DOI: 10.15849/IJASCA.241130.01

Int. J. Advance Soft Compu. Appl, Vol. 16, No. 3, November2024
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Classification of Arabic User Requirements: A Semi-Automated Approach using NLP Tools

Karmel Shehadeh[1], Nabil Arman[2], and Faisal Khamayseh[3]

[1]College of Graduate Studies, Palestine Polytechnic University, Palestine
e-mail:kshehadeh@ppu.edu
[2]Department of Computer Science and Information Technology, Palestine
e-mail: narman@ppu.edu
[3]Department of Computer Science and Information Technology, Palestine
e-mail: faisal@ppu.edu

Abstract

In this paper, we present a new Semi-Automated classification approach of Arabic user requirements into functional and non-functional requirements using natural language processing (NLP) tools, namely CAMeL Tools. We proposed a set of heuristics based on basic constructs of Arabic sentences in order to extract information from software requirements written in Arabic to classify the requirements into functional and non-functional requirements. CAMeL tools are used to generate tokens, PoS tags, and lemmas of the parsed user requirements, then a set of proposed heuristic rules are applied to CAMeL outputs to determine the closest tagging class for each statement. We implemented the proposed approach using Python code with using CAMeL Tools 1.3.1, under Ubuntu 20.04 LTS. The proposed approach is validated using a set of experiments involving a set of real cases evaluated by a group of experts, graduate, and undergraduate students who are familiar with software requirements. The results showed that the proposed approach achieved better results in the classification of Arabic software requirements than classifications by students with an accuracy of 90%.

Keywords*:Requirements Classification, Automated Software Engineering, CAMeL Tools, Functional Requirements, Non-Functional Requirements.*

1 Introduction

Requirements engineering plays a significant role in the success of software development projects. One of the principal challenges is the classification of user requirements into functional and non-functional. Performing this task automatically can lower the cost and time of analysis of user requirements. Therefore, Automated Requirements classification is one of the most important research problems in software engineering, which has proven to be a suitable and effective field for automation [1].

There is a clear and unanimous definition of the Functional Requirements (FRs) and (Non-Functional Requirements (NFRs). FRs are statements about what services the system should provide, how it should act in specific situations and how it should respond to specific inputs. The system's functional requirements may also state what it should not do. On the other hand, NFRs are constraints and limitations on the system's services and

Received 10June2024; Accepted 2October 2024

functions. They include timing constraints, constraints imposed by standards, and constraints on the development process [2].

Since there is no direct automation procedure from Arabic requirements to classification, we proposed an approach to classify user requirements written in Arabic with minimal human interventions using an Arabic natural language processing tools namely CAMeL [28]. These tools are used to parse different statements of the user requirements written in Arabic language to distinguish between FRs and NFRs. A set of steps/heuristics that represent our approach for classifying user requirements is presented [3].

The main contribution of this research is introducing a new semi-automated approach for classifying user requirements written in Arabic with minimal human intervention. Our approach leverages both linguistic and contextual features to improve classification accuracy. This contribution is detailed in Sections 4, and 5.

The rest of the paper is organized as follows: section 2 presents the literature review and related works. Section 3 describes the researchbackground. Section 4 describes the researchmethodology used in the classification of Arabic user requirements. Section 5 presents the implementation and the validation of our proposed approach. Finally, sections 6 and 7 present the discussion, conclusions and future works.

2 Related Work

In the domain of requirements classification, several classification approaches have been proposed.Some researchers are interested in classifying non-functional requirements into different categories. Others are interested in classifying software requirements into functional and non-functional. This section presents some set of related research work in this area.

2.1 Software Requirements Classification Using Rule-Based Approaches

Singh et al. [4] combined automated software requirement identification and classification into NFR sub-classes with a rule-based classification technique based on thematic roles and determined the priority of extracted NFR based on their occurrence in multiple NFR classes. They used PROMISE corpus for creating Java rules. They used Concordia RE corpus to verify that their classifier works.

Hussain et al. [5] presented a methodology for automatic requirement classification using a text classifier with a part-of-speech (PoS) tagger. The authors proposed a set of characteristics of NFR, chose a list of syntactic features as candidates and tested their probabilities of occurrence in the collection of NFR sentences.

Sharma et al. [6] presented a pattern-based rule approach to automatically parse and classify non-functional requirements based on NLP. They relied on identifying NFRs by extracting multiple features and analyzing natural language requirements. In this approach, there is a certain combination of words where their relationships are unique for each category of NFR. The evaluation results conclude recall percentage between 60% and 85% for five categories of NFR.

Cleland-Huang et al. [7] presented an information retrieval approach to identify and classify Non-Functional Requirements automatically. This approach assumes that different categories of NFR are characterized by a set of distinct keywords. The proposed approach includes three phases, namely, mining, classification, and application.

2.2 Software Requirements Classification Using Machine Learning Approaches

Some approaches used machine learning for requirements classification. We divided them into researches that classified the requirements written in English and others classified the requirements written in German.

2.2.1 Software Requirements Classification of English Specifications

Kurtanovi´c and Maalej [8] investigated how accurate can automated classification of requirements into FR and NFR, in the dataset with supervised machine learning, be. They used a second RE17 data challenge dataset. They developed and evaluated a supervised machine learning approach using meta-data, syntactical, and lexical features. Haque et al. [9] proposed an automated NFR classification approach for quality software development by combining machine learning feature extraction and classification techniques. PROMISE software requirement dataset has been used. As a result, they recommended TF-IDF (character level) for feature extraction with SGD SVM algorithm to predict the best results in NFR classification. Younas et al. [10] proposed an approach that manipulates the textual semantic of functional requirements to identify the non-functional requirements. The semantic similarity is determined by the co-occurrence of patterns in large human knowledge repositories of Wikipedia. In this paper, they used a semi-supervised machine learning method. Therefore, there is no need for a training dataset. Abdur-Rahman et al. [11] suggested deep learning approaches using Recurrent Neural Network (RNN). They trained three different classifiers: RNN, GRU, and LSTM models. They achieved a high precision rate equal 0.961, 0.967 recall, and 0.966 f1-score. They found that RNN is a more effective approach to classify NFR than CNN and GRU approaches.

2.2.2 Software Requirements Classification of German Specifications

Ott [12] evaluates several classifiers that classify requirements written in German into topics. Each topic is manually defined as a group of keywords. The best classifier is accustomed to enable inspectors to reduce requirements' defects in parallel. In this research, the author focused on the multinomial naive Bayes (MNB) and therefore used the support vector machine (SVM) algorithms. The major problem is the difficulty of getting sufficient training examples so they improved the present recall to 0.8 and precision to 0.6. The previous research enhanced by Knauss & Ott [13] where they compared three classification approaches: manual, semi-automatic and automatic. Their research results showed that a semi-automatic approach is the most promising, as it provided the best ratio of quality and effort and achieved the best learning performance. Winkler et al. [14] presented an approach to automatically classify content elements of a natural language requirements specification as "requirement" or "information". This approach depends on convolution neural networks. In the evaluation of a real-world automotive requirements specification written in German, was able to detect requirements with a precision of 0.73 and a recall of 0.89.

2.3 Summary of Related Works

A summary of research work related to functional and non-functional classification is presented in Table 1. Most previous works dealt with requirements written in English language, and some of them in German language. To our knowledge, no research dealt with requirements written in Arabic language. Moreover, one of the limitations of supervised and semi-supervised machine learning is that they rely on robust datasets. On the other hand, deep learning requires significant computational resources. Our research addresses these gaps by providing a semi-

automated approach for classification of Arabic user requirements benefiting from linguistic and contextual features to enhance classification accuracy with minimal human intervention.

Table 1: Summary of Related Works

Research	Approach	Techniques/Features	Datasets	Results/Outcomes
Singh et al. [4]	Rule-Based	Thematic roles	PROMISE corpus, Concordia RE corpus	Verified classifier, prioritized NFRs based on occurrence
Hussain et al. [5]	Rule-Based	PoS tagger, syntactic features	Unspecified	Identified NFR characteristics, tested occurrence probabilities
Sharma et al. [6]	Pattern-Based Rule	NLP, feature extraction	Unspecified	Recall percentage between 60% and 85% for five NFR categories
Cleland-Huang et al. [7]	Information Retrieval	Distinct keywords	Unspecified	Three phases: mining, classification, and application
Kurtanović and Maalej [8]	Supervised Machine Learning	Metadata, syntactical, and lexical features	RE17 data challenge dataset	achieve precision and recall of up to ~92%
Haque et al. [9]	Supervised Machine Learning	TF-IDF (character level), SGD SVM algorithm	PROMISE software requirement dataset	Recommended best feature extraction and classification techniques
Younas et al. [10]	Semi-Supervised Machine Learning	Semantic similarity, co-occurrence patterns	Wikipedia (human knowledge repositories)	No need for training dataset, utilized large repositories
Abdur-Rahman et al. [11]	Deep Learning	RNN, GRU, LSTM models	Unspecified	High precision (0.961), recall (0.967), F1-score (0.966)
Ott [12]	Supervised Machine Learning	Multinomial Naive Bayes, SVM algorithms	German language specifications	Improved recall (0.8) and precision (0.6)
Knauss&Ott [13]	Comparison (Manual, Semi-Automatic, Automatic)	Mixed classification approaches	Unspecified	Semi-automatic approach provided best quality and effort ratio
Winkler et al. [14]	Deep Learning	Convolution Neural Networks	Automotive requirements (German)	Precision (0.73), recall (0.89)

3 Background

Arabic language is a prominent member of the Semitic Languages family. It consists of 28 letters and written from right to left. Grammar in Arabic language is a collection of rules that describes well informed sentences.

Arabic language presents unique challenges for natural language processing; the particularities of the Arabic language make it more ambiguous than other natural languages. This is due to its rich morphology, complex syntax, diverse dialects and semantic characteristics. In addition, Arabic words can take many forms depending on their grammatical roles, unlike English, where words have a relatively fixed structure. Second, there is a significant lack of digital resources of the Arabic language, especially concerning the grammars and corpora [15].

3.1 Arabic User Requirements

User and system requirements are usually written in natural languages supplemented by relevant diagrams and tables. It should be clear, unambiguous, complete, easy to understand, and consistent [2].

3.2 Natural Language Processing Tools

There are many tools for Arabic morphological analysis. Each has different characteristics. This is based on how these tools are developed and the database being used. There are several tools that are freely available and very suitable for tokenizing Arabic text: Stanford [16], MADA+TOKAN [17], MADAMIRA Tools [18], and

CAMEL [19]. Arabic language's rich morphology, complex syntax, diverse dialects, and semantic characteristics lead to unique challenges for natural language processing, leading to ambiguity and a lack of digital resources. User requirements written in Arabic must be clear and consistent. There are many tools for tokenizing Arabic text such as Stanford, MADA+TOKAN, MADAMIRA, and CAMEL.

4 The Research Approach

This section presents the classification approach of the software requirements into functional and non-functional. The new approach shows the steps of tokenizing and generating PoS of the Arabic user requirements using CAMeL Tools. We proposed a set of heuristics based on basic constructs of Arabic sentences in order to classify Arabic user requirements into FR and NFR.In this section, the proposed approach methodology is discussed.

Figure 1 describes the general proposed approach architecture. We developed a set of heuristics to classify the user requirements into functional and non-functional requirements by extracting user requirements features, using Arabic grammar, analyzing PoS tags and collecting FR and NFR keywords. The input of this approach is a set of unclassified user requirements, written in Arabic language, where the first step is to normalize all requirements using CAMeL tools before passing text to other CAMeL Tools. The second step is to generate tokens for all statements using CAMel tokens generator. The third step is to generate PoS tags for all words in given sentences. Then we apply the proposed heuristics using generated PoS and Tokens. We obtain each sentence class by comparing FR_Score with NFR_Score then compareFR-CF with NFR-CF. The output of our approach is a set of classified user requirements.

4.1 Arabic User Requirements Classification Approach

Software user requirements are often classified as functional requirements or non-functional requirements [2]. This section describes how the user requirements are automatically classified into functional and non-functional requirements based on Arabic sentence grammar and keywords. We reviewed several software graduations projects for Palestine Polytechnic University (PPU) students and SRS documents to extract features that distinguish functional from non-functional requirements. These features are used in proposing a set of heuristics for our approach. To analyze the Arabic sentence, we used CAMeL NLP tools for parsing, tokenization, part of speech, and sentence splitting. We extracted the features for each non-functional and functional requirement. We asked three software engineering experts to evaluate our approach, then averaged their evaluations as a Certainty Factor (CF) for each heuristic to make them more accurate. Here, we manually selected a cut-off threshold (> 0.75), where all the heuristics exceeding the cut-off threshold were selected as valid heuristics. As we can see in table2, the average rate for all heuristics is higher than 75%, except for heuristic#6. This heuristic was evaluated by experts as being less reliable than others, so based on their opinions we have modified this heuristic to be more accurate and reliable.

4.2 Proposed Heuristics

To classify the Arabic requirements into FR and NFR, we proposed a set of heuristics for Arabic sentences depending on the output of the CAMeL tools. We initialized score variables (F_scoreandNF_score), and Certainty Factor variables (F-CF and NF-CF) for each requirement. We have presented six heuristics that determine if the statement is NFR (H1- H6) and three heuristics that determine if the statement is FR (H7- H9). Each requirement statement is checked by all heuristics and every time the condition of the heuristic is satisfied, the value of F-CF or NF-CF is incremented, and the Certainty Factor value, presented in Table2 is added to F-CF or NF-CF depending on the requirement classification. Then the average value for CFsis calculated.

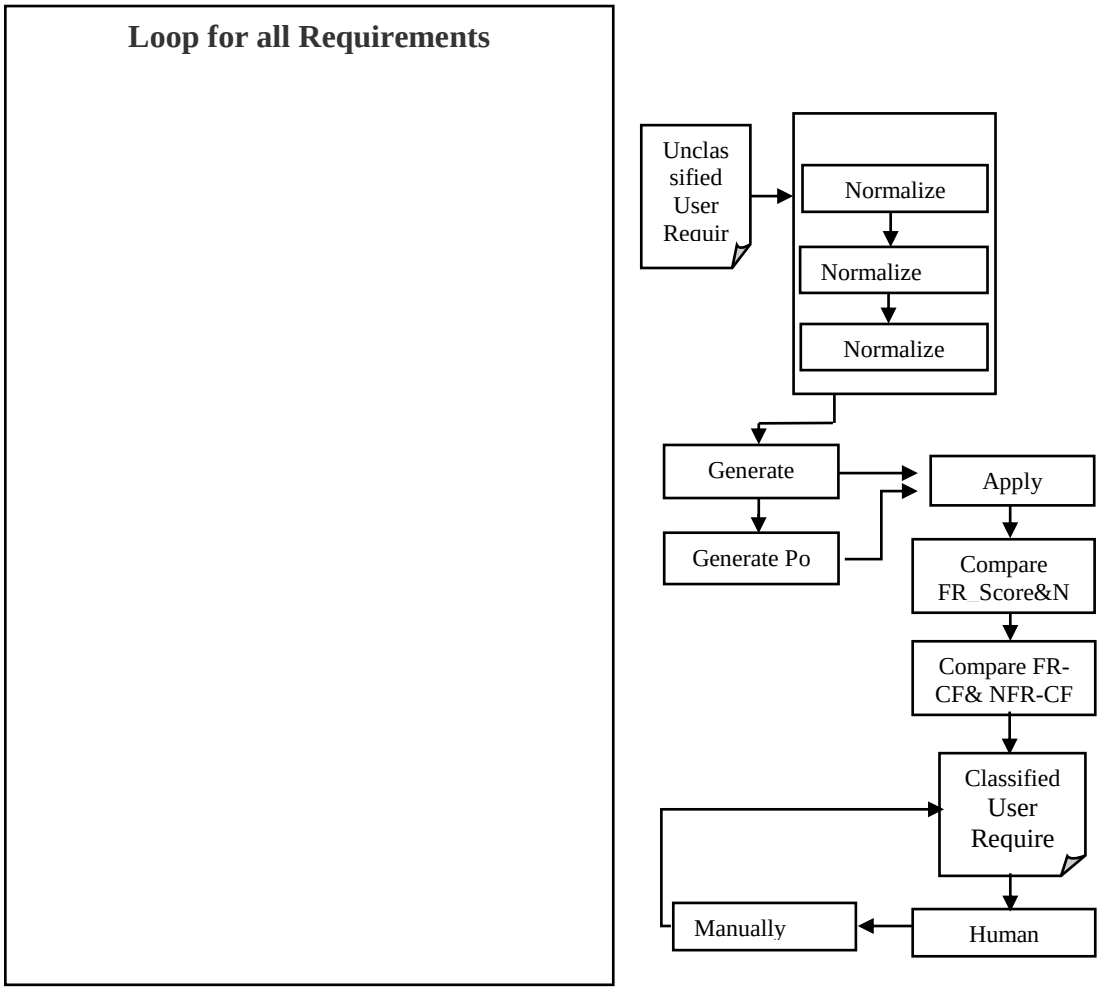

Figure 1. General proposed approach architecture.

To classify the statement, firstly the score variables F-score and NF-score are compared. Based on the result, the statement class is determined. Unless the scores are equal, the average certainty values are compared to decide the classification results. These heuristics are presented as follows:

Table 2: Expert Evaluations for proposed Heuristics.

Heuristics#	**F-CF**	**NF-CF**
H#1	21.67%	78.33%
H#2	20%	80%
H#3	14%	86%
H#4	10%	90%
H#5	17.34%	82.66%
H#6	46.7%	53.3%
H#7.1	88.33%	11.67%
H#7.1	88.33%	11.67%
H8	83.33%	16.67%
H9	76.66%	23.34%

H1: If [‘digit’] or [‘noun_num’] tag exists at sentence PoS, then it is more likely to be NFR.

The appearance of cardinals/numbers is indicating a high probability of NFR. In order to check whether cardinals /numbers are present in the sentence, we have to look for all [‘digit’] or [‘noun_num’] tags then add the certainty factor of this heuristic to the NFR CF for the certain sentence. If this condition applies to a sentence, the certainty factor of H1 is added to the NFR-

CFof this sentence. Some possible structures for requirements that show the locations of cardinals/numbers in Arabic sentences:

1. 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).

   CAMeLPoS 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 ثلاثين ثانية.

  CAMeL Tokens: [‘يقوم’, ‘النظام’, ‘ب’, ‘تحديث’, ‘العرض’, ‘كل’, ‘ثلاثين’, ‘ثانية’, ‘.’].

  CAMeLPoS: ['verb', 'noun', 'prep', 'noun', 'noun', 'noun_quant', 'noun_num', 'noun', 'punc'].

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

   CAMeLPoS Tags: (noun | pron) + Verb + (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:’’. النظام يقوم بتحديث العرض كل 30 ثانية’’

  CAMeL Tokens: [‘النظام’, ‘يقوم, ‘ب’, ‘تحديث’, ‘العرض’, ‘كل’, ‘30, ‘ثانية’, ‘.’].

  CAMeLPoS: ['noun', 'verb', 'prep', 'noun', 'noun', 'noun_quant', 'digit', 'noun', 'punc'].

H2: If [’foreign’] tag exists in the content of sentence PoS, then it is more likely to be NFR.

In order to check whether foreign words are present in the sentence, we have to look for all [’foreign’] tags and add the certainty factor of this heuristic to the NFR CF for the certain sentence. If this condition applies to a sentence, the certainty factor of H2 is added to the NFR-CF of this sentence. Some possible structures for requirements that show the locations of foreign words in Arabic sentence:

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

CAMeLPoS Tags: Verb + (noun | pron | forgan) + (noun | prep + noun | prep + Forgan | forgan) + (noun | prep + noun | prep + forgan | forgan) + (noun | prep + noun | prep + forgan | forgan).

- Example: Windows‘‘.XPيعمل البرنامج ضمن نظام التشغيل’’

  CAMeL Tokens: [‘يعمل’, ‘البرنامج’, ‘ضمن’, ‘نظام’, ‘التشغيل’, ‘Windows’, ‘XP’, ‘.’].

  CAMeLPoS: ['verb', 'noun', 'noun', 'noun', 'forgan', 'forgan', 'punc'].

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

CAMeLPoS Tags: (noun | pron | forgan) + verb + (noun | prep + noun | prep + forgan | forgan) + (noun | prep + noun | prep + forgan | forgan) + (noun | prep + noun | prep + forgan | forgan).

- Example: Windows ".XP البرنامج يعمل ضمن نظام التشغيل"

CAMeL Tokens: ['البرنامج', 'يعمل', 'ضمن', 'نظام', 'التشغيل', 'Windows', 'XP', '.'].

CAMeLPoS: ['noun', 'verb', 'noun', 'noun', 'forgan', 'forgan', 'pun'].

We excluded from this heuristic the words that appear in parentheses, where they are usually explanatory words for some terms that pertain to the system domain. In order to check the appearance of a foreign word between parentheses in the sentence, the first step is to look for this sequence of PoS tags ['punc', 'foreign', 'punc'].

- Example: "يقوم النظام بحساب عدد كريات الدم الحمراء (RBC)."

CAMEL Tokens: ['يقوم', 'النظام', 'ب', 'حساب', 'عدد', 'كريات', 'الدم', 'الحمراء', '(', 'RBC', ')', '.'].

CAMeLPoS: ['verb', 'noun', 'prep', 'noun', 'noun', 'noun_prop', 'noun', 'noun', 'punc', 'forgan', 'punc', 'punc'].

H3: If ['adj'] or ['adv'] tag exists at sentence PoS, then it is more likely to be NFR.

The appearance of adjectives/adverbs indicates a high probability of NFR. In order to check whether adjectives/adverbs are present in the sentence, we have to look for all ['adj'] or ['adv'] tags. If this condition applies to a sentence, the certainty factor of H3 is addedto the NFR-CF of this sentence. Some possible structures for requirements that show the locations of adjectives/ adverbs in Arabic sentences:

1. 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).

   CAMelPoS Tags: Verb + (noun | pron) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj).

   - Example: " يكون الموقع جذابًا لجميع الجماهير."

   CAMeL Tokens: ['يكون', 'الموقع', 'جذابا', 'ل', 'جميع', 'الجماهير','.'].

   CAMeLPoS: ['verb', 'noun', 'adj', 'prep', 'noun', 'noun', 'punc'].

2. 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)

   CAMeLPoS Tags: (noun | pron | forgan) + Verb + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj) + (noun | prep + noun | noun + adv | adj).

   - Example:"" النظام يستجيب بشكل سريع للحفاظ على البيانات المحدثة في الشاشة."

   CAMeL Tokens: ['المنتج', 'يستجيب', 'ب', 'شكل', 'سريع','ل','لحفاظ','على','البيانات','المحدثة','في', 'الشاشه','.']

   CAMeLPoS: ['noun', 'verb', 'prep', 'noun', 'adj', 'prep', 'noun', 'prep', 'noun', 'noun', 'perp', 'noun', 'punc'].

H4: If any of lemmas (the base form which is used to enter the word in a dictionary [22]) of Tokens[sentence] exists at NFR keywords lookup table lemmas, then it is more likely to be NFR.

We can find out whether the probability that this sentence is a non-functional requirement by returning the subjects/ object to their Lemma using CAMeL tools Lemmas, then comparing them with the lemma of keywords [12] and [20]. If this condition applies to a sentence, the certainty factor of H4 is added to the NFR-CF of this sentence.

- Example: ". يجب أن يكون المنتج قابلا للصيانة"

  CAMeL Tokens: ['يجب', 'ان', 'يكون', 'المنتج', 'قابلا', 'ل', 'صيانه','.'].

  CAMeL Lemma:['وجب', 'أنّ', 'كان', 'منتج', 'قابل', 'صيانه','.']

  CAMeLPoS: ['verb', 'conj_sub', 'verb', 'noun', 'adj', 'prep', 'noun', 'punc'].

H5: If ['part neg'] tag exists at sentence PoS, then it is more likely to be NFR.

In order to check whether negative prefixes are present in the sentence, we have to look for all ['part neg'] tags. The negation method in Arabic language is used to negate the sentence, whether it is verbal or nominal, using one of the negation tools ('لا', 'لما', 'لن', 'ليس', 'ما', 'لم'). If this condition applies to a sentence, then certainty factor of H5 is addedto the NFR-CF of this sentence.

- Example: ". النظام" لا يمكن انشاء حساب جديد الا بواسطة مسؤول

  CAMeL Tokens: ['لا', 'يمكن', 'انشاء', 'حساب', 'جديد', 'الا','ب','واسطة', 'مسؤول', 'النظام','.'].

  CAMeLPoS: [' part_neg ', 'verb', 'noun', 'noun', 'adj', 'part', 'prep', 'noun', 'noun', 'noun', 'punc'].

H6: If the main actor is the "system", or one of its synonyms in Arabic, then it is more likely to be NFR.

The system in Arabic could be: ('النظام', 'البرنامج', 'التطبيق', 'الموقع'). If the sentence is verbal, then the main actor is the first subject after the main verb. If this condition applies to a sentence, the certainty factor of H6 is added to the NFR-CF of this sentence. This heuristic is presented as follows:

('verb', 'conj sub',) Verb + Subject + Object -> Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun).

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

- Example:من 99 % متاحًا بنسبة الوقت."" يجب أن يكون المنتج

  CAMeL Tokens: ['يجب, 'ان, 'يكون, 'المنتج, 'متاحا', 'ب', 'نسبه','99', '%', 'من', 'الوقت', '.'].

  CAMeLPoS: ['verb', 'conj_sub', 'verb', 'noun', 'adj', 'prep','noun', 'digit', 'punc', 'prep', 'noun', 'punc'].

As we noticed from our reading of the non-functional requirements that we reviewed, many of them indicate that the actor is the system. When we asked to evaluate this rule by experts, it was rated less than 75%, so we added some conditions to make it more accurate and credible, as follows:

- In cases where the main verb in the sentence is "verb to be" that means it does not indicate an action, this is closer to non-functional requirements. For example, the system must be secure.
- In cases where the main verb denotes movement and action, and the actor is the system, this is almost closer to functional requirements. Such as: that the system calculates the rates.

H7: There are some common structures for Functional Requirements.

We concluded through our study of the SRS documents that there are certain structures that are repeated in the functional requirements sentences, which we explain as follows:

H7.1) ‘أن’+Verb + Subject + Object (1) | Object (2) | Object (3) -> ‘أن’+’يكون’+ (Noun | Pronoun) + ‘قادرا’ + (Noun | Preposition + Noun) + (Noun | Preposition + Noun).

CAMeLPoS Tags: tokens [0] = [‘أن’] + verb + (noun | pron) + tokens [1] = [‘يكون’] + tokens [3] = [‘قادرا’] + (noun | prep + noun) + (noun | prep + noun).

If this condition (the verb is’ يكون ‘, and the first object is ‘قادرا’), applies to a sentence, then the certainty factor of H7.1 is added to the FR-CF of this sentence.

- Example: “ أن يكون المشرف قادرا على تسجيل الطلاب.”

  CAMeL Tokens: [‘ان’, ‘يكون’, ‘المشرف’, ‘قادرا’, ‘على’, ’تسجيل’, ’الطلاب’,’.’].

  CAMeLPoS: ['conj_sub', 'verb', 'noun', 'adj', 'prep', 'noun', 'noun', 'punc'].

H7.2) ‘أن’+ ’يتمكن’ + Subject +Verb + Object (1) | Object (2) | Object (3) -> ‘أن’ + ’يتمكن’ + (Noun | Pronoun) + (Noun | Preposition + Noun) + (Noun | Preposition + Noun) + (Noun | Preposition + Noun).

CAMeLPoS Tags: tokens [0] = [‘أن’] + tokens [1] = [‘ يتمكن’] + verb + (noun | pron) + (noun | prep + noun) + (noun | prep + noun).

If this condition applies, where the verb is‘يتمكن’, the certainty factor of H7.2 is added beto the FR-CF of this sentence.

- Example: “أن يتمكن الطالب من تسجيل مساق.”

  CAMeL Tokens: [‘ان’, ‘يتمكن’, ‘الطالب’, ’من’, ’تسجيل’, ’مساق’,’.’].

  CAMeLPoS: ['conj_sub', 'verb', 'noun', 'prep', 'noun', 'noun', 'punc'].

H8: If any of lemmas of sentence Token exists at FR keywords lookup table lemmas (a table containing a set of words that are frequently used in functional requirements written in the basic forms of the word), then it is more likely to be FR. Some words are repeated in the main verb of functional requirements sentences. The main verb is usually the first verb (PoS: verb) in the sentence after the main actor (PoS: noun). Usually, the main verbs in functional requirements denote the movement “action”. We convert it to its lemma using CAMeL tools and verify it using the lookup table: (‘يبحث’, ’ينشئ’, ’يحسب’, ’يحذف’, ’يعرض’, ’يفعل’, ’يعبئ’, ’يحدث’, ’يعدل’, ’يسجل’, ’يضيف’). The main verb in the Arabic sentence can be known through the following linguistic structures:

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

CAMeLPoS: Verb + (noun | pron) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv) + (noun | prep + noun | noun + adv).

If this condition applies to a sentence, the certainty factor of H8is addedto the FR-CF of this sentence.

- Example: “. أن ينشئ المستخدم رسالة نصية ”

  CAMeL Tokens: [‘ان’, ‘ينشئ’, ’المستخدم’, ’رساله’, ’نصيه’,’.’].

CAMeL Lemma:['أَنَّ', 'أَنْشَأَ', 'مُسْتَخْدِم', 'رِسالَة', 'نَصّ', '.'].

CAMeLPoS: ['conj_sub', 'verb', 'noun', 'noun', 'noun, 'punc']

Also, this action can be contained in the form of a words (PoS: noun) in some sentence structures. This words usually found after the preposition in the sentence. After getting any of these words from the sentence based on the sentence structure and on the output of CAMeL tools, we convert it to its lemma using Camel tools and compare it from the lookup table: ('اضافة', 'تسجيل', 'تعديل', 'ارسال', 'اختيار', 'بحث', 'انشاء', 'حساب', 'حذف', 'عرض', 'تفعيل', 'تعبئة', 'تحديث').

- Example: ". أن يكون المستخدم قادرًا على إضافة صديق "

  CAMEL Tokens: ['ان', 'يكون', 'المستخدم', 'قادرا', 'على', 'اضافه', 'صديق', '.']

  CAMEL Lemma: ['أَنَّ', 'كان', 'المستخدم', 'قادِر', 'على', 'إضافَة', 'صِدّيق', '.']

  CAMEL PoS: ['conj_sub', 'verb', 'noun', 'adj', 'prep', 'noun', 'noun_prop', 'punc']

H9: If the sentence is a conditional sentence, it means that there is highly probability that it is a FR.

We propose a set of heuristics for conditional sentence. Conditional Arabic sentence's structure is:

[1]Conditional Particle + [2]Conditional sentence + [3]Answer Particle + [4]Conditional Answer
Conditional Arabic sentence's structure is:

1. Conditional particle: There are two common condition particles in Arabic Language ('لو', 'إذا'). These tags are (subordinating conjunction) in CAMeL tools ['conj'].
2. Conditional sentence: A conditional sentence is a verbal sentence. There are two types of the conditional sentences: proof and negation sentences.
3. Answer Particle: Answer particle is an adverb for the condition answer. Answer particles in Arabic language are: ('فإن', 'سوف', 'فسوف').The tags for them are: ('ف'connective particle + 'إن' Pseudo verb), ('سوف' Future particle), ('ف' Response conditional)
4. Conditional answer: The conditional answer is a verbal sentence.

The conditional sentences Tags:
Subordinating Conjunction + (Verb | Negative Particle +Verb) + (Connective Particle + Pseudo Verb) | Future Particle | (Response Conditional+ Future Particle) +verbal sentence.

CAMeLPoS Tags: conj + (verb | part neg + verb) + (part_rc + part_emphac | part_fut | part_rc + part_fut) + (Verb + (Noun | Pronoun) + (Noun | Preposition + Noun | Adverb + Noun)).

If this condition applies to a sentence, the certainty factor of H9 is added to the FR-CF of this sentence.

- Example: ""اذا كان عدد الطلاب في الشعبة أقل من الحد الأدنى سيتم حذفها."
  CAMeL Tokens: ['اذا', 'كان', 'عدد', 'الطلاب', 'في', 'الشعبه', 'اقل', 'من', 'الحد', 'الادنى', 'س', 'يتم', 'حذف', 'ها', '.'].

CAMeLPoS: ['conj', 'verb', 'noun', 'noun', 'prep', 'noun', 'noun', 'prep', 'noun', 'noun', 'part_fut', 'verb', 'noun', 'pron', 'punc'].

5Evaluation and Validation

In this section, we present the evaluation of our approach. The purpose of the evaluation is to calculate the accuracy of our approach and compare the results with classifications made by

graduate students. We use three threshold types of discriminator metrics to evaluate our approach. These metrics are Accuracy (Acc), Precision (P) and Recall (R) [21].User requirements for a set of cases are classified using the proposed approach of this research and by a set of graduate students. The comparison of the results in all cases has been done and the accuracy measurements have been calculated. We chose three SRS for open-source projects to test our approach. The SRS documents are written in English. Therefore, we asked a translation expert to translate the project's user requirements into Arabic. We chose ten functional requirements by selecting the first ten odd numbers, then we chose ten non-functional requirements in the same way to be our case studies. Afterwards, we asked a software engineering expert to reclassify the requirements to use their classification as a manually classified by experts for our experiments. Three case studies were given to five master students majoring in Informatics with good knowledge of software requirements. To evaluate our approach, we implemented a Python application using the CAMeL tools for pre-processing and part of speech tag generation. Our experiments were conducted using CAMeL Tools 1.3.1, and Python 3.6.7, under Ubuntu 20.04 LTS. The requirements were stored as a CSV file and the results were stored in another CSV file.The average measure metrics results of all case studies including accuracy, precision, and recall obtained from the manually classified by experts, the graduate students, and our approach are presented in Table 2.

Table 2: Measure metrics results.

	Project#1			Project#2			Project#3			Average		
Metrics	Experts	Graduate Students	Our Approach	Experts	Graduate Students	Our Approach	Experts	Graduate Students	Our Approach	Experts	Graduate Students	Our Approach
Accuracy	100%	93%	95%	100%	84%	85%	100%	87%	90%	100%	88.6%	90%
Precision (FR)	100%	90%	91%	100%	81%	100 %	100%	93%	88%	100%	88%	93%
Precision (NFR)	100%	93.8	100 %	100%	86%	76%	100%	82%	81%	100%	87.2%	85.88%
Recall (FR)	100%	94%	100 %	100%	88%	70%	100%	80%	80%	100%	87.3%	83.33%
Recall (NFR)	100%	90%	90%	100%	80%	100 %	100%	94%	90%	100%	88%	93.33%

Generally, we concluded from these results, that our approach achieved better classification of Arabic user requirements into functional and non-functional than classifications of graduate students. Additionally, our approach is more accurate in determining non-functional requirements than the functional ones. Hence, the average recall of functional is equal to 83.33% while the average recall of non-functional is equal to 93.33%.

6 Conclusion

In this paper, we proposed a novel approach for semi-automated classification of Arabic user requirements using natural language processing tool (CAMeL tools). A set of heuristics is presented to classify Arabic user requirements. These heuristics use the tokens, PoS tags and lemmas produced by CAMeL tools. We implemented the proposed approach using Python and CAMeL tools API. Results indicated that user requirements classified by our approach are highly consistent with the ones manually classified by the expert (benchmark). Results also showed that applying our approach leads to higher accuracy at non-functional requirements as compared to

the student's classification accuracy. However, our approach classification accuracy is slightly lower in classifying the functional requirements compared to the student's classification.

References

[1] Pérez-Verdejo, J., Sánchez-García, A. &Ocharán-Hernández, J. (2020). A Systematic Literature Review on Machine Learning for Automated Requirements Classification. In*8th International Conference in Software Engineering Research and Innovation (CONISOFT)*, (pp. 21-28).

[2] Sommerville I. (2011). Software Engineering, 9th Edition. Addison-Wesley, USA.

[3] Shehadeh, K., Arman, N. &Khamayseh, F. (2021). Semi-Automated Classification of Arabic User Requirements into Functional and Non-Functional Requirements using NLP Tools. In*Proc. 2021 International Conference on Information Technology (ICIT)*, (pp. 527-532).

[4] Singh, P., Singh, D. & Sharma, A. (2016). Rule-based system for automated classification of non-functional requirements from requirement specifications. In *Proc. 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. (pp. 620-626).

[5] Hussain, I., Kosseim, L. &Ormandjieva, O. (2008). Using linguistic knowledge to classify non-functional requirements in SRS documents. In *International Conference on Application of Natural Language to Information Systems*. (pp. 287-298).

[6] Sharma, V., Ramnani, R. &Sengupta, S. (2014). A framework for identifying and analyzing non-functional requirements from text. In *Proc. of the 4th international workshop on twin peaks of requirements and architecture*. (pp. 1-8).

[7] Cleland-Huang, J., Settimi, R., Zou, X. &Solc, P. (2006). The detection and classification of non-functional requirements with application to early aspects. In *Proc. 14th IEEE International Requirements Engineering Conference (RE'06)*. (pp. 39-48).IEEE.

[8] Kurtanović, Z. &Maalej, W. (2017). Automatically classifying functional and non-functional requirements using supervised machine learning. In *proc. 2017 IEEE 25th International Requirements Engineering Conference (RE)*. (pp. 490-495). IEEE.

[9] Haque, M., Rahman, M. &Siddik, M. (2019). Non-functional requirements classification with feature extraction and machine learning: An empirical study. In Proc. *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. (pp. 1-5). IEEE.

[10] Younas, M., Jawawi, D., Ghani, I. &Arif, M. (2020).Extraction of non-functional requirement using semantic similarity distance. *Neural Computing and Applications, 32*(*11)*, 7383-7397.

[11] Rahman, A., Haque, A., Tawhid, N. &Siddik, S. (2019). Classifying non-functional requirements using RNN variants for quality software development. In *Proc. of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation*. (pp. 25-30).

[12] Ott, D. (2013). Automatic requirement categorization of large natural language specifications at mercedes-benz for review improvements. In *Proc. International Working Conference on Requirements Engineering: Foundation for Software Quality*. (pp. 50-64).

[13] Knauss, E. &Ott, D. (2014). (Semi-) automatic categorization of natural language requirements. In *Proc. International Working Conference on Requirements Engineering: Foundation for Software Quality*. (pp. 39-54).

[14] Winkler, J. &Vogelsang, A. (2016). Automatic classification of requirements based on convolutional neural networks. In *Proc. 2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*. (pp. 39-45), IEEE.

[15] Khoufi, N., Aloulou, C. &Belguith, L. (2016). Parsing Arabic using induced probabilistic context free grammar." *International Journal of Speech Technology*, 19(2), 313-323.

[16] Manning, C., Surdeanu, M., Bauer, J.,Finkel, J., Bethard, S. &McClosky, D. (2014). The stanfordcorenlp natural language processing toolkit. In*Proc. 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14): System Demonstrations*.(pp.55–60).

[17] Habash, N., Rambow, O. & Roth, R. (2009). MADA+ TOKAN: A toolkit for Arabic tokenization, diacritization**,** morphological disambiguation, POS tagging, stemming and lemmatization. In *Proc. of the 2nd international conference on Arabic language resources and tools (MEDAR)*.(pp. 102-109).

[18] Al-Badrashiny, M., Eskander, R., Habash, N. &Rambow, O. (2014). Automatic transliteration of romanized dialectal Arabic. In *Proceedings of the eighteenth conference on computational natural language learning*.(pp. 30-38).

[19] Obeid, O., Zalmout, N., Khalifa, S., Taji, D., Oudah, M., Alhafni, B., Inoue, G., Eryani, F., Erdmann, A. &Habash, N. (2020). CAMeL tools: An open-source python toolkit for Arabic natural language processing. In *Proceedings of the 12th language resources and evaluation conference*. (pp. 7022-7032).

[20] Chung, L., Nixon, B., Yu, E. &Mylopoulos, J. (2012). Non-functional requirements in software engineering. International Series in Software Engineering (SOFT, volume 5).

[21] Hossin, M. &Sulaiman, M. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process. 5*(*2*). 1-11.

[22]Nordquist, R. (2019). Lemmas Explained. ThoughtCo. https://www.thoughtco.com/what-is-a-lemma-1691108. (Accessed: August 10, 2024).