A Semi-Automated Approach for Classifying Non-Functional Arabic User Requirements using NLP Tools

Abstract—**Requirements Engineering is a critical phase in the software development life cycle, encompassing both Functional Requirements (FR) and Non-Functional Requirements (NFR). NFR defines the quality attributes of the system, including performance, security, availability, look and feel, fault tolerance, legal and operational, essential for meeting user needs and imposing additional constraints on software quality. Prioritizing NFR from user requirements is challenging, requiring specialized skills and domain knowledge. Manual categorization is time-consuming and mentally taxing for developers, making automated or semi-automated classification of NFR from requirements documents valuable. This approach reduces manual effort and time in identifying specific NFR among numerous requirements. This paper introduces a novel semi-automated categorization approach for Arabic Non-functional user requirements using CAMeL Tools, a natural language processing tool as of the process of extending computer science conference paper. We propose a set of heuristics based on fundamental Arabic sentence constructions to extract information and categorize requirements into seven NFR classes. Tokens, PoS tags, and lemmas of parsed user requirements are generated using CAMeL tools. The closest class for each statement is determined by applying heuristic criteria to CAMeL outputs. The implementation of our approach using CAMeL Tools 1.3.1 and Python code in a Windows 10 environment demonstrates its practical applicability and efficiency in classifying Arabic Non-functional user requirements.**

Keywords—Requirements Engineering, Functional Requirements, Non-Functional Requirements, natural language processing tool.

I. INTRODUCTION

Requirements are a critical component in software development and automated software engineering, significantly influencing the project's success or failure [1],[2],[3],[16]. Requirement Engineering is essential to the software development process. Within this field, Requirements Analysis focuses on identifying user expectations for a new or updated product. This comprehensive process assesses whether the outlined requirements are thorough or incomplete [4]. User requirements fall into two categories: Functional Requirements (FR) and Non-Functional Requirements (NFR) [1]. FR define specific functions the software must perform, addressing the system's input and output behaviors. NFR, also known as quality requirements, pertain to the system's quality attributes, describing how it should function in a specific environment. Identifying NFR is particularly challenging, as they encompass diverse quality aspects such as performance, security, availability, aesthetics, fault tolerance, legal compliance, and operational characteristics [6], [7].

There are various techniques used in the extraction process of Non-Functional Requirements (NFR). Previous studies have demonstrated that most of the techniques used in the extraction process of NFR are supervised. However, the supervised learning approach is labor-intensive and has much overhead to train the model. If the training data are not available, then the domain experts will prepare training data manually. The analyst reads the requirement document and classifies NFR manually [2]. In case of a large dataset, extra effort is required to train the data and to get acceptable results. Furthermore, these systems are effective for small systems while facing challenges on a large scale or when the system is not well structured. The principal drawback of applying supervised methods to NFR detection is related to the amount of pre-categorized requirements needed to reach good levels of precision in the classification process.

Manually classifying NFRs is an onerous task, requiring domain-specific knowledge. It could be error prone and inefficient in large-scale projects. Complexity increases further with Arabic content because of its varied dialects, complex syntax, and deep culture. Traditional methods are manual and have mainly addressed the content classification problem. There is a limitation to manual and traditional contents classification, especially with little or no Arabic language datasets available for applying machine learning algorithms. Machine learning algorithms have been used in the requirement classification for some languages like English. To tackle these challenges, our research proposes a semi-automated approach using NLP tools specifically designed for Arabic. Leveraging the capabilities of CAMeL Tools, our approach aims to reduce the manual effort and time involved in classification.

We are developing a system that can automatically identify and classify NFRs in Arabic text with a high degree of accuracy as well as cultural sensitivity. This ground-breaking approach will transform the way NFRs are categorized, resulting in software development practices that are more inclusive, efficient and responsive to the needs of their global users, particularly Arabic speakers.

II. Literature Review

In this section, a review of research studies focused on automating the identification, classification, and analysis of Non-functional Requirements (NFR) is presented. Authors in [3] introduced an experiment aimed at categorizing NFRs into various groups, such as availability, security, usability, look and feel, legal and licensing, maintainability, operability, performance, scalability, fault tolerance, and portability, derived from the requirements document of an IoT-oriented healthcare system. Machine learning techniques and a hybrid KNN rule-based algorithm were considered for categorization, while Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) were employed for feature extraction. Additionally, a new dataset containing IoT-oriented healthcare system requirements was created for this purpose. However, due to the limited size of this dataset with only 104 requirements, generalizing the findings of this study might be challenging. In [4], the authors proposed a systematic approach to differentiate and categorize Non-Functional Requirements (NFR) through semantic and syntactic analysis using machine learning (ML) techniques applied to unrestricted documents. They utilized a dataset consisting of 79 unrestricted requirement reports in various formats. Four distinct natural language processing techniques, including statistical analysis and state-of-the-art semantic analysis provided by Google word2vec and bidirectional encoder representations from transformers models, were employed to extract features from requirement phrases. In [5], the authors adopted a semi-supervised machine learning approach that eliminates the need for training datasets. However, the Wikipedia data dump used to train their model can be considered somewhat supervised. To enhance extraction efficiency, the authors employed pre-processing techniques such as Part of Speech (POS) tagging and word augmentation. The identified NFR categories in this study included legal, look and feel, maintainability, operational, performance, scalability, security, usability, fault tolerance, and portability. The proposed method was evaluated using the tera-PROMISE and CCHIT datasets, with a focus on how pre-processing influenced NFR extraction performance. Initially, conventional pre-processing methods were applied, and data was assessed in terms of precision-recall. Subsequently, word augmentation and POS tagging pre-processing were implemented as the chosen approach, resulting in gradual performance improvement in NFR extraction. Overall, the final method outperformed other approaches, with average precision, recall, and F-measure values for the CCHIT dataset calculated at 55%, 85%, and 64%, respectively, and for the PROMISE dataset at 75%, 59%, and 64%, respectively. In [6], the author proposed empirical research, that four feature selection methods and seven machine learning algorithms were used to automatically categorize NFR into eleven categories. Precision, recall, F1-score, and accuracy of the classification results using all possible combinations of approaches and algorithms were all measured statistically during the study. The procedure compared Bag of Words (BoW) and Term Frequency-Inverse Document (TF-IDF) feature extraction methods with Naive Bayes (NB), K- Nearest Neighbors (KNN), Support Vector Machines (SVM), Stochastic Gradient Descent SVM (SGD SVM) and Decision Trees (D-tree) machine learning algorithms. The BoW, TF-IDF (character level), TF-IDF (word level), and N-gram feature extraction methods were used to evaluate each algorithm's performance in classifying software needs.

Authors in [7] used the user-reviews of the well-known App platforms, Apple App Store, Google Play, and Windows Phone Store, which have over 4 million apps for their classification. These reviews are considered very important for developers as they help them in the maintenance and evolution for the software. The authors classified these reviews into four categories of NFR as reliability, usability, portability, and performance. They combined four classification strategies: Bag of Words (Bow), Term Frequency-Inverse Document (TF-IDF), chi-square (CHI2), and AUR-Bow which were proposed in their work, with three machine learning algorithms including Naive Bayes, J48, and Bagging to classify user reviews. The authors conducted tests to compare the F-measures of the classification results. They found that the combination of AUR-Bow with Bagging accomplished the finest result, mainly a precision of 71.4%, a recall of 72.3%, and an F-measure of 71.8% among all the combinations. Finally, they concluded that user-reviews used in their study were better classified using the Bagging algorithm with Naïve Bayes and J48.

In [8], the authors implemented a multi-step unsupervised methodology to identify and classify NFRs into different categories. Early methods relied on manually classified data to train models, but large training datasets are often unavailable, limiting accuracy. To enhance NFR traceability, the authors extracted natural language content from source code and considered software requirements using word semantic similarity algorithms. The proposed unsupervised method demonstrated moderate complexity and scalability, functioning without the need for extensive datasets. Researchers in [9] proposed an automated method to detect NFRs using the Fuzzy Similarity Based K-nearest Neighbor (FSKNN) algorithm, which classifies requirement sentences without considering semantic variables and relatedness measures. Results showed that incorporating semantic parameters increased accuracy by 43.7%, compared to the fuzzy similarity-based K-nearest neighbor algorithm's accuracy of 41.4%. In [10], the authors developed a classification system for NFRs specific to Information Systems (IS). Existing classification schemes for NFRs did not address the needs of IS, web-based systems, or real-time systems. The proposed classification method organized NFRs in a tree-like structure, finding similar NFRs for both real systems and web-based systems. The study relied on the similarity of NFRs to identify different categories. Accuracy and confidentiality were common NFRs for both real-time systems and information systems, while interoperability and privacy were common to both web-based systems and information systems. Security, performance, and usability were shared NFRs for both real-time and web-based systems. Our work supports several research initiatives and enhancements using natural language processing with a focus on semi-automated approaches regarding Arabic user requirements [12]-[19].

III. Background

Requirements Engineering (RE) is a basic subject in software engineering which is the driving force of the requirement engineering habits in software engineering on which the excellent structure of software systems will imbue [20]. RE abides by a simple definition. It is a method of developing software repeatedly by tracing user's anticipations and requirements totally and carefully in order to prove that the ultimate product is what the users unwish. It is also the most essential stage inside the whole software development life cycle and is responsible for linking the Conceptual design of software system and its physical formation [21]. RE starts with the identification of stakeholders. Requirements are then further explored, constrained, and expressed as unambiguously as possible in order to be accurately represented as a software requirements specification (SRS), which is the primary RE documentation essential for the software system and includes functional and Non-functional requirements [22].

A. User Requirements Written in Arabic

Expressing software requirements in Arabic involves distinct challenges due to the language's complex morphology and syntax, cultural nuances, and regional dialects. The rich morphological structure of the Arabic language that introduces several word variations and, in many cases, meaning changes based on vocalization, introduces complexity in writing correct, clear, and unambiguous requirements [23]. The lack of vocalization in written text means that the same sentence can lead to several interpretations by the software developer, causing confusion regarding the functionality intended software. Additionally, the cultural aspects of Arabic; the fact that there are no direct equivalents to a number of technical terms in other languages plays a part in the challenge of translation and getting the terminologies right in software engineering. The presence of different dialects in the Arabic language also adds to the challenge as in one dialect you may be very clear in what you mean in terms of the requirement, but that requirement may be misunderstood in a different dialect, given rise to misunderstandings in the software of what should actually be done [23].

B. Natural Language Processing Tools

At the core of our research lie Natural Language Processing (NLP) tools, which are the technological enablers to analyze and understand human language with the help of any computational agents [24]. NLP tools are a specialized field of computer science, artificial intelligence, and linguistics that enables computers to process and interpret human language as it is spoken and deciphered, whether that language is in the form of Arabic speech or any type of written text. NLP is a critical component of our research work, as it will be the key player in successfully characterizing the Non-functional needs of Arabic users. NLP Technologies can play a large role in bridging the complexity of human languages, in this case Arabic, and the complexity of the revised world of software development. They can help us overcome the problems that Arabic language texts consist of, which include dialect varieties, tangled morphological functions, and cultural diversions. These techniques have made it possible to understand the text, harvest the relative information, and finally classify the Non-functional requirements. The goal of this research is to increase both the effectiveness and accuracy of our classification system using NLP technology, where the underlying system should be language and culture independent and, most importantly, international, which can eventually help both NLP and software development disciplines [25].

B. CAMeL Tools

The technology developed by CAMeL Tools goes above and beyond basic corpus lexicography. Arguably, groundbreaking even for corpus linguistics, CAMeL Tools sets new standards in Arabic Natural Language Processing (NLP) stemming from a computational perspective. It is about Arabic’s specificities in relation, among others, to its morphological system, its dialectal diversity, as well as its orthographic idiosyncrasy. CAMeL Tools doesn’t only perform basic analysis on text sets but also advanced operations such as morphological analysis, part-of-speech tagging, named entity recognition, or sentiment analysis [26]. Each tool in the suite has been developed in response to a need in Arabic language processing, harnessing the latest machine learning and deep learning techniques. For example, the morphological analyzer is crucial for a derivational language like Arabic to identify word roots and patterns. The tools have been trained on extensive corpora for Modern Standard Arabic as well as different regional dialects, so as to be effective in different contexts. Unique to CAMEL Tools is how they are developed. They are built by leveraging the open-source software development paradigm to actively invite researchers and developers across the globe to involve, adapt, refine, and optimize them with respect to real-life applications and user feedback, which creates an ecosystem that evolves and creates the tools continuously based on feedback and user satisfaction with proven workable performance metrics. CAMeL Tools value exists in various platforms that were utilized to engage from educational software helping people to learn Arabic and/or reconcile Arabic grammar exceptional cases to big data analytics platforms performing analytics on Arabic social media sentiment. Its development is a milestone in the Arabic NLP, bridging the human languages and computational understanding. The tool robustness and adaptability make inclusiveness and representation in the digital landscape specifically for the Arabic NLP, which is demanding and nuanced but not included in Arabic [26].

IV. PROPOSED CLASSIFICATION APPROACH

In our study we introduce a novel semi-supervised approach that is dedicated to classify NFRs in Arabic software documentation into seven main classes: Performance (PE), Security (SE), Availability (A), Look and Feel (LF), Fault Tolerance (FT), Legal (L), and Operational (O). Scalability class will be excluded in this study as it is overlapped with other classes. This section elucidates the approach for classifying user requirements into the different categories, leveraging the grammatical structure and keywords of Arabic sentences. Our methodology involves a thorough examination of various software graduation projects and Software Requirements Specifications (SRS) documents. From this analysis, we discerned distinctive attributes that enable the classification of different classes. These attributes form the basis for a set of heuristics in our approach. The process of analyzing Arabic sentences entails the utilization of CAMeL NLP tools, which facilitate parsing, tokenization, part-of-speech tagging, and sentence segmentation.

We utilize an empirical methodology detailed in Figure 1 to classify Non-functional Arabic user requirements. We devised a set of heuristics specifically tailored to categorize user requirements into seven NFR categories by analyzing features extracted from user requirements, leveraging Arabic grammar, analyzing Parts of Speech (PoS) tags, and compiling relevant NFR keywords. The process begins with inputting a collection of unclassified user requirements in Arabic. Initially, all requirements are normalized using CAMeL tools before being processed further. Tokens for all statements are then generated using the CAMeL tokens generator, followed by the generation of PoS tags for all words in the given sentence. Subsequently, the proposed heuristics are applied utilizing the generated PoS and tokens. Each sentence's classification involves comparing the NFR score with other pertinent metrics such as confidence factors. Ultimately, the output of the approach is a categorized collection of Non-functional Arabic user requirements.

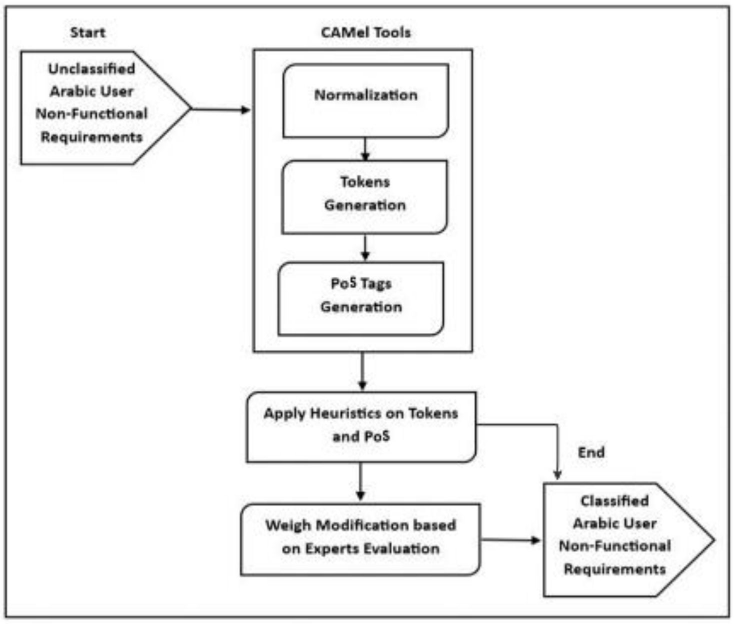
Start
Unclassified Arabic User Non-Functional Requirements
CAMel Tools
Normalization
Tokens Generation
PoS Tags Generation
Apply Heuristics on Tokens and PoS
Weigh Modification based on Experts Evaluation
End
Classified Arabic User Non-Functional Requirements

FIGURE 1. EMPIRICAL METHODOLOGY

A. The Proposed Heuristics

We developed a set of proposed heuristics organized in set of classes to handle Non-Functional User Requirements Linguistic Features as follows:

- **Class 1: Performance (PE)**

H#1: This Heuristic suggests the possibility of encountering common structural elements containing expressions such as "at an acceptable time" and "in the right time", namely "في وقت مقبول" and "في الوقت المناسب" "This heuristic is identified through an examination of diverse instances:

Example: "يجب أن يكون الموظفون قادرين على إكمال مجموعة من المهام في الوقت المناسب"

Translation: "Employees must be able to complete a set of tasks in a timely manner".

CAMeL Tokens: ['يجب', 'أن', 'يكون', 'الموظفون', 'قادرين', 'على', 'إكمال', 'مجموعة', 'من', 'المهام', 'في', 'الوقت', 'المناسب']

CAMeL PoS :['verb', 'conj_sub', 'verb', 'noun', 'adj', 'prep', 'noun', 'noun', 'prep', 'noun', 'prep', 'noun', 'adj']

H#2: If [‘digit’] or [‘noun num’] tag exists at sentence PoS, then it is more likely to be performance requirements.

The presence of numbers often signifies a high probability of a performance requirement. If numbers appear within sentences, regardless of their representation in numerical or alphabetical form, it tends to indicate that the sentence is likely a performance requirement. This distinction can be made based on their Part of Speech (PoS) tags. To determine the presence of numbers in a sentence, we need to identify all instances tagged as ['digit'] or ['noun num'].

A possible structure for requirements that show the locations of numbers in Arabic sentences:

(Verb + Subject + Object (1) | Object (2) | Object (3) -> Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | + name number + Noun | Digit + Noun) + (Noun | Preposition + Noun | name number + Noun | Digit + Noun) + (Noun | Preposition + Noun | name number + Noun | Digit + Noun)

CAMeL PoS Tags :

(Verb + (noun — pron) + (noun | prep + noun | noun num + noun | digit + noun) + (noun | prep + noun | noun num + noun | digit + noun) + (noun | prep + noun | noun num + noun | digit + noun).

Example:"يقوم النظام بتحديث العرض كل 60 ثانية“

Translation: "The system updates the display every 60 seconds".

CAMeL Tokens: ['يقوم', 'النظام', 'ب', 'تحديث', 'العرض', 'كل', '60', 'ثانية']

CAMeL PoS: ['verb', 'noun', 'prep', 'noun', 'noun', 'noun', 'num', 'noun']

H#3: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the performance requirements.

- **Class2: Security Requirements (SE)**

H#4: Security requirements establish the limitations and restrictions on system access to safeguard it from unauthorized entry .

Negative sentences could be categorized more closely with security requirements. This determination can be made by examining whether any negative prefixes are present in the sentence. The presence of negative prefixes in sentences, regardless of whether they are expressed numerically or alphabetically, increases the likelihood that the sentence falls under security requirements. To check for the presence of negative prefixes in a sentence, it is necessary to inspect all ['part neg'] tags.

Negative Arabic sentences is constructed by adding one of the following negation tools:

" لا، ليس، غير، لم، لمّا، لن، لام الجحود، ما "

The following sentences are examples of security requirements with negation tools:

a) Example: " لن يتمكن المستخدمون من الوصول المباشر إلى ملفات البيانات أو قواعد البيانات "

Translation: "Users will not be able to access data files or databases directly".

CAMeL Tokens['لن', 'يتمكن', 'المستخدمون', 'من', 'الوصول', 'المباشر', 'إلى', 'ملفات', 'البيانات', 'أو', : 'قواعد', 'البيانات']

CAMeL PoS: ['part_neg', 'verb', 'noun', 'prep', 'noun', 'adj', 'prep', 'noun', 'noun', 'conj', 'noun', 'noun'].

b) Example" لا يمكن إنشاء حساب مستخدم إلا بواسطة مسؤول النظام" :

Translation: "User accounts can only be created by the system administrator".

CAMeL Tokens: ['لا', 'يمكن', 'إنشاء', 'حساب', 'مستخدم', 'إلا', 'بواسطة', 'مسؤول', 'النظام']

CAMeL PoS: ['part_neg', 'verb', 'noun', 'noun', 'noun', 'conj', 'prep', 'noun', 'noun']

H#5: Conditional sentence is a linguistic structure that needs a tool to link two sentences, the first is a condition for the answer to the second, and it states that something happens because of something else associated with it and causes it.

Conditional sentences in Arabic are categorized into two types: Proof sentences and Negation sentences. The structure of these sentences comprises the Conditional Particle, the Conditional sentence, the Answer Particle, and the Conditional Answer:

1.Conditional Particle: Arabic Language utilizes two common conditional particles, namely "إذا" (idha) and "لو" (law) . These particles are represented by (subordinating conjunction) in CAMeL tools [‘conj’].

2. Conditional sentence: A conditional sentence is a verbal statement that falls into two categories: proof and negation sentence. A proof sentence consists of a conditional particle followed directly by the conditional sentence, without the presence of a negation particle .(لم) On the other hand, a negation sentence includes the negation particle (لم) after the conditional particle .

3. Answer Particle: The answer particle functions as an adverb for the conditional answer. In Arabic, the answer particles include(فان, سوف, فسوف) , with the corresponding tags being:

(” ف” connective particle +” إن” Pseudo verb): فإن

(” سوف” Future particle), (” ف” Response conditional): فسوف

(” سوف” Future particle): سوف

4. Conditional Answer: The conditional answer, a verbal sentence.

So, if the sentence structure as follow its more likely to be security requirement :

Subordinating Conjunction + (Verb | Negative Particle +Verb) + (Connective Particle + Pseudo Verb) | Future Particle | (Response Conditional+ Future Particle) + verbal sentence.

CAMeL PoS Tags:

(conj + (verb | part_ neg + verb) + (part_ rc + part_ emphac | part_ fut | part_ rc+ part_ fut) + (Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun))

Example: " إذا تم إبطال حساب مستخدم فلا يمكن إعادة إنشاء مثيل له إلا بواسطة مسؤول النظام "

Translation: "If a user account is deactivated, it cannot be re-created except by the system administrator".

CAMeL Tokens : ['إذا', 'تم', 'إبطال', 'حساب', 'مستخدم', 'فلا', 'يمكن', 'إعادة', 'إنشاء', 'مثيل', 'له', 'إلا', 'بواسطة', 'مسؤول', 'النظام']

CAMeL PoS: ['conj', 'verb', 'noun', 'noun', 'adj', 'part_neg', 'verb', 'noun', 'noun', 'noun', 'prep', 'part', 'noun', 'noun', 'noun']

Example: " سوف يتم الدخول للموقع الاكلينيكي اذا كان الشخص من طاقم التمريض فقط"

Translation: "Entry to the clinical site will only be allowed if the person is a member of the nursing staff".

CAMeL Tokens: ['سوف', 'يتم', 'الدخول', 'للموقع', 'الاكلينيكي', 'اذا', 'كان', 'الشخص', 'من', 'طاقم', 'التمريض', 'فقط']

CAMeL PoS: [' part_ fut ', 'verb', 'noun', 'noun', adj', 'conj', 'verb', 'noun', 'prep', 'noun', 'noun', 'adverb'].

H#6: After the study of different projects and SRS documents we noticed that there is a common structure repeated in the security requirements all have the word " access" as followed:

” أن” + Verb + Subject + Object (1) | Object (2) | Object (3) ->” أن” +” Verb” + (Noun | Pronoun) +” قادرا” + (Noun | Preposition + Noun) +” الوصول”

Token [0] = أن+ verb + (Noun | Pronoun) + Token [3] =" قادرا" + (Noun | Preposition + Noun) + Token [5] =" الوصول" + (Preposition + Noun)

Example: "أن يكون الطبيب قادرا على الوصول لكافة سجلات المرضى"

Translation: "The doctor should be able to access all patient records".

CAMeL Tokens['أن', 'يكون', 'الطبيب', 'قادرا', 'على', 'الوصول', 'لكافة', 'سجلات', 'المرضى'] :

CAMeL PoS: ['conj', 'verb', 'noun', 'adj', 'prep', 'verb', 'prep', 'noun', 'noun']

Verb + Subject + Object (1) | Object (2) | Object (3) -> (Verb) + (Noun | Pronoun) + (Noun | Preposition + Noun) + (Noun | Preposition + Noun) + (Noun | Preposition + Noun)

Token [0] = verb + (Noun | Pronoun) + (Noun | Preposition + Noun) +(Adjective | Adverb) + Token [5] =" الوصول" + (Preposition + Noun).

Example : " يستطيع أصحاب العقارات المسجلين فقط من الوصول الى النظام "

Translation: "Only registered property owners can access the system".

CAMeL Tokens: ['يستطيع', 'أصحاب', 'العقارات', 'المسجلين', 'فقط', 'من', 'الوصول', 'إلى', 'النظام']

CAMeL PoS: ['verb', 'noun', 'noun', 'adj', 'adv', 'prep', 'noun', 'prep', 'noun']

Subject + Verb + Object (1) | Object (2) | Object (3)-> (Noun | Pronoun) + Verb + (Noun | Preposition + Noun | Adverb + Noun | + Adjective) "الوصول" + (Noun | Preposition + Noun | Adverb + Noun | Adjective) + (Noun | Preposition + Noun | Adverb + Noun | Adjective).

Example:" الطلاب لا يستطيعون الوصول لشاشة تعديل العلامات"

Translation: "Students cannot access the grade editing screen".

CAMeL Tokens['الطلاب', 'لا', 'يستطيعون', 'الوصول', 'لشاشة', 'تعديل', 'العلامات'] :

CAMeL PoS: ['noun', 'neg', 'verb', 'noun', 'prep', 'noun', 'noun']

H#7: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the security requirements.

- **Class 3: Availability (A)**

H#8: When a sentence contains a percentage punctuation, it is more likely to indicate an availability requirement. The percent sign is easily distinguished from other punctuation marks by being preceded by a number. The percentage format is represented as [num, punc]. Therefore, if the part of speech is identified as 'num' and the '%' symbol is present in the token, the specified condition is met.

if pos == 'num' and '%' in token.

Example: " سيكون النظام متاحًا بنسبة 99٪ من الوقت خلال الأشهر الستة الأولى من التشغيل "

Translation: "The system will be available 99% of the time during the first six months of operation".

CAMeL Tokens: ['سيكون', 'النظام', 'متاحًا', 'بنسبة', '٩٩٪', 'من', 'الوقت', 'خلال', 'الأشهر', 'الستة', 'الأولى', 'من', 'التشغيل']

CAMeL PoS: ['verb', 'noun', 'adj', 'prep', 'num', 'punc', 'noun', 'prep', 'noun', 'adj', 'adj', 'prep', 'noun']

Example: " يجب أن لا يفشل المنتج أكثر من 2٪من الوقت المتاح على الانترنت "

Translation: "The product should not fail more than 2% of the available time online".

CAMeL Tokens: ['يجب', 'أن', 'لا', 'يفشل', 'المنتج', 'أكثر', 'من', ' 2 ', '٪', 'من', 'الوقت', 'المتاح', 'على', 'الانترنت']

CAMeL PoS: ['verb', 'conj_sub', 'neg', 'verb', 'noun', 'adj', 'prep', 'num', 'punc', 'prep', 'noun', 'adj', 'prep', 'noun']

H#9: A sentence indicating time duration typically adheres to the following structured formats :

Digit + punctuation + digit + token [] = صباحًا أو مساءً

The presence of such a structured sentence format is a strong indicator of an availability requirement. If the sentence follows the pattern of Digit + punctuation + digit + token [], it is more likely to convey a specific time duration, either in the morning (صباحًا) or evening.(مساءً)

Example: " يجب أن يكون النظام متاحًا للاستخدام بين الساعة 8:00 صباحًا و 6:00 مساءً "

Translation: "The system must be available for use between 8:00 AM and 6:00 PM".

CAMeL Tokens: ['يجب', 'أن', 'يكون', 'النظام', 'متاحًا', 'للاستخدام', 'بين', 'الساعة', '8', ':', '00', 'صباحًا', 'و', '6', ':', '00', 'مساءً']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'noun', 'adj', 'noun_prop', 'noun', 'noun', 'digit', 'punc', 'digit', 'noun', 'conj', 'digit', 'punc', 'digit', 'noun',]

Digit + token = صباحًا أو مساءً

Example: " يجب أن يكون المنتج متوفرا على الموقع يوميا من الساعة 9 صباحًا و لغاية الساعة 9 مساءً "

Translation: "The product must be available on the website daily from 9:00 AM to 9:00 PM".

CAMeL Tokens: ['يجب', 'أن', 'يكون', 'المنتج', 'متوفرًا', 'على', 'الموقع', 'يوميًا', 'من', 'الساعة', '9', 'صباحًا', 'و', 'لغاية', 'الساعة', '9', 'مساءً']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'noun', 'adj', 'prep', 'noun', 'adv', 'prep', 'noun', 'digit', 'noun', 'conj', 'noun', 'noun', 'digit', 'noun']

H#10: The presence of these linguistic elements indicates a heightened probability of conveying availability requirements.

" When a sentence is tagged with either 'adj' or 'adv' for its part of speech, it significantly raises the likelihood of expressing availability requirements." The sentence structure may take the form of a verbal or nominal sentence, as outlined below:

Verbal sentence format :

Verb + Subject + Object (1) | Object (2) | Object (3) -> Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun | + Adjective) + (Noun | Preposition + Noun | Adverb + Noun | Adjective) + (Noun | Preposition + Noun | Adverb + Noun | Adjective).

CAMeL PoS Tags: Verb + (noun | pron) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj)

Example: " يكون المنتج متاحًا خلال ساعات العمل العادية "

Translation: "The product is available during regular business hours".

CAMeL Tokens: ['يكون', 'المنتج', 'متاحًا', 'خلال', 'ساعات', 'العمل', 'العادية']

CAMeL PoS: ['verb', 'noun', 'adj', 'prep', 'noun', 'noun', 'adj']

Nominal sentence format :

Subject + Verb + Object (1) | Object (2) | Object (3) -> (Noun | Pronoun) + Verb + (Noun | Preposition + Noun | Adverb + Noun | + Adjective) + (Noun | Preposition + Noun | Adverb + Noun | Adjective) + (Noun | Preposition + Noun | Adverb + Noun | Adjective).

CAMeL PoS Tags: (noun | pron | foriegn) + Verb + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj)

Example: "فترة تعطل النظام قصيرة بحيث لا تزيد عن 10 دقائق في السنة"

Translation: "The system downtime period is short, not exceeding 10 minutes per year".

CAMeL Tokens: ['فترة', 'تعطل', 'النظام', 'قصيرة', 'بحيث', 'لا', 'تزيد', 'عن', '10', 'دقائق', 'في', 'السنة']

CAMeL PoS (Part of Speech): ['noun', 'verb', 'noun', 'adj', 'conj', 'neg', 'verb', 'prep', 'num', 'noun', 'prep', 'noun']

H#11: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the availability requirements.

- **Class 4: Look and feel (LF):**

H#12: Look and feel requirements, usually consider the unique needs associated with various nationalities and locations. These considerations involve recognizing the diverse cultural elements and geographical factors that shape user preferences. The words specific for names of countries, cities, or specific cultural terms are called proper nouns, and they are serving as linguistic tools that specifically denote to look and feel requirement. Therefore, the presence of the tag [noun_prop] enhances the probability of look and feel requirement.

Some examples show look and feel requirements that have proper nouns:

Example: " يجب أن يكون للموقع طابع أفريقي "

Translation: "The website should have an African character".

CAMeL Tokens: ['يجب', 'أن', 'يكون', 'للموقع', 'طابع', 'أفريقي']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'prep', 'noun', 'noun_prop']

Example: " يجب أن يتوافق المنتج مع إطار دليل تطوير التطبيقات في مدينة شيكاغو "

Translation: "The product must comply with the application development guidelines framework in the city of Chicago ".

CAMeL Tokens:['يجب', 'أن', 'يتوافق', 'المنتج', 'مع', 'إطار', 'دليل', 'تطوير', 'التطبيقات', 'في', 'مدينة', 'شيكاغو']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'noun', 'prep', 'noun', 'noun', 'noun', 'noun', 'prep', 'noun', 'noun_prop']

H#13: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the look and feel requirements.

- **Class 5: Fault Tolerance (FT)**

H#14: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the look and feel requirements.

- **Class 6: Legal (L)**

H#15: Through the study of different SRS documents, we notice that there a certain sentence structure is repeated in the legal requirement as illustrated bellow:

('verb', 'subordinating conjunction',): "يجب أن " + Subject + Object -> Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun).

CAMeL PoS Tags: Verb + (noun | pron) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv).

Example:" يجب أن يتوافق تطبيق المنازعات مع المتطلبات القانونية على النحو المحدد في لوائح التشغيل "

Translation: "The dispute resolution application must comply with the legal requirements as specified in the operating regulations".

CAMeL Tokens: ['يجب', 'أن', 'يتوافق', 'تطبيق', 'المنازعات', 'مع', 'المتطلبات', 'القانونية', 'على', 'النحو', 'المحدد', 'في', 'لوائح', 'التشغيل']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'noun', 'noun', 'prep', 'noun', 'adj', 'prep', 'noun', 'adj', 'prep', 'noun', 'noun']

Example: " يجب أن يتوافق المنتج مع لوائح التأمين المتعلقة بمعالجة المطالبات "

Translation: "The product must comply with the insurance regulations related to claims processing".

CAMeL Tokens: ['يجب', 'أن', 'يتوافق', 'المنتج', 'مع', 'لوائح', 'التأمين', 'المتعلقة', 'بمعالجة', 'المطالبات']

CAMeL PoS: ['verb', 'conj_sub', 'verb', 'noun', 'prep', 'noun', 'noun', 'adj', 'prep', 'noun', 'noun']

H#16: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the legal requirements.

- **Class 7: Operational (O)**

H#17: It is more likely to be operational requirement if the ['foreign'] tag is present at sentence PoS. Non-Arabic words frequently indicate programming languages or techniques (such as HTML, SQL, etc.). It is more likely that a sentence is a non-functional requirement if there are foreign words present in it, regardless of the sentence syntax. Therefore, based on their PoS tags, foreign words are able to distinguish the operational class.

There are several potential structures that indicate when foreign terms are used in Arabic sentences:

Verbal sentence :

Verb + Subject + Object (1) | Object (2) | Object (3) -> Verb + (Noun | Pronoun | Foriegn Word) + (Noun | Preposition + Noun |Preposition + Foriegn Word | Foriegn Word) + (Noun | Preposition + Noun | Preposition + Foriegn Word | Foriegn Word) + (Noun | Preposition + Noun | Preposition + Foriegn Word | Foriegn Word).

CAMeL PoSTags :Verb + (noun | pron | foriegn) + (noun | prep + noun | prep + foriegn | foriegn) + (noun | prep + noun | prep + foriegn | foriegn) + (noun | prep + noun | prep + foriegn | foriegn).

Example: " يتفاعل النظام مع أي متصفح HTML "

Translation: "The system interacts with any browser HTML".

CAMeL Tokens: ['يتفاعل', 'النظام', 'مع', 'أي', 'متصفح', 'HTML']

CAMeL PoS: ['verb', 'noun', 'prep', 'adj', 'noun', 'foriegn']

Nominal sentence :

Subject + Verb + Object (1) | Object (2) | Object (3) -> (Noun | Pronoun | Foriegn Word) + Verb + (Noun | Preposition + Noun | Foriegn Word | Preposition + Preposition) + (Noun | Preposition + Noun | Foriegn Word | Preposition + Preposition) + (Noun | Preposition + Noun | Preposition + Foriegn Word | Foriegn Word)

CAMeL PoS Tags :(noun | pron | foriegn) + verb + (noun | prep + noun | prep + foriegn | foriegn) + (noun | prep + noun | prep + foriegn | foriegn) + (noun | prep + noun | prep + foriegn | foriegn)

Example: " المنتج يجب أن يستخدم برنامج قواعد البيانات Oracle SQL Server "

Translation: "The product should use a database management program like Oracle SQL Server ".

CAMeL Tokens: ['المنتج', 'يجب', 'أن', 'يستخدم', 'برنامج', 'قواعد', 'البيانات', 'Oracle', 'SQL', 'Server']

CAMeL PoS: ['noun', 'verb', 'prep', 'verb', 'noun', 'noun', 'noun', 'foriegn', 'foriegn', 'foriegn']

H#18: Operational requirement is more likely if the primary actor is the "system "النظام" ,"or one of its Arabic synonyms like(الموقع, التطبيق, المنتج, البرنامج) :

The Arabic sentence could be verbal or nominal sentence, if the sentence is verbal then the main subject in the sentence will be: the system "النظام" ,that follow the main verb in the sentence. If the sentence is nominal, then the main subject in the sentence will be the system ,"النظام" ,as illustrated bellow:

Nominal sentence :

Subject + Verb + Object -> Subject +Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun)

CAMeL PoS: Subject +Verb + (noun | pron) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv)

Where token "النظام" = [0] or "المنتج" or "البرنامج" or "التطبيق" or"الموقع"

Example: " النظام يعمل على نسخ بيانات الأعمال احتياطيًا تلقائيًا واستعادتها عند الطلب "

Translation: "The system automatically backs up business data and restores it upon request".

CAMeL Tokens: ['النظام', 'يعمل', 'على', 'نسخ', 'بيانات', 'الأعمال', 'احتياطيًا', 'تلقائيًا', 'واستعادتها', 'عند', 'الطلب']

CAMeL PoS: ['noun', 'verb', 'prep', 'verb', 'noun', 'noun', 'adv', 'adv', 'conj_sub', 'prep', 'noun']

Verbal Sentence:

Verb + Subject + Object -> Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun)

CAMeL PoS: Verb + (noun | pron) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv).

Where token "النظام" = [1] or "المنتج" or "البرنامج" or "التطبيق" or"الموقع"

Example: (يعمل المنتج على الأجهزة الموجودة لجميع البيئات)

Translation: "The product works on the available devices for all environments".

CAMeL Tokens ['يعمل', 'المنتج', 'على', 'الأجهزة', 'الموجودة', 'لجميع', 'البيئات'] :

CAMeL PoS: ['verb', 'noun', 'prep', 'noun', 'adj', 'prep', 'noun']

H#19: Through the study of the different SRS for different projects we notice that there are many terms and words are repeated in the operational requirements such as:

Performance: استجابة Response, التحقق Verification, سرعة Speed, etc.

Security: استعلامات Inquiry, آمن Security, بيانات Data, etc.

Availability: اسبوع Week, الإنترنت Internet, تعطل Failure, etc.

Look and Feel: احترافي Professional, احترام Respect, جذاب Attractive, etc.

Fault Tolerance: استثناء Exception, استعادة Recovery, انقطاع Disconnection, etc.

Legal Requirements: الترخيص License, التوجيه Guidance, تشريع Legislation, etc.

Operational: الإلكتروني Electronic, البريد Mail, الخادم Server, etc.

V. Evaluation

A. *Heuristic Evaluation*

To enhance our method's accuracy, we enlisted the help of three seasoned software engineering specialists to evaluate it. Two of them have doctorates in software engineering, while the third is a distinguished engineer working for a top software engineering company. The experts provided a numerical percentage rating out of 100 for each heuristic as shown in table 1:

TABLE 1: EVALUATION OF HEURISTICS.

Class	Heuristic #	Average Percentage
Performance (PE)	1	85
	2	86.7
	3	81.7
Security (SE)	4	90
	5	83.3
	6	90
	7	90
Availability (A)	8	87.3
	9	87
	10	85.6
	11	87.7
Look and Feel (LF)	12	82.3
	13	85
Fault Tolerance (FT)	14	85
Legal (L)	15	83.3
	16	86.7
Operational (O)	17	81.7
	18	83.3
	19	81.6

Based on the above percentages we derive the confidence factor in the heuristic-based classification. It varies across several consistent categories. The average percentage of Performance (PE) shows a moderate to high confidence level. Both, Security (SE) and Availability (A) clearly show consistently high percentages, suggesting a high confidence level in the classification. Classes including Look and Feel (LF), Fault Tolerance (FT), Legal (L), and Operational (O) exhibit moderate percentages, indicating a moderate confidence in the classification within these areas.

B. Outcomes Evaluation

We used 105 requirement sentences from PROMISE Software Engineering Repository [27], which was divided into seven different classes: legal, operational, performance, security, availability, look and feel, and fault tolerance, which tested on our software.

a. Data Preparation

Prior to conducting the experiments, we divided the requirement sentences into individual classes. For each class, we created a separate CSV file containing the respective sentences. Additionally, we prepared another CSV file containing all the requirement sentences combined.

b. Experimental Procedure

1. Single- Class Testing: We started by giving individual tests to each class. In order to accomplish this, we fed the sentences from each class into a Python code that used the CAMeL tools to process the data and provide the appropriate PoS tags and tokens. We were able to evaluate the effectiveness of our method for each non-functional category separately since this process was performed for every class.

2. Multi- Class Testing: We then tested all the classes collectively as part of a thorough assessment. We delivered the CSV file comprising sentences from every class along with the Python code. After processing the combined data, the code classified the required statements into the appropriate non-functional classes using our suggested heuristics.

c. Result Analysis

Through this section we show and analyze the resulting classification report for single – class testing and multi-class testing.

1. Single-Class Testing

The Single- Class Testing Results give a thorough summary of how well each class performed in the examined system's categorization. The number of input sentences per class, the number of sentences that were successfully classified, and specific classification metrics: precision, recall, F1-score, and accuracy are summarized in table 2.

TABLE 2: SINGLE- CLASS TESTING RESULTS

#	**Class**	**# of Input Sentences**	**# of Correctly Classified Sentences**	**Classification Report**			
				Precision	Recall	F1- Score	Accuracy
1	Performance	19	16	1.00	0.84	0.91	0.84
2	Security	17	13	1.00	0.82	0.90	0.82
3	Availability	10	8	1.00	0.88	0.93	0.88
4	Look and feel	20	19	1.00	0.90	0.95	0.90
5	Fault tolerance	10	9	1.00	0.90	0.95	0.90
6	Legal	9	7	1.00	0.86	0.92	0.86
7	Operational	20	17	1.00	0.85	0.92	0.85

2. Multi- Class Testing

The Multi-Class Testing Results are shown in table 4, providing a detailed summary of how well each class performed overall in classifying the input sentences of the case study. It lists all of the input sentences in all classes, counts the number of sentences that were successfully classified, and provides comprehensive classification metrics including recall, precision, F1-score, and overall accuracy.

TABLE 4: MULTI-CLASS TESTING RESULTS

Class	# of Input Sentences	# of Correctly Classified Sentences	Classification Report			
			Average Precision	Average Recall	Average F1- Score	Overall Accuracy
All Classes	105	94	0.88	0.89	0.88	0.88

VI. Conclusion

This paper focuses on automating the categorization of Arabic user non-functional requirements using CAMeL tools, a natural language processing tool. Strategies were developed to classify these requirements effectively based on linguistic components like tokens, Part of Speech (PoS) tags, and lemmas. The Python-based approach aims to assist software engineers in handling Arabic user non-functional requirements more efficiently, reducing the time and resources needed for manual classification. By improving the accuracy and efficiency of analysis, the methodology empowers engineers to make better decisions, resulting in higher-quality software products. Additionally, the research contributes to advancing Arabic natural language processing, addressing unique linguistic challenges and laying the groundwork for future innovations. Overall, the project benefits software development practices and advances Arabic NLP, offering wide-ranging advantages for technological progress and societal impact in Arabic-speaking regions.

The future research directions outlined in our paper on semi-automated classification of non-functional Arabic user requirements include:

1. Heuristic refinement: Enhancing the heuristics developed in this paper to boost classification accuracy by incorporating additional language features, exploring different classification strategies, or refining the rules and criteria of the heuristics.

2. Data Collection Expansion: Gathering a larger and more diverse dataset of Arabic-language Software Requirements Specifications (SRS) to validate and enhance the methodology. This may involve collaborating with more organizations or software providers to obtain additional SRS projects in Arabic.

3. Testing on More Case Studies: Conducting further studies to evaluate the robustness and generalizability of the method by testing it on a broader range of SRS documents from different sectors and disciplines.

4. Creating Huge Datasets: Collaborating with user requirements researchers to develop substantial datasets of Arabic user requirements, which can facilitate the development of new fully automated categorization systems. These datasets can also be used to implement machine learning techniques to improve the efficiency and accuracy of classification.

These research areas aim to advance the current state of semi-automated classification of non-functional Arabic user requirements, ultimately enhancing the precision and effectiveness of software development processes in Arabic-speaking environments.

References

[1] Singh, P., Singh, D. and Sharma, A., (2016). "Rule-based system for automated classification of Non-functional requirements from requirement specifications," in International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 620-626, doi: 10.1109/ICACCI.2016.7732115.

[2] Sommerville, I., (2011). Software engineering, 9th Edition. ISBN10, 137035152, p.18.

[3] Khurshid, I., Imtiaz, S., Boulila, W., Khan, Z. Abbasi, A. Javed, A. and Jalil, Z. (2022). "Classification of Non-Functional Requirements From IoT Oriented Healthcare Requirement Document", in Front Public Health. 18;10:860536. doi: 10.3389/fpubh.2022.860536. PMID: 35372217; PMCID: PMC8974737.

[4] Shreda, Q. A. and Hanani, A. A. (2021). "Identifying Non-functional Requirements from Unconstrained Documents using Natural Language Processing and Machine Learning Approaches," in IEEE Access, doi: 10.1109/ACCESS.2021.3052921 .

[5] Younas, M., Jawawi, D.N.A., Ghani, I. et al. (2020). " Extraction of Non-functional requirement using semantic similarity distance" , in Neural Computer & Application 32, 7383–7397 .

[6] M. A. Haque, M. Abdur Rahman and M. S. Siddik ,(2019). "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), pp. 1-5, doi: 10.1109/ICASERT.2019.8934499.

[7] Mengmeng Lu., Peng Liang,(2017). " Automatic Classification of Non-Functional Requirements from Augmented App User Reviews", in EASE'17: Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering.

[8] Mahmoud, A., Williams, G., " Detecting, classifying, and tracing Non-functional software requirements", in Requirements Eng 21, 357–381 (2016).

[9] Ramadhani DA, Rochimah S, and Yuhana UL, (2015). " Classification of Non-functional requirements using semantic-FSKNN based ISO/IEC 9126", in TELKOMNIKA (Telecommunication Computing Electronics and Control), 13(4): 1456-1465.

[10] Gazi Y, Umar MS, and Sadiq M, (2015). "Classification of NFRs for information systems", in International Journal of Computer Applications, 115(22): 19-22.

[11] Awad, E., Khamayseh, F., and Arman, N., "Semi-Automated Classification of Non-Functional Arabic User Requirements using NLP Tools," in Proceedings of the 41st IBIMA Conference on Artificial Intelligence and Machine Learning, Granada, Spain, 26-27 June 2023, ISSN: 2767-9640

[12] Khamayseh, B., Arman, N. and Khamayseh, F., “Security Requirements Classification into Functional and Non-functional”, 40th IBIMA Computer Science Conference: 23-24, Granada Spain, November 2022

[13] Shehadeh, K., Arman, N., Khamayseh, F. "Semi-Automated Classification of Arabic User Requirements into Functional and Non-Functional Requirements using NLP Tools," 2021 International Conference on Information Technology (ICIT), 2021, pp. 527-532, doi: 10.1109/ICIT52682.2021.9491698.

[14] Jabbarin, S. and Arman, N., 2014, January. Constructing use case models from Arabic user requirements in a semi-automated approach. In 2014 World Congress on Computer Applications and Information Systems (WCCAIS) (pp. 1-4). IEEE.

[15] Arman, N. and Jabbarin, S., 2015. Generating use case models from Arabic user requirements in a semiautomated approach using a natural language processing tool. Journal of Intelligent Systems, 24(2), pp.277-286.

[16] Arman, N., 2015. Normalizer: a case tool to normalize relational database schemas, *Inform. Technol. J.***5** (2006), 329–331, ISSN: 1812-5638.

[17] Nassar, I.N. and Khamayseh, F.T., 2015, April. Constructing activity diagrams from Arabic user requirements using Natural Language Processing tool. In 2015 6th International Conference on Information and Communication Systems (ICICS) (pp. 50-54). IEEE.

[18] Alami, N., Arman, N. and Khamyseh, F., 2017, May. A semi-automated approach for generating sequence diagrams from Arabic user requirements using a natural language processing tool. In 2017 8th International Conference on Information Technology (ICIT) (pp. 309-314). IEEE.

[19] Alami, N., Arman, N. and Khamayseh, F., 2020. Generating sequence diagrams from Arabic user requirements using Mada+ Tokan tool. Int. Arab J. Inf. Technol., 17(1), pp.65-72.

[20] Adetoba, B., & Ogundele, I. (2018). Requirements engineering techniques in software development life cycle methods: A systematic literature review. International Journal of Advanced Research in Computer Engineering & Technology, 7(10), 733-743.

[21] Batool, I., Kosar, L., & Mehmood, M. Non-Functional Requirements As Constraints And Their Values In Software Development: A Review. 2018.

[22] Umar, M., & Khan, N. A. . Analyzing Non-Functional Requirements (NFRs) for software development. 2011 IEEE 2nd International Conference on Software Engineering and Service Science. IEEE. doi: 10.1109/ICSESS.2011.5982328.

[23] E. Gottesdiener, "Requirements by collaboration: getting it right the first time," in IEEE Software, vol. 20, no. 2, pp. 52-55, March-April 2003, doi: 10.1109/MS.2003.1184167.

[24] Khurana, D., Koli, A., Khatter, K., & Singh, S. (2023). Natural language processing: State of the art, current trends and challenges. Multimedia tools and applications, 82(3), 3713-3744.

[25] Wahdan, A., Al-Emran, M., & Shaalan, K. (2023). A systematic review of Arabic text classification: areas, applications, and future directions. Soft Computing, 1-22.

[26] Obeid, O., Zalmout, N., Khalifa, S., Taji, D., Oudah, M., Alhafni, B., ... & Habash, N. (2020, May). CAMeL tools: An open source python toolkit for Arabic natural language processing. In Proceedings of the Twelfth Language Resources and Evaluation Conference (pp. 7022-7032).

[27] Karim, S., Warnars, H. L. H. S., Gaol, F. L., Abdurachman, E., & Soewito, B. (2017, November). Software metrics for fault prediction using machine learning approaches: A literature review with PROMISE repository dataset. In 2017 IEEE international conference on cybernetics and computational intelligence (CyberneticsCom) (pp. 19-23). IEEE.