertia I_k as a function of k. $\in$ refers to a predefined small value in order to show the changing of inertia and the detection of the number of clusters according to the value of k. Synthetic data is generated based on a random selection of the centers, and the potential number of clusters and the value of the sum of the distances of intra-connections inside the clusters are calculated.

4.5 | Case-based reasoning

Case-Based Reasoning (CBR) architecture is employed to fuse the result of clustering that is non-labelled with the available labels in the database to provide the functionality of predicting the class of tested SRS. CBR mainly conducts statistical voting to the class based on the percentage of labels inside the predicted cluster. In other words, if a sample is decided to belong to a certain cluster, the class prediction function will calculate the percentage of each potential class in the cluster and decide which class has the highest percentage, then this class is labelled

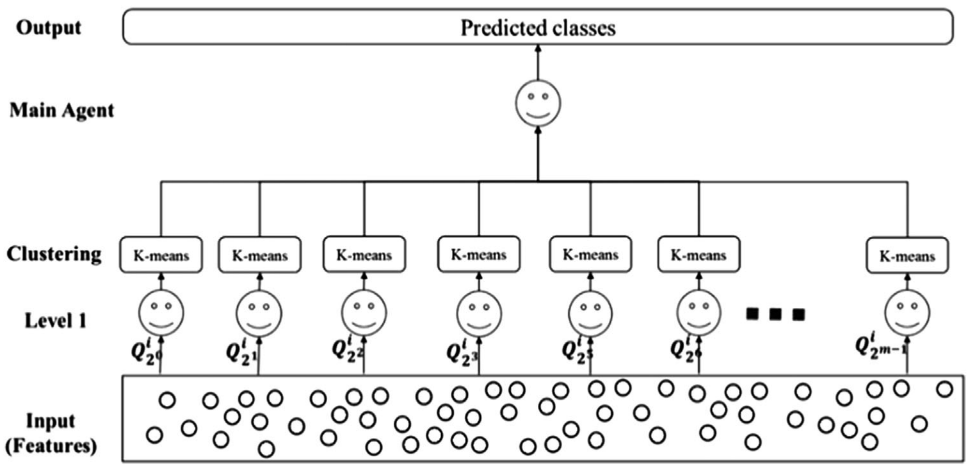
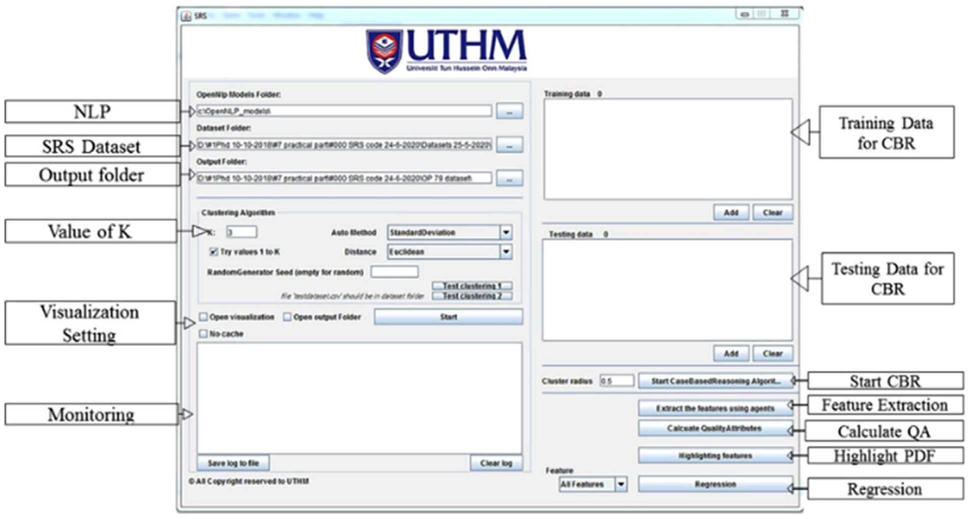


FIGURE 5 Agent-based K-means clustering architecture



OpenNlp Models Folder: c:\OpenNLP_models\
Dataset Folder: D:\#1Phd 10-10-2018\#7 practical part\#000 SRS code 24-6-2020\Datasets 25-5-2020\
Output Folder: D:\#1Phd 10-10-2018\#7 practical part\#000 SRS code 24-6-2020\OP 79 dataset\
Clustering Algorithm
K: 3
Auto Method: StandardDeviation
[✓] Try values 1 to K
Distance: Euclidean
RandomGenerator Seed (empty for random):
File 'testdataset.csv' should be in dataset folder
[ ] Open visualization [ ] Open output Folder
[ ] No cache
© All Copyright reserved to UTHM
Training data 0
Testing data 0
Cluster radius: 0.5
Feature: All Features

FIGURE 6 The GUI of AQA-SRS framework

accordingly. This process is depicted by Equation (9),

$$\text{PC}\ (x) = \ \text{argmax}\left(\text{percentage}\left(\text{samples}\ (x)\right)\right) \tag{9}$$

where PC denotes the predicted class of a sample data that cluster x is assigned to. It is a function that calculates the percentage of contribution of each class in the cluster from the perspective of a number of points, and argmax is a function of finding the maximum value.

In the proposed AQA-SRS framework, the role of the CBR is to enable SRS test comparisons with the benchmark SRSs assessed by experts in order to construct the case base. Four steps are used in order to put CBR in its formalized way, which use to evaluate the similarity between new SRS and the benchmark SRSs that retain the case-base. CBR requires four major steps, which are retrieve, reuse, revise, and retain [3–5]. The explanation of each step is as follows:

Retrieve: It has the task of comparing the new SRS with the existing SRS and determining the one that is similar to the current SRS. In the CBR terms, the new SRS is named as a test case, and the old stored ones as benchmark cases.

Reuse: It is responsible for mapping the solution of the benchmark cases that are most similar to the test case.

Revise: This is responsible for evaluating the performance after the reuse step was conducted and is done by finding the new accuracy considering that new labelled data was tested.

Retain: The final step is to accept the new case in case base for future usage, which is done by judging the newly obtained accuracy from the revising step.

The case base in Figure 6 is constructed by adding new tested SRS samples SRS to the existing clusters if the distance from their centres is lower than the radius. This enables a change in the cluster centre gradually as new points are added to it, which gives the cluster the capability to track new changes. At the same

ALGORITHM 2 Pseudocode of case-based construction

```
Input
    New case (SRS);
    Benchmark Dataset;
Output
    Update benchmark dataset;
    The label of the new case;
Start
    Extract features of the new case;
Measure the distance of the new case from the benchmark dataset;
    Find the least record distance
Label the new case according to the least record distance;
Calculate the distance between the new case and the center of the new
    label cluster;
If (distance < = radius)
    Add the new case to the cluster;
Modified benchmark dataset = the benchmark dataset+ new case;
Else
    Keep the same benchmark dataset;
End
```

time, it enables using the newly added points as examples of cases for labelling or reasoning future points. Over time, the case base will contain a richer set of examples of records from the training dataset. This allows the AQA-SRS framework to be dynamic and adaptable to new changes in the documents. Algorithm 2 represents the pseudocode of the case base construction process.

From the algorithm, if the SRS has a standard format and has been appropriately constructed, the matching algorithm of the case base construction automatically decides to include it as a standard case, for instance, IEEE standard SRS format [34, 36]. However, in many cases, the SRS has no clear format and only include definitions of requirement and specifications. Such SRS cannot be assessed easily in a proper manner by an automated tool and definitely cannot be used as a standard SRS. To deal with such cases, the human expert is needed to decide whether to include or exclude the SRS from the case base. Additionally, ensuring accurate decisions by an algorithm is very costly because it relies on many offline factors.

4.6 | Report generation

The final phase in the proposed AQA-SRS framework is the detailed report, which contains a comprehensive illustration of all assessment process parts. The first part of the quality report contains the main information (namely, project ID, project name, and date). The next part of the report involves the analysis result of Q^i. In the third part, the Q^a result is reported. Finally, the overall quality of the SRS document is stated. The overall quality of the SRS document depends on the percentage of Q^a, and each Q^a relies on different Q^i. However, the result of Q^i is calculated by counting the defects on the SRS document. To solve this hierarchal problem, the AQA-SRS performed a normalization equation to calculate the percentage of each Q^i. This, in turn, helps calculate the SRS document's overall quality. The following equation calculates the overall quality of the SRS document.

$$SRS\_quality = \sum_{i=1}^{Q^a} \sum_{j=1}^{Q^i} \omega_i * \Delta Q^i \tag{10}$$

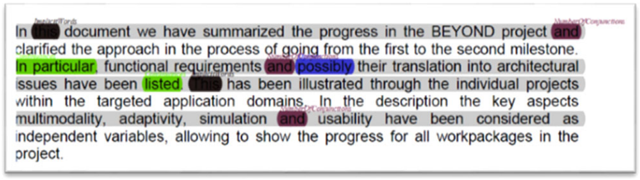
where ω represents the correlation coefficient for each Q^i. Δ refers to the number of indicators that are responsible for assessing the individual quality attribute, Q^a.

5 | IMPLEMENTATION AND RESULTS

The AQA-SRS framework is implemented as a tool that is capable of assessing the overall quality of the SRS documents, such as completeness, correctness, and ambiguities. The implementation of the proposed AQA-SRS framework is described in this section. The tool is implemented in Java programming language in a Java-based agent development platform called JADE [37]. Figure 6 displays the GUI of the tool. It consists of five main processes, which are input/output, clustering, visualization, case base construction, and finally, calculation and highlights.

The details of the main processes of the AQA-SRS tool are as follows:

- Input and output: This part of the tool consists of three main windows; choose the Natural Language Processing (NLP) techniques, access to the database that contains the SRSs, and specify the output location. These windows allow the user to browse and select the preferred location to fetch and send that data.
- Clustering specifications: This part allows the user to either specify the value of K (number of the cluster) as the seed point or let the tool determine them in an automated way. Further, this process permits the user to select the method and distance metric for clustering.
- Visualization setting: This part is responsible for displaying all the events that occur in the system within the processing part, opening a new window to visualize the quality assessment output.
- Case base construction: This window authorizes the user to use the CBR by adding the training and testing SRS and specifying the threshold for matching. All details are illustrated in Section 4.5.
- Calculation and highlights: This part is responsible for displaying features extracted through NLP, calculating the overall quality for SRSs, highlighting the extracted features, and calculating the regression.
- After completing all five of the processes, AQA-SRS will produce output from two main perspectives. The first is to highlight all defects within the SRS documents. The second is to generate an assessment report that consists of all information related to the overall quality assessment of the SRS documents. The details of the outputs of the AQA-SRS tool are as follows:
- Defect highlighting: The AQA-SRS used a modern method to display the defects in the SRS document. The method



In this document we have summarized the progress in the BEYOND project and clarified the approach in the process of going from the first to the second milestone. In particular, functional requirements and possibly their translation into architectural issues have been listed. This has been illustrated through the individual projects within the targeted application domains. In the description the key aspects multimodality, adaptivity, simulation and usability have been considered as independent variables, allowing to show the progress for all workpackages in the project.

FIGURE 7 AQA-SRS highlighting example

was prepared to facilitate the user's task of tracking errors and reducing time. It is worth mentioning that the method is adopted from state-of-the-art [38]. The AQA-SRS used distinct colours to highlight the defects. Each colour refers to a specific kind of defect. Further, define the type of defect (i.e. implicit word, continues, and so on).

Figure 7 shows a simple example of a defect highlighted by the AQA-SRS tool. Each defect has been coloured with a distinct colour along with the name of the quality indicator (Q^i) that consists of this type of defect. The method simplifies the understanding and discovery of the redundancy of the defects in the paragraph and document.

5.1 | Comparative evaluation

This section provides a statistical evaluation of the number of Q^a and Q^i for three methods such; ARM, Reconstructed ARM, and AQA-SRS. This evaluation aims to show each method's ability to extract the defect from the SRS document. This, in turn, increases the quality of the assessment for the SRS document. The Reconstruction ARM dataset is used to produce a statistic quality report to conduct the evaluation. While the AQA-SRS was evaluated and refined using a number of additional features, these features provide a representative sample of some of the differences encountered between the AQA-SRS, reconstructed ARM, and ARM.

In order to compare the three tested methods, two main perspectives are utilized. First is the number of quality attributes (Q^a) and their respective indicators (Q^i) that are used to assess the quality of the SRS document. Second is the number of features extracted from each SRS document. This statistical analysis is used as a foundation to generate the SRS quality assessment report. The report contains the indicators that require improvement. The results of the comparative analysis between the Q^a and Q^i of the three methods are shown in Figure 8.

The ARM uses 11 Q^a, the reconstructed ARM uses five Q^a, and the AQA-SRS uses 11 Q^a to evaluate the quality of the SRS document (Table 3). In contrast, the ARM and reconstructed ARM used 9 and 5 Q^i, respectively, whereas the AQA-SRS used 11 Q^i. These Q^a and Q^i are employed to find the overall quality of the SRS document. Table 4 represents the number of extracted features by three frameworks.

The statistical analysis for the number of extracted features from tested SRS documents is represented in Figure 9a–d. The results show that the three methods have an equal number of

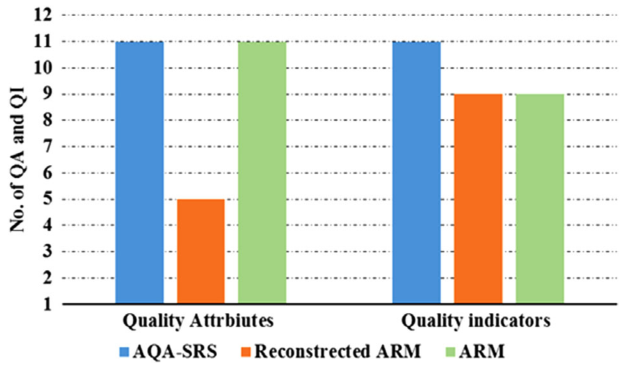


FIGURE 8 No. of Q^a and Q^i

TABLE 4 No. of Q^a and Q^i

Indicator	SRS	AQA-SRS	Reconstructed ARM	ARM
Imperatives	warc	135	135	135
	Peazip	28	25	26
	JHot draw	34	34	34
	Vyasa	18	18	18
Continuances	warc	158	158	158
	Peazip	350	350	340
	JHot draw	235	225	227
	Vyasa	132	132	130
Options	warc	22	17	17
	Peazip	11	4	3
	JHot draw	21	16	14
	Vyasa	7	2	2
Weak Phrases	warc	18	15	15
	Peazip	60	53	53
	JHot draw	88	73	73
	Vyasa	5	3	3
Directives	warc	9	5	5
	Peazip	8	3	3
	JHot draw	12	3	3
	Vyasa	6	1	1

imperative extracted features. This is because the same terms (words/phrases) are used. But the variation can be seen in the rest of the Q^i.

The AQA-SRS method outperformed the other two methods in the continues, directives, options, and weak phrases. Due to this, the AQA-SRS defined a new group of features that are collected based on the state-of-the-art methods and the experts.

5.2 | Expert validation

Thus, the results of the proposed framework are compared with results produced by the human expert. It is worth mentioning

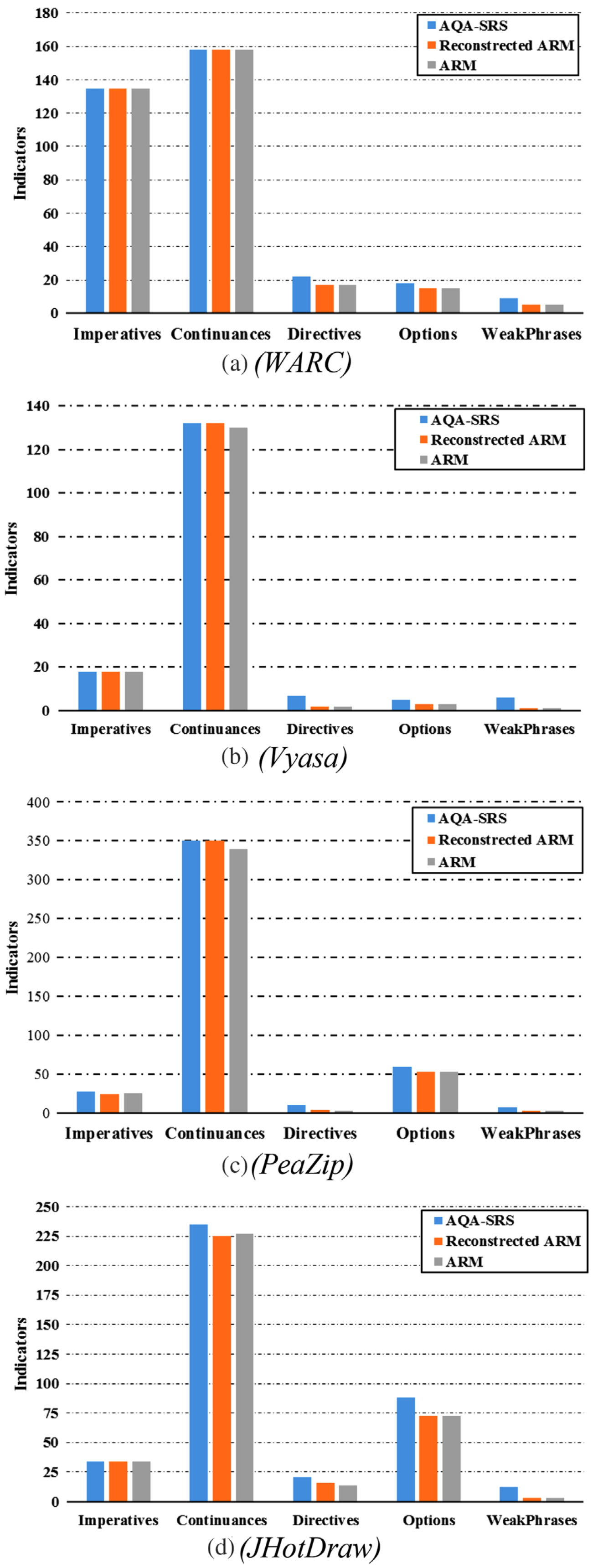


FIGURE 9 Statistical analysis for extracted features. (a) (WARC), (b) (Vyasa), (c) (PeaZip), (d) (JHotDraw)

that 10 SRS documents are assessed by the software engineering experts and used to calculate the agreement percentage (reliability) of the proposed AQA-SRS framework. These SRSs documents are submitted to the software engineering company as a domain expert. Two experts carry out the quality assessment of the SRSs, (1) one software engineer with four years of experience and (2) a software analyst with eight years of experience. The total evaluation period for ten SRSs has taken seven weeks. For privacy matters, we are not allowed to reveal the identity of the companies. Each SRS document refers to a different project from several problem domains, as depicted in Table 5.

The material supplement file contains the manual assessment results as Appendix A1. Experts assess each SRS by giving a value for each quality attribute. This paper applies the same rules to the expert and the proposed framework to compare the results.

Based on Equation (4), the $\alpha_{nominal}$ for all tested SRSs is equal to 78%. This means the method is considered reliable due to the $\alpha_{nominal}$ value is between 0 and 100, 100 indicates perfect reliability, and 0 indicates the absence of reliability. The consistency between the AQA-SRS and the expert is acceptable. As the AQA-SRS implemented several methods and techniques for pre-processing the data, these methods and techniques helped construct a reliable framework.

5.3 | Analysis and discussion

SRS is one of the early stages of the software engineering process. It mainly documents functional and non-functional requirements and establishes the tasks the software application is set to accomplish. Consequently, the development team must fulfil all the requirements stated in the SRS documents to ensure the quality of the software product. Poorly written SRS has a negative impact on the entire project and might cause project failure. Generally, some motivational points show the need for automatic SRS review, and these points are given as follows. Firstly, the automatic review is faster and provides instant feedback. For instance, the feedback for a paragraph review can be achieved by only 500 ms. Secondly, an automatic review is of a low cost. The high cost of performing manual reviews remains a major problem. Thirdly, manual review is associated with human error, while automated review provides consistent assessment. With the manual method, two different reviewers can provide different results on different occasions for a requirement even though this can be advantageous in terms of quality factors, such as if the requirement satisfies a given guideline, but it paves the way for inconsistencies. Finally, the approach of manual inspection of the SRS documents by multiple reviewers has the challenge of integrating the review results into one document. It is because of the difference in the depth of the reviews as some reviews are only performed sporadically or superficially [14].

Many researchers proposed different automated methods or tools to assess the quality of the SRS document to tackle these challenges. The quality of SRS is measured in terms of Q^a, for instance, correctness, completeness, and understandability. In

TABLE 5 Assessed SRS

No.	Domain	No. of page	Year	Structure	Level	Source	No. of pictures/ table
1	Wind turbine applications	69	2001	Unstructured	H	Ind	F
2	EVLA Correlator monitor and control	17	2002	Unstructured and one statement	H	Ind	M
3	Triangulation games	23	2005	Structured	H	Uni	–
4	Grid 3D application	15	2005	Unstructured and one statement	H	Uni	M
5	Water use tracking system's	42	2007	Unstructured and one statement	H	Ind	F
6	Video search engine	14	2009	Unstructured and one statement	H	Uni	–
7	Partition editor GUI	26	2010	Unstructured	H	Uni	M
8	GAMMA-J's web store	44	–	Unstructured and one statement	H	Uni	F
9	PDF split and merge	34	2010	Unstructured and one statement	L	Ind	M
10	Smart house	15	2010	One statement	L	Uni	–

Ind, Industry; Uni, University; H, High, L, Low; F, Few; M, Many.

general, the existing SRS quality attributes combine automated, semi, and non-automated. The semi and non-automated quality assessment process requires a review session by humans. However, the assessment process is hugely dependent on the expertise of human experts. The judgment of human experts could also be inconsistent due to numerous factors, including experience, knowledge, and domain. Also, it is costly and time-consuming. To overcome these issues, this paper focused on an automated quality assessment method in SRS documents. Nonetheless, there is no known metric used to measure fully automated quality attributes [14, 21, 29]. Likewise, the problem is more complicated when one simultaneously assesses different quality attributes. Because each quality attribute can be identified based on a specific feature set, they can be extracted from the SRS document.

This paper identifies three main problems related to automated SRS quality assessment. It attempts to solve these problems by achieving three research objectives. The following illustrates the identified problems, the corresponding objectives, and the achievement of the objectives. The first problem that has been addressed in this paper is a weakness in extracting features from the SRS documents [21, 26, 14] which will affect the process of constructing relationships between different features, the depth of the analysis, and the overall assessment process.

To address this problem, this paper proposed the AQA-SRS method by incorporating NLP for feature extraction, a multi-agent system integrated with the K-means algorithm for clustering the extracted features, and CBR for process management and reporting. To achieve this objective, two main perspectives are required. Firstly, the need for a group of Q^a and Q^i that can be measured automatically. Second, there is a need for a flexible platform that allows the quality assessment of the SRS documents to be performed based on the Q^a and Q^i. For this purpose, this research analyzed the literature and defined a new group of Q^a and Q^i that can be measured in an automated way. AQA-SRS method operates based on four main steps, which are pre-processing, processing, post-processing, and reporting.

In the pre-processing step, tokenization and Part-of-Speech (PoS) tagging techniques are utilized to prepare the SRS document for the subsequent assessment step (processing). These techniques are used to increase the accuracy of the feature extraction process. In the processing step, specific features and text are extracted and kept until the next step of post-processing. MAS and K-means algorithms are employed for performing collaborative clustering to the extracted features. Finally, during the post-processing and reporting steps, the CBR is adopted to construct all the SRS assessment cases and reuse them for future assessment by relying on matching and mismatching techniques as well as managing the overall reporting step. All these algorithms and techniques are integrated and cooperative to achieve the desired objectives.

The second problem that has been addressed in this paper is that there exists redundancy among the quality attributes. To avoid this problem, the features or indicators extracted from the SRS documents need to be clustered into corresponding quality indicators in order to be measured. While previous work used the K-means algorithm to perform this task [29], this paper formulated a Multi-Agent K-means (MA-K-means) model for handling the automated metrics in the AQA-SRS method. The MA-K-means model is able to automatically find the optimal number of clusters and the labels for each given dataset and compute a distance similarity based on different parameters.

Finally, the third problem that has been addressed here is the lack of a benchmark dataset and standard evaluation methodology. The related work, in general, used private datasets during testing or collaborated with industrial companies and performed private tests. For instance, the work of [2, 28, 39]; uses a private dataset that is not accessible online. This issue difficult the process of validating the proposed methods using a benchmark dataset. Furthermore, the related work, such as the work of [29], mainly uses expert validation without specifying the details of the validation process. These issues made the validation processes become challenging.

To attain the evaluation, this paper constructs a new dataset and produces a rich case base by using several SRSs. Furthermore, it uses two standard datasets, which are adopted from PURE and reconstructed ARM, to evaluate the performance of the proposed AQA-SRS. The PURE and Reconstruction ARM datasets contain 79 and 4 SRS documents, respectively. Several performance metrics, including a comparative analysis of Q^a and Q^i and the agreement percentage, are computed to evaluate the performance of the tested methods.

6 | CONCLUSIONS

The main focus of this paper is represented by automatic quality assessment in SRS documents. Subsequently, this paper presents an Automated Quality Assessment of SRS (AQA-SRS) framework. The framework integrates an NLP for feature extraction, a multi-agent system with K-means for features clustering, and CBR for process management and reporting. The AQA-SRS framework includes 11 Q^a and their corresponding 11 Q^i that are constructed based on a deep analysis of the SRS textual content. The framework is implemented in Java, and the test results reveal that the developed AQA-SRS method efficiently assesses SRS documents. It achieves a total agreement of 78% with the two tested methods: ARM and Rendex, and software engineering experts. Furthermore, in terms of continuation, directions, alternatives, and weak phrases, the AQA-SRS method outperforms the other methods. As a result, the AQA-SRS specified a new set of SRS features based on the most recent research and expertise. There are two areas where future research can be more relevant: (i) Increase the document analysis operators to find all significant features that directly impact the quality of the requirements, and (ii) better solving the Q^a such as complete, correct, and ambiguous by tackling the semantic, syntactic, and logical aspects.

AUTHOR CONTRIBUTIONS

M.A.J.: Conceptualization; Methodology; Writing—original draft; Writing—review and editing. A.M.: Formal analysis; Methodology; Supervision. M.A.M.: Data curation; Methodology

ACKNOWLEDGEMENTS

The authors express appreciation to the Malaysia Ministry of Higher Education (MoHE). This research was funded by the Fundamental Research Grant Scheme (FRGS/1/2019/ICT04/UTHM/03/1) grant vot number K210. The authors would like to thank the Center of Intelligent and Autonomous Systems (CIAS) at the Faculty of Computer Science and Information Technology (FSKTM), Universiti Tun Hussein Onn Malaysia (UTHM), for supporting this work.

CONFLICT OF INTEREST

The authors do not have a conflict of interest.

DATA AVAILABILITY STATEMENT

All data are available.

ORCID

Mohammed Ahmed Jubair https://orcid.org/0000-0001-6719-2922

REFERENCES

1. Wilson W.M., Rosenberg L.H., Hyatt L.E.: Automated analysis of requirement specifications. In: Proceedings of the 19th International Conference on Software Engineering, pp. 161–171, Boston Massachusetts USA, (1997)
2. Nordin A., Ahmad Zaidi N.H., Mazlan N.A.: Measuring software requirements specification quality. J. Telecommun. Electron. Comput. Eng. 9(3–5), 123–128 (2017)
3. Mostafa S.A., Jani H.M.: Online checklist-based approach to software requirements specifications quality analysis. In Proceedings of 1st TNB ICT Technical Conference, College of Information Technology. (2011, February)
4. Jani H., Islam T.: A framework of software requirements quality analysis system using case-based reasoning and neural network. In: *Proceedings of the 2012 6th International Conference on New Trends in Information Science and Service Science and Data Mining ISSDM 2012*, pp. 152–157 (2012)
5. Jani H.M.: Applying case-based reasoning to software requirements specifications quality analysis system. In: The 2nd International Conference on Software Engineering and Data Mining, pp. 140–144 (2010)
6. Ali S.W., Ahmed Q.A., Shafi I.: Process to enhance the quality of software requirement specification document. In: 2018 International Conference on Engineering and Emerging Technologies (ICEET), pp. 1–6 (2018), https://doi.org/10.1109/ICEET1.2018.8338619
7. Femmer H.: Automatic requirements reviews - potentials, limitations and practical tool support. In: International Conference on Product-Focused Software Process Improvement, pp. 482–496 (2017), https://doi.org/10.1007/978-3-319-69926-4
8. Parnas D.L., Lawford M.: Inspection's role in software quality assurance. IEEE Software 20(4), 16–20, (2003), https://doi.org/10.1109/MS.2003.1207449
9. Thitisathienkul P., Prompoon N.: Quality assessment method for software requirements specifications based on document characteristics and its structure. In: *Proceedings of the Second International Conference on Trustworthy Systems and Their Applications*, Hualien, Taiwan, pp. 51–60 (2015), https://doi.org/10.1109/TSA.2015.19
10. Alshazly A.A., Elfatatry A.M., Abougabal M.S.: Detecting defects in software requirements specification. Alexandria Eng. J. 53(3), 513–527 (2014), https://doi.org/10.1016/j.aej.2014.06.001
11. Haque M.A., Rahman M.A., Siddik M.S.: Non-functional requirements classification with feature extraction and machine learning: An empirical study. In: 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT) (2019), https://doi.org/10.1109/ICASERT.2019.8934499
12. Femmer H.: Requirements quality defect detection with the qualicen requirements scout. *CEUR Workshop Proceedings*, Vol. 2075 (2018)
13. Rossanez A., Carvalho A.M.B.R.: Semi-automatic checklist quality assessment of natural language requirements for space applications. In: *Proceedings of the 2016 Seventh Latin-American Symposium on Dependable Computing (LADC)*, pp. 123–126 (2016), https://doi.org/10.1109/LADC.2016.26
14. Antinyan V., Staron M.: Rendex: A method for automated reviews of textual requirements. J. Syst. Software 131, 63–77 (2017), https://doi.org/10.1016/j.jss.2017.05.079

15. Saavedra R., Ballejos L.C., Ale M.A.: Software Requirements Quality Evaluation: State of the art and research challenges. In XIV Simposio Argentino de Ingeniería de Software (ASSE)-JAIIO 42 (2013)
16. Mostafa S.A., Gunasekaran S.S., Khaleefah S.H., Mustapha A., Jubair M.A., Hassan M.H.: A fuzzy case-based reasoning model for software requirements specifications quality assessment. Int. J. Adv. Sci. Eng. Inf. Technol. 9(6), 2134 (2019), 10.18517/ijaseit.9.6.9957
17. Rashwan A.: Automated quality assurance of non-functional requirements for testability (Doctoral dissertation, Concordia University). (2015)
18. Hussain I., Ormandjieva O., Kosseim L.: Automatic quality assessment of SRS text by means of a decision-tree-based text classifier. In: *Proceedings of the Seventh International Conference on Quality Software (QSIC 2007)*, Portland, OR, pp. 209–218 (2007), https://doi.org/10.1109/QSIC.2007.4385497
19. Godois L.M., Adamatti D.F., Emmendorfer L.R.: A multi-agent-based algorithm for data clustering. Prog. Artif. Intell. 9(4), 305–313 (2020), https://doi.org/10.1007/s13748-020-00213-3
20. Siahaan D., Umami I.: Natural language processing for detecting forward reference in a document. IPTEK J. Technol. Sci. 23(4), 138–142 (2012), 10.12962/j20882033.v23i4.99
21. Carlson N., Laplante P.: The NASA automated requirements measurement tool: A reconstruction. Innov. Syst. Softw. Eng. 10(2), 77–91 (2014), https://doi.org/10.1007/s11334-013-0225-8
22. Stephen E., Mit E.: Framework for measuring the quality of software specification. J. Telecommun. Electron. Comput. Eng. 9(2–10), 79–84 (2017)
23. Bakar N.H., Kasirun Z.M., Salleh N.: Terms extractions: An approach for requirements reuse. In: *2015 IEEE 2nd International Conference on Information Science and Security (ICISS)*, Seoul, Korea, pp. 31–34 (2016), https://doi.org/10.1109/ICISSEC.2015.7371034
24. Bakar N.H., Kasirun Z.M., Salleh N.: Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review. J. Syst. Softwares 106, 132–149 (2015), https://doi.org/10.1016/j.jss.2015.05.006
25. Génova G., Fuentes J.M., Llorens J., Hurtado O., Moreno V.: A framework to measure and improve the quality of textual requirements. Requir. Eng. 18(1), 25–41, (2013), https://doi.org/10.1007/s00766-011-0134-z
26. Thitisathienkul P., Prompoon N.: Quality assessment method for software development process document based on software document characteristics metric. In: Ninth International Conference on Digital Information Management (ICDIM 2014), Phitsanulok, Thailand, pp. 182–188 (2014), https://doi.org/10.1109/ICDIM.2014.6991412
27. Husain M.S., Khanum M.A.: Word sense disambiguation in software requirement specifications using Wordnet and association mining rule. In: ICTCS '16: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, pp. 4–7 (2016), https://doi.org/10.1145/2905055.2905179
28. Mezghani M., Choi J.K., Sèdes F.: A clustering approach for detecting defects in technical documents. 13th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2018), Sep 2018, Cracovie, Poland. pp. 27–33, ffhal-02191796f (2018)
29. Manek P.G., Siahaan D.: Noise detection in software requirements specification document using spectral clustering. JUTI J. Ilm. Teknol. Inf. 17(1), 30 (2019), 10.12962/j24068535.v17i1.a771
30. Ferrari A., Spagnolo G.O., Gnesi S.: PURE: A dataset of public requirements documents. In: *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp. 502–505 (2017), https://doi.org/10.1109/RE.2017.29
31. Antinyan V., Staron M., Sandberg A., Hansson J.: A complexity measure for textual requirements. In: *Proceedings of the 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, pp. 148–158 (2017), https://doi.org/10.1109/IWSM-Mensura.2016.030
32. Oo K.H., Nordin A., Ritahani A.R., Sulaiman S.: An analysis of ambiguity detection techniques for software requirements specification (SRS.). Int. J. Eng. Technol. 7(2.29), 501 (2018), 10.14419/ijet.v7i2.29.13808
33. Abad Z.S.H., Karras O., Ghazi P., Glinz M., Ruhe G., Schneider K.: What works better? A study of classifying requirements. In: *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp. 496–501 (2017), https://doi.org/10.1109/RE.2017.36
34. IEEE: Recommended practice for software requirements specification, IEEE Std 830–1993, IEEE Computer Society. Software Engineering Standard Comminttee of the IEEE Std Computer Society (1998)
35. Doe J.: *IEEE recommended practice for software requirements specifications*. (2011)
36. Board I.E.E.E.-S.A.S.: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Std 1471–2000, pp. 1–23 (2000)
37. Mesbahi N., Kazar O., Benharzallah S., Zoubeidi M., Bourekkache S.: Multi-agents approach for data Mining based K-Means for improving the decision process in the ERP systems. Int. J. Decis. Support Syst. Technol. 7(2), 1–14 (2015), https://doi.org/10.4018/IJDSST.2015040101
38. Sabriye W.M.N.W.Z., Jim'ale A.O.: A framework for detecting ambiguity in software requirement specification. In: *2017 8th International Conference on Information technology*, pp. 1431–1433 (2017), https://doi.org/10.1007/978-3-662-53120-4_6466
39. Mezghani M., Kang J., Sèdes F.: Using k-means for redundancy and inconsistency detection: Application to industrial requirements. Lect. Notes Comput. Sci. 10859 LNCS(June), 501–508 (2018), https://doi.org/10.1007/978-3-319-91947-8_52

How to cite this article: Jubair, M.A., Mostafa, S.A., Mustapha, A., Salamat, M.A., Hassan, M.H., Mohammed, M.A., AL-Dhief, F.T.: A multi-agent K-Means with case-based reasoning for an automated quality assessment of software requirement specification. IET Commun. 1–17 (2022). https://doi.org/10.1049/cmu2.12555